یادگیری ژرف

رویکردی تئوری و کاربردی

مؤلف:

سینا رنجبر کوه فرهادی

DOI: 10.5281/zenodo.6672318

یادگیری ژرف

مؤلف:
سینا رنجبر کوه فرهادی
DOI: 10.5281/zenodo.6672318

فهرست مطالب

پیشگفتار ۱

فصل ۱ مقدمه یادگیری ژرف ۴

۱.۱ معرفی ۴

۱.۲ هوش مصنوعی ۴

۱.۳ نظریه آتوماتا ۸

۱.۴ نظریه بازی ۱۱

۱.۵ یادگیری ماشین ۱۱

۱.۶ نگاه کلی کتاب ۱۲

فصل ۲ جبر خطی ۱۴

۲.۱ مقدمه ۱۴

۲.۲ تعاریف اولیه ۱۴

۲.۳ عملیات اصلی در تانسورها و ماتریس ها ۱۶

۲.۳.۱ جمع و تفریق ماتریس با اسکالر ۱۶

۲.۳.۲ جمع و تفریق ماتریس با ماتریس ۱۶

۲.۳.۳ ترانهاده ماتریس ۱۶

۲.۳.۴ ضرب اسکالر در ماتریس ۱۷

۲.۳.۵ ضرب بردار در ماتریس ۱۷

۲.۳.۶ ضرب ماتریس در ماتریس ۱۷

۲.۴ فضاهای برداری ۱۸

۲.۴.۱ تعریف فضای برداری حقیقی ۱۸

۲.۴.۲ زیرفضاها ۱۹

۲.۴.۳ افراز فضا و وابستگی خطی ۱۹

۲.۵ دترمینان ۲۰

۲.۶ معکوس ماتریس ۲۱

۲.۶.۱ محاسبه معکوس ماتریس ۲۱
۲.۶.۲ ماتریس همانی ۲۱
۲.۶.۳ شبه معکوس ماتریس ۲۱
۲.۷ مقادیر ویژه و بردارهای ویژه ۲۲
۲.۷.۱ تجزیه ویژه ۲۲
۲.۷.۲ بردار ویژه و مقدار ویژه ۲۲
۲.۷.۳ محاسبه مقادیر و بردارهای ویژه ۲۳
۲.۸ نرم ۲۴
۲.۸.۱ نرم اقلیدسی ۲۴
۲.۸.۲ نرم یک ۲۵
۲.۸.۳ نرم بینهایت ۲۵
۲.۸.۴ نرم ماتریس ۲۵
۲.۹ خواص کلی ماتریس ها ۲۵
۲.۱۰ مسائل ۲۷
فصل ۳ آمار و احتمال ۲۸
۳.۱ مقدمه ۲۸
۳.۲ تعاریف آمار ۲۸
۳.۳ تعاریف احتمال ۲۹
۳.۴ انواع احتمال ۳۰
۳.۴.۱ احتمال ۳۰
۳.۴.۲ احتمال مشترک ۳۰
۳.۴.۳ احتمال شرطی ۳۱
۳.۵ امید ریاضی ۳۱
۳.۶ واریانس و انحراف معیار ۳۲
۳.۶.۱ تصحیح بسل ۳۳

۳.۷ توزیع احتمالاتی ۳۴
۳.۷.۱ توزیع یکنواخت ۳۴
۳.۷.۲ توزیع گوسی ۳۵
۳.۷.۳ توزیع برنولی ۳۵
۳.۷.۴ توابع مربوط به توزیع های احتمالاتی ۳۶
۳.۸ توزیع تجربی ۳۶
۳.۹ مسائل ۳۸
فصل ۴ داده کاوی ۳۹
۴.۱ مقدمه ۳۹
۴.۲ دسته بندی دادگان ۳۹
۴.۲.۱ دادگان ساختار یافته ۴۰
۴.۲.۲ دادگان غیرساختار یافته ۴۳
۴.۳ پردازش جریان ۴۳
۴.۴ پردازش دادگان حجیم ۴۴
۴.۵ رایانش موازی ۴۵
۴.۶ تئوری اطلاعات ۴۶
۴.۶.۱ آنتروپی ۴۶
۴.۶.۲ آنتروپی مشترک و آنتروپی شرطی ۴۷
۴.۶.۳ اطلاعات متقابل ۴۷
۴.۷ مسائل ۴۹
فصل ۵ پرداش سیگنال ۵۱
۵.۱ مقدمه ۵۱
۵.۲ حوزه تعریف سیگنال ۵۱
۵.۳ تبدیل حوزه سیگنال به حوزه فرکانس و بالعکس ۵۲
۵.۳.۱ تبدیل فوریه ۵۳

۵.۳.۲ تبدیل موجک ۵۶
۵.۴ فرکانس نمونه برداری سیگنال ۵۷
۵.۴.۱ پدیده تشدید در نمونه برداری ۶۰
۵.۵ ضرب پیچشی ۶۱
۵.۶ مسائل ۶۳
فصل ۶ یادگیری ماشین ۶۵
۶.۱ مقدمه ۶۵
۶.۲ پیش پردازش دادگان ۶۵
۶.۳ تبدیل و انتخاب ویژگی ۶۶
۶.۳.۱ تحلیل مولفه های اصلی ۶۸
۶.۳.۲ تحلیل مولفه های اصلی مبتنی بر کرنل ۶۹
۶.۴ مسائل مطرح در یادگیری ماشین ۶۹
۶.۴.۱ رگرسیون ۶۹
۶.۴.۲ طبقه بندی ۷۲
۶.۴.۳ مدل خود همبسته ۷۷
۶.۴.۴ خوشه بندی ۷۷
۶.۵ دسته بندی مدل های یادگیری ماشین ۷۸
۶.۵.۱ دسته بندی مدل ها براساس نحوه آموزش ۷۸
۶.۵.۲ مدل های مولد و متمایز کننده ۷۹
۶.۶ مسائل ۸۰
فصل ۷ شبکه های عصبی مصنوعی ۸۱
۷.۱ مقدمه ۸۱
۷.۲ مدل پرسپترون ۸۱
۷.۲.۱ توجیه استفاده از مدل پرسپترون ۸۵
۷.۲.۲ بررسی همگرائی و انتخاب نرخ آموزش ۸۶

۷.۲.۳ لزوم نرمال سازی ۸۷
۷.۲.۴ ارزیابی مدل ۸۸
۷.۳ تعریف یک شبکه عصبی ۸۹
۷.۴ بیش برازش و کم برازش ۹۴
۷.۵ مقادیر اولیه و توابع فعال ساز ۹۶
۷.۵.۱ روش Xavier ۹۸
۷.۵.۲ روش He ۹۸
۷.۶ انواع شبکه های عصبی ۹۹
۷.۶.۱ شبکه های عصبی مبتنی بر کرنل ۹۹
۷.۶.۲ شبکه های عصبی دارای نورون های انعطاف پذیر ۹۹
۷.۶.۳ شبکه های عصبی دارای نورون های راف ۱۰۰
۷.۷ مسائل ۱۰۲
فصل ۸ توابع هزینه و معیارهای ارزیابی ۱۰۳
۸.۱ مقدمه ۱۰۳
۸.۲ توابع هزینه ۱۰۳
۸.۲.۱ تنظیم کننده وزن ۱۰۶
۸.۲.۲ تابع هزینه مسئله طبقه بندی ۱۰۷
۸.۲.۳ توابع هزینه توزیع های احتمالاتی ۱۱۰
۸.۲.۴ تابع هزینه بر پایه آموزش عاطفی ۱۱۳
۸.۳ معیارها ۱۱۳
۸.۳.۱ خط رگرسیون ۱۱۴
۸.۳.۲ ماتریس آشفتگی ۱۱۵
۸.۳.۳ نمودار ROC ۱۱۸
۸.۴ دسته بندی نمونه های مجموعه دادگان ۱۱۹
۸.۴.۱ ارزیابی متقابل K-fold ۱۲۰

۸.۴.۲ ارزیابی متقابل Leave one out ۱۲۱
۸.۵ مسائل ۱۲۲
فصل ۹ روش های بهینه سازی در یادگیری ژرف ۱۲۳
۹.۱ مقدمه ۱۲۳
۹.۲ حالت های مختلف پیاده سازی روش های بهینه سازی ۱۲۳
۹.۲.۱ روش تصادفی ۱۲۳
۹.۲.۲ روش دسته ای ۱۲۴
۹.۲.۳ روش دسته ای کوچک ۱۲۴
۹.۳ الگوریتم های بهینه سازی ۱۲۵
۹.۳.۱ گرادیان نزولی ۱۲۵
۹.۳.۲ NAG ۱۲۶
۹.۳.۳ Adagrad ۱۲۶
۹.۳.۴ RMSprop ۱۲۷
۹.۳.۵ Adadelta ۱۲۸
۹.۳.۶ Adam ۱۲۹
۹.۳.۷ گرادیان نزولی طبیعی ۱۳۰
۹.۴ مسائل ۱۳۳
فصل ۱۰ یادگیری بازنمائی ۱۳۴
۱۰.۱ مقدمه ۱۳۴
۱۰.۲ ماشین بولتزمن ۱۳۸
۱۰.۲.۱ تابع هزینه در ماشین بولتزمن ۱۴۰
۱۰.۲.۲ آموزش ماشین بولتزمن به روش قاعده هبیان بر اساس گرادیان نزولی ۱۴۱
۱۰.۲.۳ ماشین های بولتزمن با واحدهای پنهان و نمایان ۱۴۳
۱۰.۲.۴ ماشین بولتزمن به عنوان یک روش متمایزکننده ۱۵۰
۱۰.۲.۵ تابع هزینه در های ماشین بولتزمن متمایزکننده ۱۵۲

۱۰.۲.۶ آموزش ماشـــین بولتزمن متمایرکننده به روش قاعده هبیان بر اســـاس گرادیان نزولی ۱۵۴
۱۰.۲.۷ مشکلات و چالش های آموزش ماشین های بولتزمن ۱۵۶
۱۰.۲.۸ ماشین های بولتزمن محدود شده ۱۵۷
۱۰.۲.۹ ماشین های بولتزمن محدود شده ژرف و شبکه باور عمیق ۱۵۹
۱۰.۳ خود رمزگذارها ۱۶۴
۱۰.۳.۱ روابط پیشرو خودرمزگذارها ۱۶۵
۱۰.۳.۲ تابع هزینه خود رمزگذارها ۱۶۶
۱۰.۳.۳ آموزش خود رمزگذارها به روش گرادیان نزولی ۱۶۶
۱۰.۳.۴ آموزش سراسری و محلی در خودرمزگذارها ۱۷۰
۱۰.۳.۵ خود رمزگذارهای پشته ای ۱۷۱
۱۰.۳.۶ چالش ها و مشکلات آموزش خود رمزگذارها ۱۷۲
۱۰.۳.۷ تنک زایی در خود رمزگذارها ۱۷۴
۱۰.۳.۸ خود رمزگذار نویززدا ۱۷۷
۱۰.۳.۹ خود رمزگذار انقباضی ۱۷۸
۱۰.۳.۱۰ خود رمزگذار متغیر ۱۸۰
۱۰.۳.۱۱ خود رمزگذار واسا اشتاین ۱۹۷
۱۰.۳.۱۲ خود رمزگذارهای راف ۲۰۱
۱۰.۳.۱۳ خود رمزگذارها با آموزش عاطفی ۲۰۳
۱۰.۳.۱۴ خود رمزگذارهای بازگشتی ۲۰۳
۱۰.۴ مسائل ۲۰۸
فصل ۱۱ سری های زمانی و شبکه های عصبی بازگشتی ۲۰۹
۱۱.۱ مقدمه ۲۰۹
۱۱.۲ ساختار دادگان با وابستگی زمانی ۲۱۰
۱۱.۲.۱ ایجاد مجموعه دادگان آموزشی با استفاده از نمونه های یک سری زمانی ۲۱۲

۱۱.۲.۲ استفاده از چند سری زمانی در مدلهای خود هم بسته..... ۲۱۴
۱۱.۲.۳ انتخاب مدل مناسب برای آموزش مدل پیش بینی کننده سری زمانی..... ۲۲۳
۱۱.۳ شبکه های هاپفیلد..... ۲۲۵
۱۱.۳.۱ ذخیره یک الگو (تصویر) در ساختار شبکه هاپفیلد..... ۲۲۵
۱۱.۳.۲ بازیابی الگو (تصویر) ذخیره شده در ساختار شبکه هاپفیلد..... ۲۳۰
۱۱.۴ شبکه های بازگشتی جردن..... ۲۳۲
۱۱.۵ شبکه های بازگشتی المن..... ۲۳۵
۱۱.۶ شبکه های بازگشتی المن–جردن..... ۲۳۷
۱۱.۷ ساختار نورون ها با پسخور محلی..... ۲۴۱
۱۱.۸ پس انتشار در طول زمان..... ۲۴۴
۱۱.۹ واحد دروازه ای LSTM..... ۲۴۵
۱۱.۹.۱ بردار حافظه بلند مدت..... ۲۴۶
۱۱.۹.۲ بردار ورودی و خروجی و حافظه کوتاه مدت..... ۲۴۷
۱۱.۹.۳ گیت فراموشی..... ۲۴۸
۱۱.۹.۴ گیت ورودی..... ۲۴۹
۱۱.۹.۵ گیت خروجی..... ۲۵۱
۱۱.۹.۶ آموزش یک واحد LSTM با الگوریتم پس انتشار خطا به روش گرادیان نزولی۲۵۳
۱۱.۹.۷ انواع مختلف واحدهای LSTM..... ۲۵۵
۱۱.۱۰ واحد دروازه ای GRU..... ۲۵۸
۱۱.۱۰.۱ بردار ورودی، خروجی و حافظه..... ۲۵۹
۱۱.۱۰.۲ گیت بازنشانی..... ۲۶۰
۱۱.۱۰.۳ گیت به روزرسانی..... ۲۶۱
۱۱.۱۰.۴ آموزش یک واحد GRU با الگوریتم پس انتشار خطا به روش گرادیان نزولی۲۶۲
۱۱.۱۱ لایه های پیچشی در دادگان زمانی..... ۲۶۴
۱۱.۱۱.۱ نحوه حرکت یک کرنل به روی نمونه ورودی و گام حرکتی..... ۲۶۶

۱۱.۱۱.۲ روابط پیشرو در ضرب پیچشی یک بعدی ۲۶۹
۱۱.۱۱.۳ بعد گذاری ۲۷۰
۱۱.۱۱.۴ آموزش کرنل یک بعدی پیچشی به روش گرادیان نزولی ۲۷۱
۱۱.۱۱.۵ لایه تجمیع در ساختارهای پیچشی ۲۷۳
۱۱.۱۲ شبکه های شبه بازگشتی ۲۷۵
۱۱.۱۳ شبکه های هاپفیلد مدرن ۲۷۹
۱۱.۱۳.۱ شبکه هاپفیلد مدرن در ساختارهای عمیق ۲۸۲
۱۱.۱۴ رایانش مخرنی ۲۸۴
۱۱.۱۵ مسائل ۲۸۵
فصل ۱۲ شبکه های عصبی پیچشی ۲۸۶
۱۲.۱ مقدمه ۲۸۶
۱۲.۲ لزوم استفاده از ساختارهای پیچشی ۲۸۶
۱۲.۳ ضرب پیچشی دو بعدی ۲۸۸
۱۲.۳.۱ روابط پیشرو ضرب پیچشی دو بعدی ۲۸۹
۱۲.۳.۲ توابع فعال ساز در ساختارهای پیچشی دو بعدی ۲۹۲
۱۲.۳.۳ لایه پیچشی دو بعدی ۲۹۷
۱۲.۳.۴ بعد گذاری ۲۹۸
۱۲.۴ لایه تجمیع دو بعدی ۳۰۱
۱۲.۴.۱ تجمیع حداکثری دو بعدی ۳۰۲
۱۲.۴.۲ تجمیع میانگین ۳۰۳
۱۲.۴.۳ مشتقات لایه تجمیع ۳۰۳
۱۲.۵ نرمال سازی دسته ای ۳۰۴
۱۲.۶ نرمال سازی لایه ای ۳۰۸
۱۲.۷ نرمال سازی پاسخ محلی ۳۱۰
۱۲.۷.۱ روش نرمال سازی داخل هر کانال ۳۱۰

۱۲.۷.۲ روش نرمال سازی بین کانال ها ۳۱۱
۱۲.۸ لایه مسطح ساز ۳۱۱
۱۲.۹ لایه تجمیع میانگین سراسری دو بعدی ۳۱۳
۱۲.۱۰ ضرب پیچشی دو بعدی ترانهاده ۳۱۵
۱۲.۱۰.۱ ضرب پیچشی دو بعدی با اتساع ۳۱۹
۱۲.۱۰.۲ ضرب پیچشی دو بعدی ترانهاده به عنوان ضرب پیچشی ۳۲۲
۱۲.۱۱ ضرب پیچشی سه بعدی ۳۲۳
۱۲.۱۱.۱ بعد گذاری در ضرب پیچشی سه بعدی ۳۲۶
۱۲.۱۱.۲ لایه تجمیع سه بعدی ۳۲۷
۱۲.۱۱.۳ لایه تجمیع سراسری سه بعدی ۳۲۷
۱۲.۱۱.۴ اتساع سه بعدی ۳۲۸
۱۲.۱۲ ضرب پیچشی جدائی پذیر فضائی ۳۲۸
۱۲.۱۳ ضرب پیچشی جدائی پذیر مبتنی بر عمق ۳۳۱
۱۲.۱۴ اتصالات جهش ۳۳۴
۱۲.۱۵ ساختارهای پیچشی و کاربرد آن ها ۳۳۶
۱۲.۱۵.۱ ساختارهای پیچشی با لایه های تماما متصل ۳۳۶
۱۲.۱۵.۲ ساختارهای پیچشی با خروجی متغیر (ساختارهای پیچشی مبتنی بر ناحیه) ۳۵۳
۱۲.۱۵.۳ ساختارهای پیچشی با خروجی تانسور (تصویر) ۳۷۳
۱۲.۱۵.۴ ساختارهای پیچشی-بازگشتی ۳۸۰
۱۲.۱۵.۵ مکانیزم توجه در ساختارهای پیچشی ۳۸۴
۱۲.۱۶ داده افزائی در ساختارهای پیچشی ۳۹۱
۱۲.۱۷ نگاشت فعال سازی کلاس ۳۹۳
۱۲.۱۸ مسائل ۳۹۵
فصل ۱۳ شبکه های مولد ۳۹۶

۱۳.۱ مقدمه ۳۹۶
۱۳.۲ شبکه های متخاصم مولد ۳۹۶
۱۳.۲.۱ روند پیشرو در شبکه های متخاصم مولد ۳۹۷
۱۳.۲.۲ تابع هزینه شبکه های متخاصم مولد ۳۹۸
۱۳.۲.۳ آموزش شبکه های متخاصم مولد ۳۹۹
۱۳.۲.۴ کاربرد شبکه های مولد متخاصم ۴۰۰
۱۳.۲.۵ ساختارهای شبکه های مولد متخاصم ۴۰۱
۱۳.۳ ترکیب خودرمزگذارها و شبکه های مولد متخاصم ۴۰۶
۱۳.۴ مسائل ۴۰۷
فصل ۱۴ مدل های زبانی و مکانیزم توجه ۴۰۸
۱۴.۱ مقدمه ۴۰۸
۱۴.۲ مدل های زبانی ۴۰۸
۱۴.۲.۱ معایب مدل های دنباله به دنباله ۴۱۱
۱۴.۲.۲ LSTM دو جهته ۴۱۲
۱۴.۳ مکانیزیم توجه ۴۱۳
۱۴.۳.۱ مکانیزم خود محور ۴۱۵
۱۴.۴ مسائل ۴۲۰
فصل ۱۵ تعمیم دامنه و یادگیری انتقالی ۴۲۱
۱۵.۱ مقدمه ۴۲۱
۱۵.۲ تعریف کلی یادگیری انتقالی ۴۲۱
۱۵.۲.۱ یادگیری انتقالی در ساختارهای پیچشی ۴۲۵
۱۵.۲.۲ یادگیری انتقالی در مدل های زبانی ۴۲۶
۱۵.۳ یادگیری N-Shot ۴۲۶
۱۵.۳.۱ تعیین مراکز کلاس ها با استفاده فراداده ۴۲۹
۱۵.۴ مسائل ۴۳۰

فصل ۱۶ رویکردهای جستجوی ساختار عصبی ۴۳۱

۱۶.۱ مقدمه..... ۴۳۱

۱۶.۲ رویکردهای تعیین ساختار عصبی بر پایه یادگیری..... ۴۳۱

۱۶.۲.۱ الگوریتم ژنتیک..... ۴۳۲

۱۶.۳ رویکردهای تعیین ساختار عصبی بر پایه روش های جستجو..... ۴۳۶

۱۶.۴ رویکردهای خود سازمانده در تعیین ساختار عصبی بهینه..... ۴۳۷

۱۶.۵ رویکرد جستجوی ساختار عصبی مشتق پذیر..... ۴۳۷

۱۶.۶ مسائل..... ۴۳۹

۱۷ پیوست ۱ ..... ۴۴۰

۱۷.۱ پیچیدگی محاسباتی..... ۴۴۰

۱۷.۲ روند مقعر محدب..... ۴۴۰

۱۸ مراجع ۴۴۳

فهرست سرنام‌های کتاب

AI	Artificial Intelligence
ANN	Artificial Neural Network
AT	Attention Mechanism
CAM	Class Activaion Mapping
CC	Computational Complexity
CCCP	The Concave-Convex Procedure
CDF	Cumulative Distribution Function
CNN	Convolutional Neural Network
DANN	Domain Adversarial Neural Network
DAP	Deep Adaption Network
DARTS	Differentiable Architecture Search
DRL	Deep Reinforcment Learning
ECG	Electrocardiography
EEG	Electroencephalography
ELBO	Evidence Lower Bound
FPN	Feature Pyramid Network
GAN	Generative Adversarial Network
GAP	Global Average Pooling
GPU	Graphics Processing Unit
HOG	Histogram Of Oriented Gradients
ML	Machine Learning
MLP	Multi-Layer Perceptron
MSE	Mean Squared Error
NAS	Neural Architecture Search
NLL	Negative Log-Likelihood
NLP	Natural Language Processing
NMS	Non-Maximal Suppression
OCR	Optical Character Recognition
PCA	Principle Components Analysis
PDF	Probability Density Function
QRNN	Quasi Recurrent Neural Network
RBF	Radial Based Function
RFE	Recursive Feature Elimination
RKHS	Reproducing Kernel Hilbert Space
RL	Reinforcement Learning

RNN	Recurrent Neural Network
RPN	Region Proposal Network
SGD	Stochastic Gradient Descent
SIFT	Scale-Invariant Feature Transform
SSD	Single Shot Multibox Detector
ST	Self-Attention Mechanism
STFT	Short-time Fourier transform
SVM	Support Vector Machine
TPU	Tensor Processing Unit
YOLO	You Only Look Once

نمادهای استفاده‌شده در کتاب

E	تابع هزینه
e	خطا
α	اسکالر تنظیم حساسیت و ضریب حد بالا در نورون های راف
β	اسکالر تنظیم حساسیت و ضریب حد پایین در نورون های راف
η	نرخ آموزش
k	گام زمانی گسسته
$\mathbf{x}$	بردار ورودی
o	خروجی
o_U	خروجی حد بالا
o_L	خروجی حد پایین
n_0	تعداد نورن‌های لایه ورودی
n_1	تعداد نورن‌های لایه اول
n_m	تعداد نورون های لایه m
r	تابع خطای الگوریتم عاطفی
t	زمان پیوسته
$\boldsymbol{\upsilon}$	بردار ویژه
$\mathbf{h}$	بردار پنهان
$\mathbf{W}^{\mathbf{s}}$	ماتریس وزن لایه s–ام
$\mathbf{w}^{\mathbf{s}}$	بردار وزن لایه s–ام
w_{ji}^{s}	وزن بین نورن j–ام و i–ام در لایه s–ام
y_d, y, d	خروجی مطلوب
$\hat{y}$	خروجی مدل
$\mathbf{W}^{L}$	ماتریس حد پایین وزن
$\mathbf{W}^{U}$	ماتریس حد بالای وزن
net	مجموع وزن‌دار نورن
net_U	مجموع حد بالای وزن‌دار نورون
net_L	مجموع حد پایین وزن‌دار نورون
$.^{h}$	اندیس بالانویس لایه پنهان
$.^{o}$	اندیس بالانویس لایه خروجی
$.^{+}$	شبه معکوس

شانزده

$\|\cdot\|_n$	نرم n–ام
$\|\cdot\|_2$	نرم اقلیدسی
$\|\cdot\|_\infty$	نرم بینهایت
$\|\cdot\|_F$	نرم فربینیوس

پیشگفتار

امروزه رشد روزافزون شاخه های مختلف فناوری و تاثیر آن در جامعه بشری مشهود است. هوش مصنوعی و یادگیری ژرف نیز از این روند مستثنی نبوده و از این رو به ویژه در دو دهه اخیر شاهد گسترش چشمگیر این شاخه از علوم کامپیوتر و به کارگیری آن در سایر علوم و فناوری ها هستیم. پیدایش مفاهیمی همانند بینائی ماشین، خودروهای خودران، دستیارهای هوشمند و... تنها گوشه ای از این کاربردهای یادگیری ژرف هستند. در همین راستا به جهت تسهیل پیاده سازی رویکردهای مبتنی بر هوش مصنوعی و یادگیری ژرف بسترهای نرم افزاری و سخت افزاری متنوعی نیز در سال های اخیر عرضه شده اند، به کارگیری چنین بسترهایی علیرغم تسریع توسعه و پیاده سازی مباحث این بخش در مواردی به جهت عدم داشتن دانش پیش زمینه ای مناسب سبب اتلاف استعدادهای موجود و نتایج نامطلوب شده است. از این رو در کتاب حاضر تلاش شده است تا با ارائه رویکردی تئوری و کاربردی حدالامکان پیش زمینه مناسبی برای علاقه مندان به حوزه یادگیری ژرف به صورت خودآموز فراهم شود. خواننده با مطالعه این کتاب علاوه بر تسلط بر مفاهیم مطرح در حوزه یادگیری ژرف با نوآوری ها و رویکردهای کاربردی معرفی شده نیز آشنا می شود، علاوه بر این مطالبی که عموما به عنوان پیش نیاز یادگیری ژرف مطرح هستند نیز در فصل های ابتدائی کتاب ارائه شده است تا علاقه مندان بدون نیاز به داشتن پیش زمینه مرتبط نیز بتوانند از محتویات کتاب بهره مند شوند.

در این میان آنچه که در توسعه زمینه های فناوری به آن کمتر توجه شده، نحوه تولید و ارائه مطالب مرتبط با این فناوری ها است. در سال های اخیر با افزایش گرایش به تمرکززدائی از سامانه های مختلف و ارائه بسترهای غیرمتمرکز و خودگردان در حوزه های مختلف همچون فناوری بلاک چین[1]، نحوه ایجاد، اعتبارسنجی، ارائه و انتشار مطالب علمی نیز دستخوش تغییر شده است. امروزه عموما محتوای عملی نه در قالب مطالب نظری گنجانده شده در کتب و مقالات دانشگاهی، بلکه به صورت مطالب حاصل از تجربیات عملی و نتایج عینی، اعتبارسنجی شده و در دسترس عموم قرار می گیرد. برهمین اساس این کتاب نیز به صورت رایگان متنشر شده تا به نوبه خود سهمی در این روند داشته باشد.

[1] Blockchain

امیدوارم که مطالب کتاب مطلوب شـما علاقه مندان و دوسـتداران مفاهیم حوزه یادگیری ژرف بوده و با ارسـال نظر اصـلاحی و تکمیلی خود به مؤلف، در بهبود ویرایش های بعدی این کتاب یاری نمائید.

سینا رنجبر کوه فرهادی
تابستان ۱۴۰۰

فصل ۱
مقدمه یادگیری ژرف

۱.۱ معرفی

ریشه تمایل به کاربرد دستمایه های پیرامون قبل از پیدایش تاریخ بشری در هوش غریزی همه گونه های جانداران گیتی مشهود است. از این رو نوع بشر نیز در تکامل این روند غریزی همواره در پویش برای بهره گیری از امکلنات محیط خود برای لبداع ابزارهایی برای خود بود، تلاش برای بهبود کارایی این مصنوعات در سراسر تاریخ ادامه داشته و با گسترش تمدن بشری همواره پیشرفت هایی در این ابداعات صورت گرفته است.

در مسیر این روند رو به تکامل همواره تلاش آدمی بر آن بوده است که به ابداعات خود قوه هوش را اضافه کند، نشانه هایی از این تمایلات به ساخت چنین ابزار هایی حتی در تاریخ اساطیری تمدن ها نیز وجود دارد. شاید بتوان اولین گام عملی انسان در این جهت را تربیت و به کارگیری حیوانات برای تسهیل بیشتر امور خود دانست. اختراع ماشین هایی همچون کامپیوترها، سامانه ها و دستگاه های خودکار صنعتی، خودرو های خودران و... معلول این پویش بشری در تمدن امروزی است.

بی تردید ابداع چنین ماشین هایی نیازمند دانش مدون و منظمی است که اساس کارکرد این ماشین ها بر آن استوار باشد. این علوم نیز همچون ابداعات در طول قرون متمادی پیشرفت هایی را به خود دیده است. پیدایش علومی چون هوش مصنوعی در عصر حاضر نشانگر این سیر تکاملی است. علومی بین رشته ای که موضوعات مختلفی از سایر علوم پایه، مهندسی و... را در خود دارد، در ادامه به معرفی مختصر هوش مصنوعی و موضوعات و مسائل مورد بحث در این رشته می پردازیم.

۱.۲ هوش مصنوعی[1]

علم هوش مصنوعی که امروزه در دسته بندی دانشگاهی، رشته ای از زیر شاخه های علوم کامپیوتر[2] محسوب می شود، شالوده ای از علوم ریاضی، علوم انسانی، علوم تجربی و زیستی

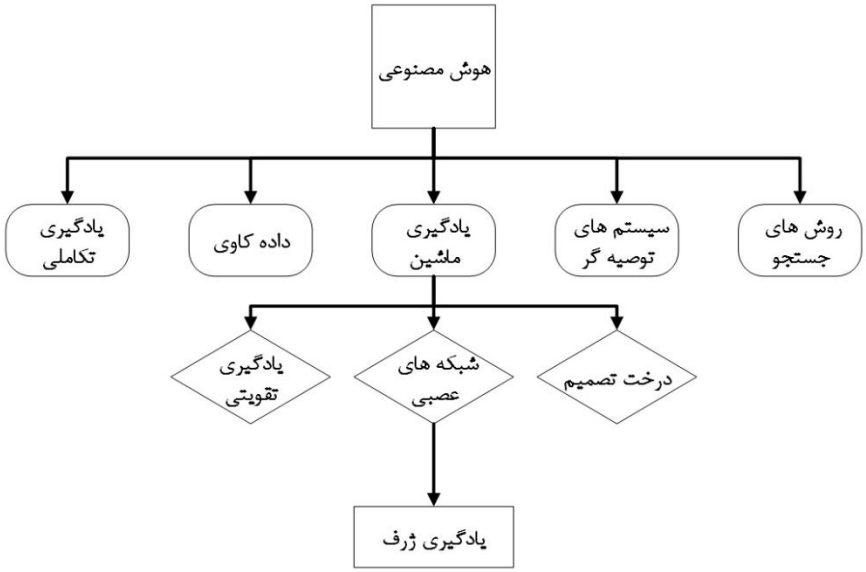
[1] Artificial intelligence (AI)
[2] Computer science

مهندسی کامپیوتر، مهندسی کنترل و... است که شامل مباحث گسترده ای چون یادگیری ماشین[۱]، داده کاوی[۲]، یادگیری تکاملی[۳]، سیستم های توصیه گر[۴] و... است.

گسترش و تکامل هوش مصنوعی در چند دهه اخیر معلول پیشرفت و کاربرد هر چه بیشتر مباحثی مانند دانش رمزنگاری[۵]، علوم شناختی[۶]، مهندسی پزشکی[۷]، تحلیل آماری[۸] و... است. معرفی و توسعه ماشین تورینگ[۹] در سال ۱۹۳۶ میلادی توسط آلن تورینگ[۱۰] و توسعه مدل های مبتنی بر زنجیره پنهان مارکف[۱۱] و فرایند مارکف[۱۲] که بر پایه نظریات آندری مارکف[۱۳] (۱۹۰۶م.) تعریف شده اند، نمونه هایی از گام های اولیه در راستای تکامل و گسترش هوش مصنوعی است.

جبر خطی و آمار و احتمالات مباحثی از علم ریاضیات هستند که در هوش مصنوعی به طور گسترده استفاده می شوند. یادگیری ماشین نیز یکی از مهم ترین موضوعات هوش مصنوعی است؛ شبکه های عصبی مصنوعی یکی از عناوین مطرح شده در هوش مصنوعی به حساب می آید که یادگیری ژرف بر اساس آن ارائه شده است. شکل ۱-۱ تصویری کلی از مهم ترین مباحث و بخش های مختلف هوش مصنوعی است که در ادامه به معرفی هر کدام می پردازیم:

[1] Machine learning
[2] Data mining
[3] Evolutionary learning
[4] Recommender systems
[5] Cryptography
[6] Cognitive science
[7] Biomedical engineering
[8] Statistical analysis
[9] Turing machine
[10] Alen Turing
[11] Hidden Markov chain
[12] Markov process
[13] Andrey Markov

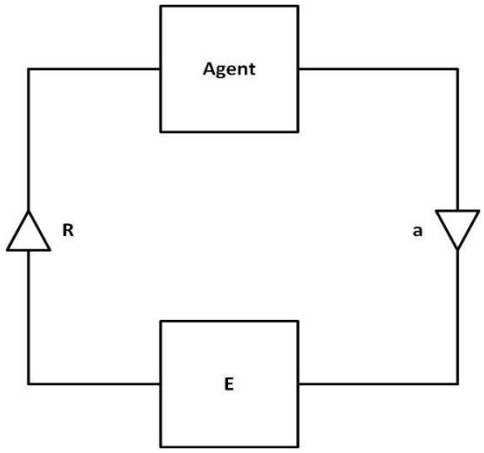


شکل ۱-۱: مباحث مختلف هوش مصنوعی

- **یادگیری تقویتی:** شاخه ای از هوش مصنوعی، یادگیری ماشین، است کاربرد گسترده ای در بهینه سازی، سیستم های چند عاملی، مهندسی کنترل و... دارد. در یادگیری تقویتی به عملکرد عامل، در محیط نسبت به حالت[1] فعلی، s، آن پاداش[2]، R، به عنوان بازخورد داده می شود. هدف در این سناریو بهبود عملکرد سیستم با روش سعی و خطا به منظور بیشینه کردن مجموع پاداش عامل است. پاداش برای هر عملکرد عامل به عنوان تابعی از حالت فعلی، s، عملکرد عامل، a، و حالتی که عامل با اجرای عمل از حالت فعلی، s، وارد حالت جدید، s'، می شود، تعریف می شود.

[1] State
[2] Reward

شکل ۲-۱ گراف سناریو یادگیری تقویتی را نشان می دهد که در آن عامل[1] در محیط[2]، E، بوده و در حالت s، با اجرای عمل a وارد حالت،s'، شده و پاداش،$R_{(s,a,s')}$، را دریافت می کند.



شکل ۲-۱: سناریو یادگیری تقویتی

- **سیستم های توصیه گر:** سیستم های توصیه گر از مباحث کاربردی در فروشگاه های اینترنتی، سیستم های پخش موسیقی و ویدئو و سایر مواردی است که در آن ها بهبود تجربه کاربری مد نظر است. هدف در این سیستم ها پیشنهاد اقلام مناسب برای هر کاربر با ارائه روش هایی برای تحلیل ترجیحات کاربر در انتخاب اقلام دلخواه و ویژگی های مختلف اقلام است. نبود اطلاعات کافی در مورد کاربر جدید که منجر به پیشنهاد اقلام نامناسب به کاربر می شود از چالش های اصلی در این نوع سیستم ها است، که به شروع سرد مرسوم است.

- **یادگیری ماشین:** از مباحث مطرح در هوش مصنوعی است که موضوعات مختلف آن به خصوص شبکه های عصبی پایه طرح یادگیری ژرف است که در ادامه به معرفی آنها به عنوان پیشنیاز برای یادگیری ژرف می پردازیم. ترکیب روش های مبتنی بر یادگیری تقویتی با شبکه های عصبی عمیق در سال های اخیر منجر به پیدایش مباحثی تحت عنوان یادگیری تقویتی عمیق[3] شده است.

- **داده کاوی:** با پیشرفت تکنولوژی حجم وسیعی از دادگان مختلف از سیستم های

[1] Agent
[2] Environment
[3] Deep Reinforcement Learning

حوزه های گوناگون به طور پیوسته ایجاد می شود. نیاز مبرم برای ذخیره بهینه این دادگان و استخراج دقیق و سریع اطلاعات مورد نیاز از این دادگان به ارائه روش های داده کاوی منجر شد. مباحثی چون تحلیل کلان داده ها در ادامه بر اساس این روش ها توسعه داده شده اند.

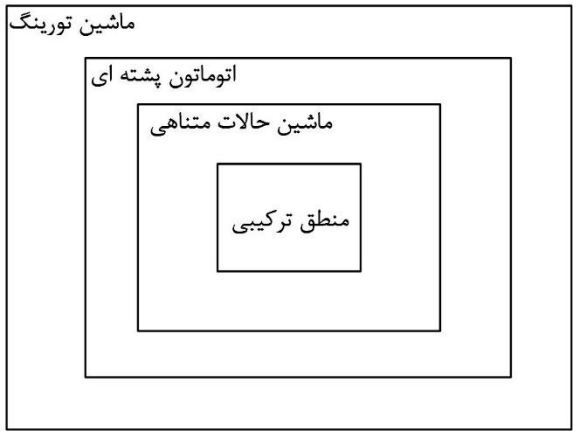
- **یادگیری تکاملی**: در یادگیری تکاملی که زیر شاخه مبحث گسترده‌تر پردازش تکاملی است، به مطالعه روش های مبتنی بر جمعیت برای ارائه راهکارهای بهینه برای مسائل گوناگون پرداخته می شود. الگوریتم ژنتیک[1] از مهم ترین روش های مطرح در یادگیری تکاملی است که در آن مقادیر و وضعیت پارامترهای موثر در مسئله، با الهام از علوم ژنتیک طبیعی به صورت کروموزوم[2] های جداگانه در نظر گرفته شده و پاسخ هر کدام تحلیل می شود. هدف الگوریتم یافتن مقادیر بهینه‌ترین کروموزم است. الگوریتم های مدل شده از رفتار جمعی گونه های جانوری، هوش جمعی و... نیز از سایر موضوعات مطرح در این بخش است.

۱.۳ نظریه آتوماتا[3]

نظریه آتوماتا که به نام نظریه ماشین ها نیز شناخته می شود، یکی از مهم ترین نظریه های مطرح در علوم کامپیوتر است که نقش مهمی در هوش مصنوعی داشته و به مطالعه ماشین های انتزاعی می پردازد، آتوماتا در لغت ریشه یونانی داشته و به معنی خودساز است.

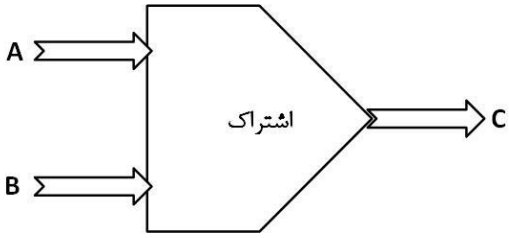
در این نظریه حالت های مختلف یک ماشین انتزاعی به ازای ورودی های مختلف بررسی می شود. منطق ترکیبی[4] از مفاهیم اساسی مطرح در این نظریه است که ماشین های حالت محدود، و پس ار آن ماشین های خودکار برگشتی بر اساس آن توسعه داده شده اند. ماشین تورینگ نیز در واقع یک ماشین انتزاعی که بر خلاف سایر ماشین های مطرح در این نظریه برای حالت های بینهایت نیز تعریف شده و شکل جامع تری از مباحث مطرح در این نظریه است. شکل ۳-۱ تصویر کلی روابط بین موضوعات نظریه آتوماتا را نشان می دهد، که در ادامه به بررسی آن ها می پردازیم:

[1] Genetic algorithm
[2] Chromosome
[3] Automata theory
[4] Combinational logic



شکل ۳-۱: موضوعات نظریه آتوماتا

- **منطق ترکیبی:** در طراحی سیستم های کامپیوتری مدار های ترکیبی[1] و ترتیبی[2]، که دارای ورودی و خروجی های دو وضعیتی، صفر و یک، هستند، کاربرد گسترده-ای دارند. مدارهای ترکیبی مدارهایی هستند که در آن ها وضعیت خروجی مدار وابسته فقط به وضعیت فعلی ورودی ها وابسته است در حالی که در مدارهای ترتیبی علاوه بر وضعیت فعلی، وضعیت قبل ورودی ها نیز در خروجی مدار موثر است. منطق حاکم بر مدار های ترکیبی که شامل اجتماع، اشتراک و سایر ترکیب های ممکن برای ورودی ها که بر اساس آنها وضعیت خروجی تعیین می شود، منطق ترکیبی است. تصویر ۴-۱ یک مدار ترکیبی اشتراک شامل دو ورودی A و B و خروجی C است.

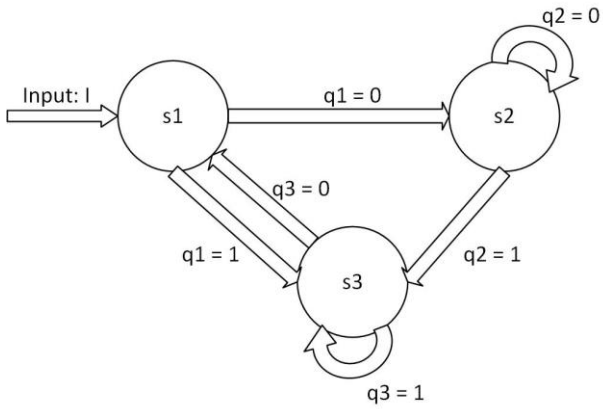


A	B	C
0	1	0
1	0	0
0	0	0
1	1	1

شکل ۴-۱ : مدار ترکیبی اشتراک

[1] Combinational
[2] Sequential

- **ماشین حالات متناهی**[1]: ماشین حالات متناهی یک مدل ریاضی است که دارای تعداد مشخصی حالت بوده و نسبت به وضعیت ورودی و حالت فعلی به حالتجدید منتقل می شود. برای انتقال از حالت فعلی به حالت جدید در این ماشین‌ها از منطق ترکیبی برای تحلیل وضعیت ورودی و حالات استفاده می‌شود. شکل ۵-۱ گراف یک ماشین حالت متناهی با حالت های s_1 s_2 s_3 و ورودی، I، است که در صورت تحقق هر وضعیت مشخص شده از منطق های q_i , $i = 1,2,3$، وضعیت ماشین تغییر می‌کند.



شکل ۵-۱ : گراف ماشین حالات متناهی

- **اتوماتون پشـــته‌ای**[2]: یکی دیگر از ماشـــین های انتزاعی مطرح در نظریه آتوماتا، اتوماتون پشـته‌ای اسـت. این ماشـین‌ها منطق کارکردی مشـابه با ماشـین‌های حالات متناهی دارند، با این تفاوت که بر خلاف ماشـــین حالات متناهی دارای پشـته‌ای از حالات و نوار ورودی اسـت. اتوماتون پشـته‌ای نوار ورودی و وضـعیت فعلی را تحلیل کرده و وضعیت جدید را بر روی پشته‌ی حالات مشخص می‌کند. به عبارت دیگر اتوماتون پشته‌ای تعمیم یافته ماشین حالات متناهی است.

- **ماشین تورینگ:** بر خلاف سایر مدل های مطرح شده در آتوماتا، ماشین تورینگ امکان تحلیل نوار. «یک حافظه نامحدود به صـورت یک نوار قطعه‌بندی شـده که روی هر قطعه نمادی وجود دارد و در هر لحظه فقط یکی از قطعات در ماشـــین قرار دارد که به آن نماد اسـکن شـده گفته می شـود. بر اسـاس نماد اسـکن شـده ماشـــین می‌تواند نماد بعدی روی نوار را تغییر دهد ولی اصـول کارکرد ماشـین تغییری نمی‌کند. همچنین امکان دارد نوار به جلو یا عقب حرکت کند، بدین

[1] Finite-state machine
[2] Pushdown automaton

ترتیب امکان دارد هر کدام از نمادهای روی نوار به عنوان نماد اسکن شده شناخته شود.» این تعریفی است که آلن تورینگ برای مدل خود ارائه کرده است. ماشین تورینگ اساس کارکرد واحد پردازنده مرکزی در رایانه‌های امروزی است.

۱.۴ نظریه بازی[1]

نظریه بازی کاربرد گسترده‌ای در حوزه‌های مختلف علوم انسانی، مهندسی کنترل، علوم کامپیوتر و... دارد. ایده اولیه نظریه از تحلیل روابط و قوانین حاکم در بازی‌های دو نفره الهام گرفته شده است. چارلز والدگراو[2] (۱۷۱۳ م.) یک مدل کمین بیش برای یک بازی کارتی دو نفره فرانسوی به نام Le Her ارائه کرد و در ادامه مباحث این نظریه برای شبیه سازی مسائل گوناگون مطرح در حوزه هایی مانند اقتصاد، زیست شناسی و... به طور گسترده توسط افرادی با تخصص‌های مختلف توسعه داده شده‌اند. هدف نظریه بازی توسعه مدل‌های ریاضی برای بهینه سازی عملکرد چند عامل است که با هم فعالیت همکارانه دارند.

۱.۵ یادگیری ماشین

برای یادگیری ماشین تا کنون تعاریف مختلفی ارائه شده است، از نظر آرتور ساموئل[3] (۱۹۵۹ م.) «یادگیری ماشین دانشی است که به رایانه قابلیت یادگیری بدون برنامه ریزی صریح را می دهد.» تام میچل[4] (۱۹۹۸ م.) یادگیری ماشین را اینگونه تعریف می کند: «برنامه کامپیوتری که از تجربه E حاصل از وظیفه T و تعدادی معیار عملکردی P آموزش دیده و عملکرد آن برای وظیفه T اگر با معیار های P ارزیابی شود باعث بهبود تجربه T می شود.»

فرض کنید برای یک رایانه یک برنامه براساس منطق های مختلف ارائه شود، که در صورت تحقق هر وضعیت از منطق ها رایانه خروجی تعریف شده در ازای آن را تولید می کند. حال اگر تعداد منطق های تعریف شده بیشتر شود در این صورت علاوه بر نیاز به منابع محاسباتی قدرتمندتر در رایلنه برای اجرای بهینه برنامه، تعریف کردن منطق ها به رایلنه با ترتیب مناسب نیز دشوار خواهد بود؛ در مواردی ممکن است حتی برای حالاتی که امکان تحقق آن در سامانه وجود دارد منطق مناسبی تعریف نشده و باعث بروز خطا شود.

در حالت کلی هدف یادگیری ماشین حل این معضل با ارائه مدلی است که قابلیت یادگیری بر اساس مجموعه ای از دادگان شامل حالت های ممکن و خروجی معادل هر حالت را داشته باشد و به ازای هر پیشامد ممکن در بازه عملکردی، حتی حللت هایی که مقادیر آن در مجموعه دادگان

[1] Game theory
[2] Charles Waldegrave
[3] Arthur Samuel
[4] Tom Mitchell

آموزشی نباشد، خروجی مناسبی داشته باشد.

طبقه بندی[1]، رگرسیون[2] و خوشه بندی[3] عمده مسائل مطرح در یادگیری ماشین است. مسائل و مدل های مطرح در یادگیری ماشین را می توان در گروه های مختلف دسته بندی کرد که یکی از آنها مدل هایی با یادگیری با نظارت، با سرپرست، و مدل هایی با یادگیری بی نظارت، بدون سرپرست، است. در فصل های بعدی به بررسی مسائل یادگیری ماشین سایر دسته بندی های ارائه شده برای مدل های آن می پردازیم.

۱.۶ نگاه کلی کتاب

این کتاب شامل سه بخش است. بخش اول، از ابتدا تا فصل ششم، مقدمه شامل، جبر خطی، احتمالات و ریاضیات آماری، داده کاوی، پردازش سیگنال و یادگیری ماشین که در آن به بررسی پیشنیاز های لازم برای مبحث یادگیری ژرف پرداخته شده است. در بخش دوم، فصل هفتم تا دوازدهم، به بررسی مدل های مختلف در یادگیری ژرف از جمله، شبکه های عصبی، مدل های یادگیری بازنمائی، شبکه های پیچشی، شبکه های بازگشتی و حافظه دار و... می پردازیم. در نهایت در بخش سوم، فصل سیزدهم تا انتها، روش های کاربرد های مختلف و چالش های مطرح آن ها پرداخته و رویکردهای مختلف ارائه شده برای رفع هر کدام از آن ها را بررسی می کنیم.

از این کتاب می توان به عنوان یک مرجع درسی برای یادگیری ژرف در یک نیمسال تحصیلی استفاده کرد، همچنین برای علاقه مندان به مبحث یادگیری ژرف که پیش زمینه ای از مباحث هوش مصنوعی و یادگیری ماشین ندارند نیز قابل استفاده بوده و به صورت خودآموز اطلاعات جامعی را به آنان ارائه می کند.

مسائل و سوالاتی نیز در انتهای هر فصل هم به صورت سوالات تحلیلی و تشریحی و هم به صورت مسائل عملی برای موضوعات مطرح در دنیای واقعی ارائه شده که چالش هایی را برای خواننده پس از مطالعه هر فصل ایجاد کرده و باعث تعمیق بیشتر مطالب در ذهن خواننده می شود.

[1] Classification
[2] Regression
[3] Clustering

فصل ۲
جبر خطی[1]

۲.۱ مقدمه

هدف از طرح مباحث جبرخطی حل مسائل مربوط به توابع خطی و برداری است. در این فصل ابتدا به تعاریف اولیه مطرح در جبر خطی پرداخته و پس از آن روابط و مباحث مربوط به این تعاریف را که در مطالب یادگیری ژرف مورد استفاده قرار می گیرند را بررسی می کنیم.

۲.۲ تعاریف اولیه

اسکالر[2]: اسکالرها، متغیرها یا اعدادی هستند که به صورت منفرد در روابط ظاهر می شوند. می توان چنین برداشت کرد که اسکالر یک **تانسور[3]** صفر بعدی است. در ادامه به تعریف تانسور می پردازیم. در این کتاب در حالت کلی اسکالرها با حروف کوچک نمایش داده می شوند. برای مثال، $s \in \mathbb{N}$ نماد اسکالر طبیعی s است.

بردار[4]: بردارها از آرایه متوالی اعداد، اسکالر، تشکیل می‌شوند. اندازه، طول، یک بردار برابر با تعداد اعداد آن بردار است. ترتیب قرارگیری اعداد که به درایه مرسوم است، در بردار با یک عدد زیروند[5] مشخص می‌شود؛ در صورت استفاده از زیروندهایی با مقادیر مثبت ترتیب درایه ها از ابتدا به انتهای بردار و در صورت استفاده از زیروند منفی ترتیب برعکس حالت مثبت، از انتها به ابتدا خواهد بود. در صورت تغییر ترتیب توالی و جا به جایی درایه های یک بردار با هم، بردار حاصل با بردار اولیه برابر نخواهد بود. در صورتی که همه درایه های عضو یک بردار با اندازه n زیر مجموعه اعداد حقیقی باشد، $\forall\, x_n \in \mathbb{R}$، در این صورت بردار متعلق به یک دستگاه کارتزین[6]، فضای دکارتی، n-بعدی، $\mathbb{R}^n$، است. در این صورت نقطه ابتدایی بردار مفروض در فضای دکارتی در مبدا و انتهای آن در نقطه‌ای برابر با مقادیر درایه های بردار قرار دارد. از

---

[1] Linear algebra
[2] Scalar
[3] Tensor
[4] Vector
[5] Subscript
[6] Cartesian coordinate system

نمایش برداری می‌توان برای بیان مختصات یک نقطه استفاده کرد. بردار یک تانسور یک بعدی است. در حالت کلی در این کتاب بردارها با حروف کوچک برجسته نمایش داده می‌شوند. رابطه ۲-۱ نشان‌دهنده بردار $\boldsymbol{v}$ به اندازه n و درایه های $v_1, v_2, \dots, v_n$ است.

$$\boldsymbol{v} = \begin{bmatrix} v_1 \\ v_2 \\ \vdots \\ v_n \end{bmatrix} \qquad (۲-۱)$$

ماتریس[1]**:** ماتریس‌ها نمایش دیگری برای مجموعه آرایه اعداد هستند. ماتریس‌ها تشابهات بسیاری با بردارها دارند با این تفاوت که برخلاف بردار، درایه‌های یک ماتریس در دو جهت سطری و ستونی تعریف می‌شود؛ به بیان دیگر بردار یک ماتریس با یک ستون است. ترتیب درایه‌های ماتریس با زیروندهای دوگانه مشخص می‌شود. ماتریس‌ها تانسور های دو بعدی محسوب می‌شود. رابطه ۲-۲ نمایش یک ماتریس $n \times m$، n : تعداد سطرها - m : تعداد ستون‌ها، است. در حالت کلی در این کتاب ماتریس‌ها با حروف بزرگ نمایش داده می‌شوند.

$$D = \begin{bmatrix} d_{1,1} & d_{1,2} & \cdots & d_{1,m} \\ d_{2.1} & d_{2,2} & \cdots & \vdots \\ \vdots & \vdots & \ddots & \vdots \\ d_{n,1} & d_{n,2} & \cdots & d_{n,m} \end{bmatrix}_{n \times m} \qquad (۲-۲)$$

تانسور: می‌توان تانسور را تعمیم یافته ماتریس فرض کنیم؛ برخلاف ماتریس‌ها که همواره دو بعدی، سطر و ستون، هستند، یک تانسور می‌تواند هر بعدی داشته باشد. زیروند مشخص کننده ترتیب قرار گیری هر درایه در تانسور n-بعدی نیز مشابه ماتریس ها با استفاده از زیروندهای n-گانه نمایش داده می شود. تعاریف پیشتر مطرح شده برای اسکالر، بردار و ماتریس هر کدام حالتی خاص از تانسور است. در بخش‌های بعدی به بررسی عملیات مختلفی که بر روی تانسورها اجرا می‌شود می‌پردازیم. در حالت کلی در این کتاب تانسورها با حروف بزرگ برجسته نمایش داده می شوند.

[1] Matrix

۲.۳ عملیات اصلی در تانسورها و ماتریس ها

در این قسمت به بررسی عملیات های تعریف شده برای تانسورها و ماتریس ها می پردازیم. برای سهولت در بیان مفاهیم همه روابط را برای ماتریس‌ها طرح می کنیم در حالی که این روابط قابل تعمیم به ابعاد بالاتر، تانسور، نیز هستند.

۲.۳.۱ جمع و تفریق ماتریس با اسکالر

جمع یا تفریق یک اسکالر با یک ماتریس به همه درایه‌های ماتریس اعمال می‌شود. رابطه ۲-۳ جمع اسکالر s با ماتریس $A_{n\times m}$ را نشان می‌دهد. به طور مشابه برای یک تانسور با ابعاد بالاتر نیز اسکالر با همه درایه های تانسور جمع یا تفریق می شود.

$$s + A_{n\times m} = \begin{bmatrix} a_{1,1} + s & \cdots & a_{1,m} + s \\ \vdots & \ddots & \vdots \\ a_{n,1} + s & \dots & a_{n,m} + s \end{bmatrix}_{n\times m} \quad (۲-۳)$$

۲.۳.۲ جمع و تفریق ماتریس با ماتریس

شرط لازم برای انجام عملیات جمع یا تفریق دو ماتریس، دو تانسور، برابر بودن ابعاد، $n \times m$، آنهاست. در صورت برقراری شرط درایه‌های نظیر به نظیر با هم جمع یا تفریق شده و ابعاد ماتریس حاصل برابر با دو ماتریس ابتدایی است. درایه‌های ماتریس C حاصل از جمع دو ماتریس A و B از رابطه ۲-۴ به دست می‌آید.

$$c_{i,j} = \sum_i^n \sum_j^m a_{i,j} + b_{i,j} \quad (۲-۴)$$

۲.۳.۳ ترانهاده[1] ماتریس

ترانهاده ماتریس A با نماد A^T نشان داده می‌شود، که درایه‌های آن از رابطه ۲-۵ به دست می‌آید. به عبارت دیگر با تبدیل سطرهای ماتریس به ستون ترانهاده آن ایجاد می‌شود. ابعاد ترانهاده یک ماتریس با ابعاد $n \times m$ برابر با $m \times n$ است. برای یک تانسور با ابعاد بالاتر از ۲، ترانهاده رابطه ۲-۵ بین دو بعد انتخابی تعریف می شود، بنابراین برای یک تانسور n-بعدی می توان $\binom{n}{2}$ ترانهاده مختلف تعریف کرد.

$$A_{i,j}^T = A_{j,i} \quad (۲-۵)$$

[1] Transpose

۲.۳.۴ ضرب اسکالر در ماتریس

ضرب اسکالر در ماتریس مشابه با حالت جمع و تفریق بوده و به همه درایه‌های ماتریس، تانسور، اعمال می‌شود. رابطه ۶-۲ نمایش ضرب اسکالر s در ماتریس $A_{n\times m}$ است.

$$s.A_{n\times m} = \begin{bmatrix} a_{1,1}.s & \cdots & a_{1,m}.s \\ \vdots & \ddots & \vdots \\ a_{n,1}.s & \dots & a_{n,m}.s \end{bmatrix}_{n\times m} \quad (۲-۶)$$

۲.۳.۵ ضرب بردار در ماتریس

برای ضرب یک بردار در ماتریس، اندازه بردار حداقل با یکی از ابعاد ماتریس، تانسور، باید برابر باشد. در این صورت هر یک از درایه های بردار در راستای ابعاد دیگر که با اندازه بردار برابر نیستند مطابق رابطه ۷-۲ در همه درایه های ماتریس، تانسور، ضرب می شود. رابطه ۷-۲ نمایش ضرب بردار $\boldsymbol{v}$ با اندازه n در ماتریس $A_{n\times m}$ است. برای حالتی که اندازه بردار با هر دو طول ماتریس برابر باشد، باید راستای ضرب تعریف شود. روند مشابهی را می توان برای ضرب ماتریس در تانسور تعریف کرد، در این صورت ابعاد ماتریس باید با حداقل دو بعد از تانسور برابر بوده و با انجام عملیات درایه های ماتریس در طول ابعاد دیگر تانسور که برابر با ابعاد ماتریس نیستند ضرب شود، ضرب درایه به درایه، هادامارد، که در ادامه معرفی می کنیم حالت خاصی از این ضرب است. این روند را برای ضرب تانسور در تانسور نیز می توانیم تعمیم دهیم.

$$\boldsymbol{v}.A_{n\times m} = \begin{bmatrix} a_{1,1}.v_1 & \cdots & a_{1,m}.v_1 \\ a_{2,1}.v_2 & \cdots & a_{2,m}.v_2 \\ \vdots & \ddots & \vdots \\ a_{n,1}.v_n & \dots & a_{n,m}.v_n \end{bmatrix}_{n\times m} \quad (۲-۷)$$

۲.۳.۶ ضرب ماتریس در ماتریس

۲.۳.۶.۱ ضرب ماتریسی

حاصل ضرب ماتریسی، ماتریس $A_{n\times m}$ در ماتریس $B_{p,q}$، ماتریس C با ابعاد $n \times q$، تعداد سطر برابر با تعداد سطر ماتریس اول و تعداد ستون برابر با تعداد ستون ماتریس دوم، است. ضرب دو ماتریس با نماد × نشان داده می شود، $C = A \times B$. شرط لازم برای برای انجام این ضرب، $m = q$، برابری تعداد ستون ماتریس اول با تعداد سطر ماتریس دوم، است. درایه‌های ماتریس حاصل از رابطه ۸-۲ محاسبه می‌شود. با توجه به رابطه ۸-۲ ابتدا درایه‌های سطر اول ماتریس اول نظیر به نظیر با درایه‌های ستون اول، ماتریس دوم ضرب عددی شده و حاصل

ضرب ها با هم جمع و درایه 1,1 ماتریس حاصل ضرب محاسبه می‌شود، این روند ادامه پیدا کرده و سایر درایه های ماتریس C محاسبه می شوند.

$$C_{i.j} = \sum_k A_{i.k} B_{k,j} \quad (۲-۸)$$

۲.۳.٦.۲ ضرب درایه به درایه (ضرب هادامارد[1])

در ضرب درایه به درایه، درایه‌های ماتریس حاصل، C، از ضرب آرایه های نظیر به نظیر اعداد درایه‌های ماتریس های اولیه تشکیل شده است. ضرب عنصر به عنصر دو ماتریس A و B با نماد $A \odot B$ نشان داده می‌شود. فرمول محاسبه عناصر ماتریس حاصل رابطه ۹-۲ است. شرط لازم انجام عملیات برابری ابعاد دو ماتریس اولیه بوده، که حاصل آن نیز ماتریسی با همان ابعاد است.

$$c_{i.j} = a_{i.j}. b_{i.j} \quad (۲-۹)$$

۲.۳.۶.۳ تقسیم ماتریس ها

عملیات تقسیم برای ماتریس‌ها تعریف نمی‌شود، اما می‌توانیم یک ماتریس را در معکوس ماتریس دیگر ضرب کنیم. در ادامه به نحوه محاسبه معکوس یک ماتریس می‌پردازیم.

۲.۴ فضاهای برداری

۲.۴.۱ تعریف فضای برداری حقیقی

یک فضای برداری در واقع یک فرمول‌بندی برای فضای n-بعدی حقیقی اقلیدسی، $\mathbb{R}^n$، است به طوری که قوانین ضرب اسکالر در بردار و جمع دو بردار در آن فضای n-بعدی صدق کرده و نسبت به این دو عمل بسته این فضا است، بردار حاصل از عملیات نیز عضو همان فضا است. فضاهای برداری زیرمجموعه‌ای از میدان[2]ها هستند، یک میدان مجموعه از اسکالرها است که نسبت به عملیات ضرب و جمع بسته بوده و برای هر دو این عملیات دارای خواص معکوس پذیری، شرکت پذیری و... است. به بیان دیگر می‌توان یک فضای برداری حقیقی n-بعدی را به به شکل یک ماتریس با تعداد ستون n در نظر گرفت که سطرهای آن مقادیر ارائه شده در آن فضا (مجموعه نقاط، مجموعه بردار و...) است. از این نمایش ماتریسی فضای برداری در بخش‌های مختلف یادگیری ماشین و یادگیری عمیق استفاده می‌شود که به آن خواهیم پرداخت.

[1] Hadamard product
[2] Field

۲.٤.۲ زیرفضا[1]ها

یک زیرفضا زیرمجموعه‌ای از یک فضای برداری است، اگر دو شرط زیر را داشته باشد:

- جمع هر دو بردار عضو این زیرمجموعه عضو خودش باشد، بسته بودن نسبت به جمع.
- ضرب هر اسکالر در بردار عضو زیرمجموعه نیز عضو همان زیر مجموعه باشد، بسته بودن نسبت به ضرب.

به بیان ساده، یک زیرفضا، خود یک فضای برداری است به طوری که قوانین جمع دو بردار و ضرب اسکالر در بردار در این فضای زیرمجموعه نیز صادق بوده و نقطه مبدا آن با نقطه مبدا فضای برداری اصلی مشترک باشد.

برای مثال یک فضای برداری حقیقی سه‌بعدی، $\mathbb{R}^3$، را در نظر بگیرید، این فضای برداری فقط دارای زیرفضاهای ذیل است و سایر زیرمجموعه‌های تعریف شده برای آن زیرفضا نیستند:

- کل فضای ۳-بعدی[2]
- ابر صفحه گذرنده از مبدا مختصات فضای ۳-بعدی
- خط گذرنده از مبدا مختصات فضای ۳-بعدی
- نقطه مبدا مختصات فضای ۳-بعدی

۲.٤.۳ افراز[3] فضا و وابستگی خطی

۲.۴.۳.۱ ترکیب خطی

ترکیب خطی یک مجموعه‌ای شامل n بردار $\{\boldsymbol{v}_{1,\dots,}\boldsymbol{v}_n\}$ به صورت رابطه ۲-۱۰ تعریف می‌شود که در آن $s_{1,\dots,}s_n$ اسکالرهای حقیقی هستند.

$$s_1\boldsymbol{v}_1 + \cdots + s_n\boldsymbol{v}_n = \sum_{i=1}^{n} s_i\boldsymbol{v}_i \qquad (۲\text{-}۱۰)$$

۲.۴.۳.۲ استقلال خطی

بردارهای مجموعه $\{\boldsymbol{v_1}, \dots, \boldsymbol{v}_n\}$ از هم **مستقل خطی** هستند اگر و فقط اگر شرط رابطه‌ی ۲-۱۱ برای آن‌ها تنها در صورتی برقرار باشد که $s_1 = \cdots = s_n = 0$ که در این رابطه $s_{1,\dots,}s_n$ اسکالر های حقیقی هستند. در غیر این صورت بردارهای این مجموعه نسبت به هم **وابسته خطی** هستند.

---

[1] Sub-space

[2] هر مجموعه یک زیر مجموعه از خودش است.

[3] Span

$$s_1\boldsymbol{v}_1 + \cdots + s_n\boldsymbol{v}_n = 0 \qquad (۲-۱۱)$$

۲.۴.۳.۳ افراز

افراز شامل همه‌ی مجموعه نقاطی است که می‌توان با ترکیب خطی یک مجموعه بردار نشان داد. بنابراین تعریف سه بردار متعامد که از هم مستقل خطی هستند می‌توانند کل یک فضای برداری ۳-بعدی حقیقی، $\mathbb{R}^3$، را اسپن کنند.

۲.۵ دترمینان[1]

دترمینان تعریفی برای ماتریس است که فقط برای ماتریس های مربعی، ماتریسی که تعداد سطر و ستون آن برابر باشد، قابل ارائه است. با استفاده از دترمینان یک ماتریس مربعی می‌توان تکین[2] بودن آن ماتریس را تشخیص داد. دترمینان یک ماتریس مربعی همواره یک اسکالر است. براساس تعریف دترمینان، یک ماتریس غیرتکین[3] است اگر و فقط اگر دترمینیان آن صفر نباشد. فقط در صورتی که ماتریس غیرتکین باشد، معکوس برای آن ماتریس تعریف می‌شود. دترمینان ماتریس مربعی $A_{n\times n}$ به صورت رابطه ۱۲-۲ تعریف می‌شود. طبق رابطه ۱۲-۲ دترمینان مجموع همه‌ی جایگشت‌های ممکن برای ضرب درایه‌های ماتریس درهم با علامت جایگشت است. دترمینان ماتریس A با نماد $\det(A)$ یا $|A|$ نشان داده می‌شود.

$$|A| = \det(A) = \sum_{\pi} sign(\pi) \prod_{i=1}^{n} a_{i,\pi_i} \qquad (۲-۱۲)$$

در صورتی که π فرد باشد $sign(\pi) = 1$ و اگر π زوج باشد $sign(\pi) = -1$ است. برای توضییح زوج یا فرد بودن π مثالی را در نظر بگیرید که شامل لیست (ترتیب قرارگیری اعداد در نظر گرفته می شود) اعداد ۱ تا ۵ است $\{1,2,3,4,5\}$، اگر جایگشت $\{2,1,3,4,5\}$ از لیست اولیه را در نظر بگیریم این جایگشت با یک بار جایگزین شدن یک جفت از اعضا ایجاد شده و یک جایگشت فرد است پس در نتیجه علامت آن مثبت خواهد بود. حال جایگشت $\{2,1,4,3,5\}$ را در نظر بگیرید که درآن یک بار ۱و۲ جایگزین شده و یک بار ۳و۴ در نتیجه جایگشت زوج بوده و علامت آن منفی است.

طبق رابطه ۱۲-۲ دترمینان ماتریس مربعی 2×2 از رابطه ۱۳-۲ محاسبه می‌شود.

$$|A| = \det\left(\begin{bmatrix} a & b \\ c & d \end{bmatrix}\right) = ad - bc \qquad (۲-۱۳)$$

[1] Determinant
[2] Singular
[3] Non singular

با توجه به پیچیدگی رابطه ۱۲-۲ برای محاسبه دترمینان روش-هایی برای محاسبه دترمینان ماتریس های مربعی با ابعاد کوچک معرفی شده‌است. که بحث در مورد آن‌ها در چارچوب این کتاب نیست.

۲.٦ معکوس[1] ماتریس

۲.٦.۱ محاسبه معکوس ماتریس

معکوس یک ماتریس A که با نماد A^{-1} نشان داده می‌شود. تعریف معکوس یک ماتریس امکان حل معادلات خطی که به صورت ضرب ماتریس متغیرها و ضرایب تعریف شده‌اند را فراهم می کند. معکوس یک ماتریس بدین صورت تعریف می‌شود؛ ماتریسی که در صورت ضرب آن در ماتریس اصلی و یا بالعکس حاصل یک ماتریس همانی شود. رابطه ۱۴-۲ بیان این تعریف است.

$$A^{-1}A = I_n \qquad (۲-۱۴)$$

۲.٦.۲ ماتریس همانی[2]

ماتریس همانی یک ماتریس مربعی است که همه درایه‌های آن به جز درایه‌های قطر اصلی صفر بوده و درایه‌های روی قطر اصلی آن یک باشد. ماتریس همانی با نماد I_n نشان داده می‌شود که زیروند n نشانگر تعداد سطر یا ستون ماتریس است. در صورت ضرب یک ماتریس همانی با بعد مناسب در یک ماتریس A با ابعاد دلخواه پاسخ حاصل ضرب همواره برابر ماتریس A است.

۲.٦.۳ شبه معکوس[3] ماتریس

همان طور که اشاره کردیم معکوس ماتریس فقط برای ماتریس های مربعی غیرتکین تعریف می شود، در صورتی که یک ماتریس مربعی نباشد یا مربعی تکین باشد، دترمینان آن برابر با صفر باشد، برای آن معکوس ماتریس تعریف نمی شود. در این موارد می توانیم شبه معکوس تعریف کنیم. بدین ترتیب زمانی که معکوس ماتریس قابل تعریف نباشد می توانیم از شبه معکوس به عنوان جایگزین در روابط استفاده کنیم. معکوس و شبه معکوس یک ماتریس مربعی غیرتکین با هم برابر است به بیانی دیگر رابطه معکوس یک ماتریس حالتی خاص از رابطه شبه معکوس ماتریس است. شبه معکوس ماتریس A با توجه به معکوس پذیربودن

[1] Inverse of matrix
[2] Identity matrix
[3] Pseudo inverse

$A \times A^T$ و یا $A^T \times A$ به دو صورت شبه معکوس راست و شبه معکوس چپ به ترتیب مطابق روابط ۱۵-۲ و ۱۶-۲ تعریف شده و با نماد A^+ نمایش داده می شود.

$$A^+ = A^T(A \times A^T)^{-1} \quad (۲-۱۵)$$
$$A^+ = (A^T \times A)^{-1}A \quad (۲-۱۶)$$

۲.۷ مقادیر ویژه[1] و بردارهای ویژه[2]

۲.۷.۱ تجزیه ویژه

در بعضی مسائل تجزیه یک مسئله به المان‌های سازنده کمک بسزایی به حل آن می‌کند. یکی از روش‌های تجزیه ماتریس‌های مربعی **تجزیه ویژه** است. در این روش هدف یافتن مقادیر ویژه و بردار ویژه ماتریس است.
از دیدگاه هندسی هر ماتریس یک نگاشت خطی در فضای اقلیدسی است که در آن بردارهای ویژه جهت‌های این نگاشت و مقادیر ویژه مقیاس نگاشت در راستای هر جهت است.

۲.۷.۲ بردار ویژه و مقدار ویژه

بردار ویژه برای یک ماتریس مربعی بدین صورت تعریف می‌شود؛ برداری مخالف صفر ($\boldsymbol{v} \neq 0$) که در صورت ضرب ماتریس مربعی A در آن حاصل برابر با ضرب مقدار ویژه که یک اسکالر حقیقی غیر صفر است، می‌شود. رابطه ۱۷-۲ نمایش این تعریف است.

$$A\boldsymbol{v} = \lambda\boldsymbol{v} \quad (۲-۱۷)$$
$$\lambda \neq 0, \lambda \in \mathbb{R}$$

برای حالتی که بردار ویژه در ماتریس مربعی ضرب می‌شود رابطه ۱۸-۲ برقرار است که در این حالت به آن بردار ویژه چپ گفته می‌شود.

$$\boldsymbol{v}^T A = \boldsymbol{v}^T \lambda \quad (۲-۱۸)$$

اگر فرض کنیم که $\boldsymbol{v}$ یک بردار ویژه برای یک ماتریس مربعی A باشد در این حالت با ضرب اسکالر حقیقی مخالف صفر در $\boldsymbol{v}$، $s\boldsymbol{v}, s \neq 0, s \in \mathbb{R}$، مقدار ویژه، λ، برای بردار حاصل ضرب تغییر نمی‌کند. در این صورت بردار ویژه حاصل، $s\boldsymbol{v}$، به بردار ویژه اصلی، $\boldsymbol{v}$، وابسته خطی است،

[1] Eigen values
[2] Eigen vectors

ولی یک ماتریس مربعی می‌تواند بردارهای ویژه دیگری که نسبت به هم مستقل خطی بوده و دارای مقادیر ویژه متفاوتی هستند داشته باشد. در این صورت می‌توان بردارهای ویژه را به صورت ستونی در یک ماتریس V کنار هم قرار داد به طوری که درایه‌های هر ستون ماتریس یک بردار ویژه باشد. اگر ماتریس V بردارهای ویژه را تشکیل داده و مقدارهای ویژه نظیر هر کدام را به ترتیب ستون های ماتریس بردارها کنار هم قرار داده و ماتریس $\lambda_{1\times n}$ را تشکیل دهیم در این صورت تجزیه ویژه ماتریس مربعی A به صورت رابطه ۱۹-۲ تعریف می شود.

$$A = Vdiag(\lambda)V^{-1} \qquad (۲-۱۹)$$

در رابطه فوق عملگر $diag()$ یک ماتریس $n \times 1$ یا $1 \times n$ را به یک ماتریس قطری تبدیل می‌کند. ماتریس قطری یک ماتریس مربعی است که همه درایه های آن به جز درایه های قطر اصلی، قطری که از درایه $1,1$ شروع و به درایه n,n ختم می‌شود، صفر هستند، قطر اصلی و فرعی فقط برای ماتریس های مربعی تعریف می‌شود؛ قطر فرعی از درایه $1,n$ شروع و به درایه $n,1$ ختم می شود.

در صورتی که مقادیر یک ماتریس $m \times n$، بیانگر m نقطه در یک فضای برداری n بعدی باشد، در این صورت ستون هایی از ماتریس که مقادیر ویژه آن ها مثبت است، متناظر بعدهای مستقل فضا، بعدهایی که فضا را افراز می کنند-بعدهای متعامد-بعدهای اصلی، هستند، سایر ابعاد فضا که ستون متناظر با آن ها دارای مقدار ویژه غیرمثبت است، ابعاد وابسته به ابعاد اصلی بوده و جز بعدهای افراز کننده فضا محسوب نمی شوند در ماتریسی که ستون های آن متناظر با هر یک ابعاد یک فضا باشد مرتبه ماتریس[1] برابر با تعداد ستون های دارای مقادیر ویژه مثبت خواهد بود. در صورتی که همه ستون ها در چنین ماتریسی دارای مقادیر ویژه مثبت باشند آن ماتریس یک ماتریس مرتبه کامل[2] است.

۲.۷.۳ محاسبه مقادیر و بردارهای ویژه

رابطه ۱۷-۲ را می توانیم به صورت روابط ۲۰-۲ نیز نشان دهیم، بدین ترتیب از رابطه ۲۰-۲ می توانیم با اضافه کردن ماتریس همانی I رابطه ۲۱-۲ و از رابطه ۲۱-۲ نیز رابطه ۲۲-۲ را استنباط کنیم.

$$\boldsymbol{Av - \lambda v = 0} \qquad (۲-۲۰)$$
$$\boldsymbol{Av - \lambda Iv = 0} \qquad (۲-۲۱)$$
$$\boldsymbol{(A - \lambda I)v = 0} \qquad (۲-۲۲)$$

[1] Matrix rank
[2] Full rank matrix

همان طور که اشاره کردیم بردار ویژه $\boldsymbol{v}$ همواره مخالف صفر است، $\boldsymbol{v} \neq 0$، بنابراین برای این که رابطه ۲-۲۲ صدق کند باید رابطه ۲-۲۳ برقرار باشد. با حل معادله حاصل از رابطه ۲-۲۳ مقادیر ویژه ماتریس محاسبه می شود. پس از محاسبه مقادیر ویژه می توانیم با استفاده از رابطه ۲-۱۷ یا ۲-۱۸ بردارهای ویژه ماتریس را نیز محاسبه کنیم.

$$|A - \lambda I| = 0 \qquad (۲-۲۳)$$

۲.۸ نرم[1]

برای مقایسه بردارها مختلف از معیارهایی استفاده می‌شود که یکی از آن‌ها تابع نرم است. تابع نرم هر بردار را به یک مقدار عددی مثبت نگاشت می‌کند که بیانی از اندازه آن بردار است. در حالت کلی نرم یک بردار با نماد L^p نشان داده می شود که در آن $p \in \mathbb{R}, p \geq 0$ است. نرم بردار A طبق رابطه ۲-۲۴ محاسبه می‌شود.

$$\|A\|_p = (\sum_i |x_i|^p)^{\frac{1}{p}} \qquad (۲-۲۴)$$

رابطه ۲-۲۴ یک تابع نرم است، در حالت کلی هر تابعی می تواند تابع نرم f باشد اگر هر سه ویژگی زیر را داشته باشد:

- $f(\boldsymbol{x}) = 0 \rightarrow \boldsymbol{x} = 0$
- $f(\boldsymbol{x} + \boldsymbol{y}) \leq f(\boldsymbol{x}) + f(\boldsymbol{y})$
- $\forall a \in \mathbb{R}, f(a\boldsymbol{x}) = |a| f(\boldsymbol{x})$

در ادامه به بررسی چند حللت خاص برای تابع نرم می‌پردازیم، روابط این نرم ها برای بردارها تعریف شده است ولی می توانیم آن ها را به ابعاد بالاتر تعمیم داده و برای ماتریس ها و تانسورها نیز تعریف کنیم:

۲.۸.۱ نرم اقلیدسی

اگر $p = 2$ باشد در این حالت تابع نرم، به نرم اقلیدسی یا فاصله اقلیدسی مرسوم بوده و با نماد L^2 نشان داده می‌شود. این نرم کاربرد زیادی در مبحث مختلف از جمله یادگیری ژرف دارد. این نرم با ریشه دوم مجموع مربعات مولفه‌های بردار برابر است. رابطه ۲-۲۵، رابطه نرم اقلیدسی یک بردار است.

$$\|\boldsymbol{x}\|_2 = \sqrt{\sum_i (x_i)^2} \qquad (۲-۲۵)$$

[1] Norm

۲.۸.۲ نرم یک

نرم یک که با نماد L^1 نشـان داده می‌شـود. این نرم برابر اسـت با مجموع قدرمطلق مؤلفه های بردار. رابطه ۲۶-۲، نرم یک بردار $\boldsymbol{x}$ است.

$$\|\boldsymbol{x}\|_1 = \sum_i |x_i| \qquad (۲-۲۶)$$

۲.۸.۳ نرم بی‌نهایت

نرم بی‌نهایت یک بردار با نماد L^∞ نشـــان داده می‌شـــود. این نرم برابر اســـت با بزرگترین قدرمطلق مولفه بردار، در رابطه ۲۷-۲ این نرم برای بردار $\boldsymbol{x}$ نشان داده شده است.

$$\|\boldsymbol{x}\|_\infty = \max_i(|x_i|) \qquad (۲-۲۷)$$

۲.۸.۴ نرم ماتریس

تعاریف و روابطی فوق للذکر که برای نرم ها ارائه کردیم صـــرفا برای بردار ها تعریف می شـوند، در حالی که می توان برای یک ماتریس یا تانسـور نیز نرم تعریف کرد. یک تابع اگر علاوه بر سـه ویژگی مطرح شـده برای تابع نرم بردار، ویژگی رابطه ۲۸-۹ را نیز داشـته باشـد تابع نرم ماتریس یا تانسور است.

$$f(\boldsymbol{x}.\boldsymbol{y}) \leq f(\boldsymbol{x}).f(\boldsymbol{y}) \qquad (۲-۲۸)$$

۲.۸.۴.۱ نرم فربینوس[1]

یکی نرم های ماتریس که کاربرد زیادی دارد، نرم فربنیوس اســـت. نرم فربنیوس برای ماتریس A با نماد $\|A\|_F$ نشان داده شده و مطابق رابطه ۲۹-۲ محاسبه می شود.

$$\|A\|_F = \sqrt{\sum_{i,j} {A_{i,j}}^2} \qquad (۲-۲۹)$$

۲.۹ خواص کلی ماتریس ها

ماتریس‌ها دارای خواص عمومی ذیل هســـتند، این خواص قابل تعمیم به تانســـورها نیز هستند:

- $A + B = B + A$
- $A(B + C) = AB + AC$
- $A(BC) = (AB)C$

[1] Frobenius norm

- $(AB)^T = B^T A^T$
- $A^T B = B^T A$
- $A^T B = (A^T B)^T = B^T A$
- $\det(A^{-1}) = \frac{1}{\det(A)}$

۲.۱۰ مسائل

۱- در مورد فضاهای برداری موهومی[1] تحقیق کنید.

۲- در مورد روش‌های عددی برای محاسبه دترمینان ماتریس‌ها با ابعاد بزرگ تحقیق کرده و آن‌ها را از جنبه‌های مختلف با هم مقایسه کنید.

۳- در مورد نحوه استنباط رابطه ۲۳-۲ با استفاده از رابطه ۱۳-۲ توضیح دهید.

۴- نحوه محاسبه مرتبه[2] یک ماتریس و ارتباط آن با وابستگی خطی را توضیح دهید.

[1] Imaginary
[2] Rank

فصل ۳
آمار[۱] و احتمال[۲]

۳.۱ مقدمه

مباحث مطرح در آمار به بررسی ویژگی های مختلف در جوامع و مجموعه ها می پردازد. در احتمالات نیز به بررسی شرایط وقوع حالات مختلف تعریف شده در یک مجموعه پرداخته می شود. در این فصل ابتدا به تعاریف کلی که در معرفی مباحث مورد نیاز است پرداخته و سپس روابط و تعاریف مطرح در آمار و احتمالات را که در بخش های گوناگون موضوعات یادگیری ماشین و یادگیری ژرف کاربرد دارند را بررسی می کنیم.

۳.۲ تعاریف آمار

در این قسمت تعاریف اولیه در مباحث آماری را معرفی می کنیم:

جمیعت[۳]: جمعیت مجموعه ای شامل همه نمونه[۴] هایی است که هدف بررسی آن ها است. مجموعه خوردوهای یک سازمان و یا مجموعه افراد یک شهر نمونه هایی از جمعیت هستند که شماره هر یک از خودروها و هر یک از افراد شهر به ترتیب نمونه های هر یک از این جمعیت ها هستند.

متغیر[۵]: متغیر ویژگی است که برای همه نمونه های یک جمعیت تعریف می شود. یک متغیر می تواند کمی[۶] یا کیفی[۷] باشد. متغیرهای کمی می توانند دارای مقادیر پیوسته[۸] یا گسسته[۹] باشند، متغیر های پیوسته در بازه تعریف شده می توانند هر مقداری داشته باشند در حالی که متغیرهای گسسته فقط مقادیر تعیین شده در بازه را اختیار می کنند. برای مثال در جمعیت افراد یک شهر، وضعیت تاهل افراد یک متغیر کیفی، اندازه قد افراد یک متغیر کمی پیوسته و

---

[1] Statistics
[2] Probability
[3] Population
[4] Sample
[5] Parameter
[6] Quantitative
[7] Qualitative
[8] Continuous
[9] Discrete

در جمعیت خودروهای یک سازمان، تعداد چرخ های هر خودرو یک متغیر کمی گسسته است.

فضای نمونه ای[1]: فضای نمونه ای مجموعه ای شامل همه حالات ممکن است که یک متغیر می تواند اختیار کند. یک فضای نمونه ای نیز مانند متغیر به صورت کیفی، کمی گسسته و کمی پیوسته تعریف می شود. مجموعه اعداد طبیعی ۱ تا ۲۰، بازه فرضی، فضای نمونه ای کمی گسسته متغیر تعداد چرخ های خودروها، مجموعه اعداد حقیقی در بازه ۱متر تا ۲ متر فضای نمونه ای کمی پیوسته متغیر قد افراد و مجموعه ای شامل دو حالت مجرد و متاهل فضای نمونه ای کیفی برای متغیر وضعیت تاهل افراد است.

توزیع احتمالاتی[2]: یک توزیع احتمالاتی با نسبت دادن احتمال وقوع هر یک از حالات تعریف شده در فضای نمونه ای برای متغیر متناظر با آن تعریف می شود. برای در صورتی که احتمال وقوع همه حالات فضای نمونه ای با هم برابر باشد، هر یک از حالات دارای مقداری احتمالی برابر با $\frac{1}{n}$، تعداد اعضای فضای نمونه ای $= n$، بوده و توزیع احتمالاتی یک توزیع یکنواخت[3] خواهد بود. با توجه به متغیری که توزیع احتمالاتی برای آن تعریف می شود، توزیع می تواند کیفی، کمی گسسته و کمی پیوسته باشد. در ادامه چند نمونه از توزیع های احتمالاتی خواهیم پرداخت.

۳.۳ تعاریف احتمال

در این بخش به معرفی تعاریف اولیه مبحث احتمالات می پردازیم، برخی از این تعاریف مشابه تعاریف ارائه شده برای مباحث آمار بوده و با تعریفی مشترک از دیدگاه احتمالاتی بررسی نیز می شوند.

تجربه[4] (آزمایش): یک تجربه مجموعه ای از شرایط او قیودی است که تحت آن شرایط و قیود مقدار یک متغیر بررسی می شود. تجربه معادل جمعیت در تعاریف احتمال است که برای آن همانند جمعیت، نمونه، متغیر و فضای نمونه ای تعریف می شود. در صورتی که با توجه به شرایط و قیود تعریف شده مقدار متغیر مورد نظر در یک تجربه قابل پیش بینی و محاسبه نباشد آن تجربه یک تجربه تصادفی[5] و متغیر نیز یک متغیر تصادفی[6] است. دو پرتاب متوالی تاس یک تجربه تصادفی است که در آن عدد تاس پرتاب اول و دوم، دو متغیر مربوط به این تجربه و مجموعه ای شامل همه جفت اعدادی که بین یک تا شش تعریف می شوند ، $\{(1,1),(2,1),\ldots,(6,1),\ldots,(1,2),\ldots,(6,6)\}$، فضای نمونه ای این تجربه است. مجموعه

[1] Sample space
[2] Probability distribution
[3] Uniform probability distribution
[4] Experiment
[5] Random experiment
[6] Random variable

دادگانی که برای مباحث یادگیری ژرف به کار برده می شوند شامل نمونه های دارای متغیرهای تصادفی هستند که در طی روند یک تجربه تصادفی جمع آوری می شوند، متغیرها در این دادگان به صورت اسکالر، بردار، ماتریس و.... تعریف می شوند.

پیشامد[1]: به هر یک از زیر مجموعه های فضای نمونه ای یک تجربه، پیشامد گفته می شود.

۳.۴ انواع احتمال

۳.۴.۱ احتمال

بسیاری از پدیده ها و مباحثی که مورد بررسی و تحقیق قرار می گیرند یک تجربه تصادفی محسوب می شوند. از این رو نمی توان پیش بینی دقیقی از وقوع هر پیشامد تعریف شده از فضای نمونه ای داشت، در این صورت احتمال به عنوان معیاری تعریف می شود که به هر یک از پیشامدها وزنی نسبت داده و یک پیش بینی تقریبی از شانس وقوع آن ارائه می کند. احتمال یک پیشامد در حالت کلی به صورت نسبتی از تعداد نمونه هایی که متغیر آن ها عضو پیشامد مد نظر است به تعداد اعضای فضای نمونه ای تجربه مطابق رابطه ۱-۳ تعریف می شود. برای مثال طبق رابطه ۱-۳ احتمال پیشامد اعداد کوچکتر از سه، $\{1,2\}$، در تجربه تصادفی پرتاب یک تاس با فضای نمونه ای $\{1,2,3,4,5,6\}$ برابر با $\frac{2}{6}$ است. در رابطه ۱-۳، n تعداد اعضای یک مجموعه، A مجموعه پیشامد و S مجموعه فضای نمونه ای است.

$$P_{(A)} = \frac{n(A)}{n(S)} \qquad (۳-۱)$$

مجموع احتمال کل پیشامدهای تعریف شده برای یک فضای نمونه ای در صورتی که پیشامدها دارای عضو مشترک نباشند همواره برابر با یک است. برای مثال در پرتاب یک تاس مجموع احتمال پیشامد اعداد کوچکتر مساوی دو و بزرگ تر از دو مطابق رابطه ۲-۳ است.

$$P_{(x\leq 2)} + P_{(x>2)} = \frac{2}{6} + \frac{4}{6} = 1 \qquad (۳-۲)$$

۳.٤.۲ احتمال مشترک[2]

احتمال مشترک حالتی است که دو پیشامد A و B به صورت همزمان اتفاق بیفتد. احتمال مشترک دو پیشامد A و B با نماد $P_{(A\cap B)}$ یا $P_{(AB)}$ نمایش داده می شود. در صورتی

[1] Event
[2] Joint probability

که مقدار احتمال مشترک دو پیشامد صفر باشد در این صورت دو پیشامد ناسازگار[1] هستند.

۳.۴.۳ احتمال شرطی[2]

احتمال شرطی، محاسبه احتمال وقوع پیشامد A در شرایطی است که بدانیم پیشامد B حتما اتفاق افتاده است. احتمال پیشامد A به شرط وقوع پیشامد B با نماد $P_{(A|B)}$ نمایش داده شده و مطابق رابطه ۳-۳ محاسبه می شود.

$$P_{(A|B)} = \frac{P_{(A\cap B)}}{P_{(B)}} = \frac{P_{(AB)}}{P_{(B)}} \qquad (۳-۳)$$

از رابطه ۳-۳ می توانیم رابطه ۳-۴ را برای محاسبه احتمال مشترک توسعه دهیم.

(۳-۴)

$$P_{(A\cap B)} = P_{(A|B)}P_{(B)} = P_{(B|A)}P_{(A)}$$

در حالت کلی می توانیم رابطه ۳-۴ را برای وقوع چند پیشامد به طور مشترک ،احتمال ضرب چند پیشامد، مطابق رابطه ۳-۵ تعمیم دهیم.

$$P_{(E_1E_2\ldots E_n)} = P_{(E_1)}P_{(E_2|E_1)}P_{(E_3|E_2E_1)}P_{(E_n|E_1E_2\ldots E_{n-1})} \qquad (۳-۵)$$

در صورتی که مقدار احتمال شرطی $P_{(A|B)}$ برابر با $P_{(A)}$ باشد در این صورت پیشامد A مستقل از پیشامد B است.

۳.۵ امید ریاضی[3]

امید ریاضی مقداری است که انتظار می رود متغیر تصادفی در یک نمونه که مقدار دقیق آن مشخص نیست اختیار کند. امید ریاضی با توجه به مقادیر آن متغیر در نمونه هایی که مقدار آن متغیر در آن ها تعریف شده است محاسبه شده و برابر است با مجموع ضرب های مقدار متغیر در نمونه هایی که مقدار آن تعریف شده است در احتمال وقوع آن مقدار برای متغیر که مطابق رابطه ۳-۶ تعریف می شود. در رابطه ۳-۶، N تعداد نمونه هایی است که مقدار متغیر x در آن ها تعریف شده است.

[1] Mutual exclusivity
[2] Conditional probability
[3] Expected value

$$E_{(x)} = \sum_{i=1}^{N} P_{(x_i)} x_i \qquad (۳-۶)$$

مقدار احتمال، $P_{(x_i)}$، برای وقوع هر یک از مقادیر متغیر مورد نظر، x_i، با توجه به تعداد تکرار آن مقدار در نمونه های جمعیت ،تجربه تصادفی) محاسبه می شود. در صورتی که احتمال همه مقادیر برای متغیر در برای نمونه ها با هم برابر باشد در این صورت مقدار احتمال هر یک از مقادیر متغیر برابر $P_{(x_i)} = \frac{1}{N}$ خواهد بود. در این صورت توزیع احتمالاتی متغیر مورد نظر یک توزیع یکنواخت بوده و رابطه ۶-۳ به صورت رابطه ۷-۳ خواهد بود که همان میانگین مقادیر آن متغیر است. با این توضیحات می توانیم چنین برداشت کنیم که رابطه ۶-۳ یک میانگین وزنی از مقادیر متغیر است که اگر وزن مقادیر برای متغیر با هم برابر باشند برابر با مقدار میانگین مقادیر متغیر خواهد بود.

$$E_{(x)} = \sum_{i=1}^{N} \frac{1}{N} x_i = \frac{1}{N} \sum_{i=1}^{N} x_i \qquad (۳-۷)$$

۳.۶ واریانس[1] و انحراف معیار[2]

واریانس یک متغیر معیاری از میزان پراکندگی نمونه های یک جمعیت، تجربه تصادفی، است. واریانس به صورت میانگین توان دوم اختلاف مقادیر متغیر در نمونه های جمعیت از میانگین جمعیت مطابق رابطه ۸-۳ تعریف می شود. در رابطه ۸-۳، $\bar{x}$ میانگین جمعیت و N تعداد نمونه های جمعیت است.

$$\sigma^2_{(x)} = \frac{1}{N} \sum_{i=1}^{N} (x_i - \bar{x})^2 \qquad (۳-۸)$$

به ریشه دوم، جذر، واریانس مطابق رابطه ۹-۳، انحراف معیار اطلاق می شود. هر چه مقدار واریانس یا انحراف معیار یک جمعیت بزرگ باشد بدین معنی است که مقادیر متغیر مورد بررسی در نمونه های آن جمعیت پراکندگی زیادی از مقدار میانگین آن متغیر جمعیت دارند در حالی که مقدار واریانس یا انحراف معیار یک جمعیت هر چه قدر کوچک باشد چنین استنباط می شود که مقادیر متغیر مورد بررسی در اطراف مقدار میانگین آن متمرکز شده و پراکندگی کمتری دارند.

$$\sigma_{(x)} = \sqrt{\sigma^2_{(x)}} \qquad (۳-۹)$$

[1] Variance
[2] Standard deviation

۳.۶.۱ تصحیح بسل[1]

رابطه ۸-۳ واریانس برای حالتی است که مقدار میانگین متغیر با استفاده از **همه** نمونه های قابل تعریف برای جمعیت از پیش محاسبه شده و مطابق رابطه ۷-۳ برای محاسبه آن فقط از تعدادی از نمونه ها استفاده نشده باشد. مقدار میانگینی که با استفاده از همه نمونه های قابل تعریف برای جمعیت محاسبه شده باشد میانگین جمعیت و واریانس حاصل از آن مطابق رابطه ۸-۳ واریانس آن جمعیت خواهد بود. در حالی که میانگینی که برای محاسبه آن فقط از تعداد محدودی از نمونه ها استفاده می شود تنها میانگین آن نمونه ها بوده و صرفا تقریبی از میانگین جمعیت است. تعریف همه نمونه های ممکن برای یک جمعیت در مواردی که متغیر مورد بررسی پیوسته باشد غیرممکن است، در صورتی که متغیر گسسته باشد نیز ممکن است تعریف همه نمونه ها باعث افزایش حجم محاسبات شود. بنابراین در اکثر موارد از میانگین نمونه ها مطابق رابطه ۷-۳ که میانگین تقریبی جمعیت است برای محاسبه واریانس استفاده می شود، در این صورت رابطه واریانس مطابق رابطه ۱۰-۳ تعریف می شود. تفاوت رابطه ۸-۳ و ۱۰-۳ اضافه شده ۱- در مخرج است که به آن تصحیح بسل گفته می شود.

$$\sigma^2_{(x)} = \frac{1}{N-1}\sum_{i=1}^{N}(x_i - \overline{x})^2 \qquad (۳-۱۰)$$

دلیل اضافه شدن ۱- در مخرج این است که ابتدا مقدار میانگین نمونه ای مطابق رابطه ۱۱-۳ (۷-۳) با استفاده از N نمونه از جمعیت محاسبه می شود، هر یک از این N نمونه می توانند در بازه تعریف شده برای متغیر x به طور مستقل مقداری تصادفی داشته باشند، با این توضیحات می توانیم چنین برداشت کنیم که برای مقادیر متغیر N در بازه تعریف شده برای مقادیر متغیر وابستگی و قیدی به مقادیر متغیر سایر نمونه ها تعریف نشده و از این نظر محاسبه مقدار میانگین دارای N درجه آزادی است.

$$\overline{x} = \frac{1}{N}\sum_{i=1}^{N} x_i \qquad (۳-۱۱)$$

حال اگر از مقدار میانگین رابطه ۱۱-۳ برای محاسبه واریانس استفاده کنیم رابطه ۱۲-۳ را برای محاسبه توان دوم اختلاف با میانگین هر یک از نمونه ها خواهیم داشت.

$$(x_i - \overline{x})^2 = (x_i - (x_1 + x_2, \ldots + x_i + \ldots + x_N))^2 \qquad (۳-۱۲)$$

با توجه به تکرار مقدار x_i در رابطه ۱۲-۳ می توانیم چنین نتیجه بگیریم که همواره مقدار یکی

[1] Bessel's correction

از نمونه ها در جملات مربوط به محاسبه واریانس مشابه جمله رابطه ۱۲-۳ به یکی از مقادیری که برای محاسبه میانگین استفاده شده می شود وابسته است. با این ایجاد این وابستگی بین مقدار متغیر x_i به مقدار میانگین $\bar{x}$، یکی از درجات آزادی حذف شده و به صورت تصحیح بسل $(N-1)$ در روابط ظاهر می شود. استفاده از ضریب بسل باعث بهبود تقریب واریانس نمونه ای شده و آن را به واریانس جمعیت نزدیک می کند.

۳.۷ توزیع احتمالاتی

در تعاریف مربوط به آمار اشاره کردیم که توزیع احتمالاتی، احتمال وقوع هر یک از مقادیر، $P_{(x_i)}$، را برای یک متغیر در بازه تعریف شده آن تعیین می کند. توزیع احتمالاتی می تواند دارای یک رابطه تحلیلی باشد که در این صورت مقادیر احتمال $P_{(x_i)}$ به صورت یک تابع تعریف می شوند به این توزیع های احتمالاتی توزیع پارامتریک[1] اطلاق می شود. همچنین بسیاری از توزیع های احتمالاتی نیز وجود دارند که دارای رابطه خاصی نیستند که به آن ها توزیع غیر پارامتریک[2] گفته می شود. معمولا توزیع های غیرپارامتریک توسط یک توزیع پارامتری تقریب زده می شوند. در ادامه به معرفی چند توزیع پارامتریک که در مباحث یادگیری ژرف کاربرد دارند می پردازیم.

۳.۷.۱ توزیع یکنواخت[3]

توزیع احتمالاتی یکنواخت یکی از ساده ترین توزیع های احتمالاتی است که برای انواع متغیرهای کیفی و کمی گسسته و پیوسته تعریف می شود. در توزیع یکنواخت احتمال وقوع هر یک از مقادیر برای متغیر با هم برابر است. اگر بازه تعریف شده برای یک متغیر کمی پیوسته x از عدد a تا b، $b > a$، باشد، در این صورت رابطه توزیع یکنواخت برای این متغیر به صورت رابطه ۳-۱۳ تعریف می شود.

$$P_{(x)} = \frac{1}{b-a} \qquad (۳-۱۳)$$

در صورتی که توزیع یکنواخت برای یک متغیر کمی گسسته که شامل n مقدار است تعریف شده باشد در این صورت رابطه آن مطابق رابطه ۱۴-۳ خواهد بود.

$$P_{(x)} = \frac{1}{n} \qquad (۳-۱۴)$$

[1] Parametric
[2] Non-parametric
[3] Uniform distribution

توزیع یکنواخت برای متغیرهای کیفی نیز به صورت رابطه ۳-۱۴ تعریف می شود که در آن n تعداد وضعیت های تعریف شده برای متغیر است.

۳.۷.۲ توزیع گوسی[1]

توزیع گوسی یک توزیع احتمالاتی است که صرفا برای متغیرهای کمی پیوسته تعریف می شود. رابطه این توزیع برای متغیر x مطابق رابطه ۳-۱۵ تعریف می شود. در رابطه ۳-۱۵، $\bar{x}$ میانگین و σ انحراف معیار متغیر هستند.

$$P_{(x)} = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{1}{2}(\frac{x-\bar{x}}{\sigma})^2} \qquad (۳-۱۵)$$

در صورتی که مقدار میانگین و واریانس ،انحراف معیار) یک توزیع گوسی به ترتیب صفر و یک باشد در این صورت به توزیع استاندارد نرمال[2]، گوسی، اطلاق می شود. مطابق رابطه ۳-۱۶ می توانیم مقادیر یک متغیر با توزیع گوسی را به توزیع گوسی استاندارد نرمال تبدیل کنیم به این روند استاندارد سازی یا نرمال سازی امتیاز z[3] گفته می شود. در رابطه ۳-۱۶، x مقدار متغیر دارای توزیع گوسی با میانگین $\bar{x}$ و انحراف معیار σ است که تبدیل به مقدار z با توزیع گوسی استاندارد نرمال می شود.

$$z = \frac{x - \bar{x}}{\sigma} \qquad (۳-۱۶)$$

طبق قضیه حد مرکزی[4] در اکثر موارد در صورتی تعداد نمونه های یک جمعیت افزایش یابد ،داده های زیادی تولید شود)، توزیع آن نمونه ها به یک توزیع گوسی نزدیک خواهد بود، به همین دلیل در بسیاری از مسائل از توزیع گوسی برای تقریب رفتار توزیع داده های مورد بررسی استفاده می شود.

۳.۷.۳ توزیع برنولی[5]

توزیع احتمالاتی برنولی برای متغیرهای کیفی دو وضعیتی[6] تعریف می شود. رابطه توزیع برنولی مطابق رابطه ۳-۱۷ است.

---

[1] Gaussian distribution
[2] Normal standard distribution
[3] Z-score normalization
[4] Central limit theorem
[5] Bernoulli distribution
[6] Boolean

$$P_{(x)} = \begin{cases} q = 1 - p, & if \quad x = 0 \\ p, & if \quad x = 1 \end{cases} \qquad (۳-۱۷)$$

در رابطه ۳-۱۷، p احتمال وقوع وضعیت $x = 1$ است. رابطه ۳-۱۷ را می توانیم به صورت رابطه ۳-۱۸ نیز نشان دهیم. توزیع برنولی در مدل سازی مباحثی مانند تعیین فعال یا غیرفعال بودن مدل هایی مانند نورون عصبی در یادگیری ژرف به کار برده می شود.

$$P_{(x)} = p^x (1-p)^{1-x} \qquad (۳-۱۸)$$

۳.۷.۴ توابع مربوط به توزیع های احتمالاتی

در بخش های قبل برای معرفی هر یک توزیع های احتمالاتی برای آن ها یک تابع تعریف کردیم که به ازای هر ورودی مقدار احتمال متناظر با آن را محاسبه می کرد مانند توابع رابطه ۳-۱۳، ۳-۱۵ و...، به این توابع، تابع چگالی احتمال[1] توزیع گفته می شود. یکی دیگر از توابعی که برای توزیع های احتمالاتی تعریف می شود تابع توزیع تجمعی[2] است، تابع توزیع تجمعی مطابق رابطه ۳-۱۹ محاسبه می شود. در رابطه ۳-۱۹، $F_{(x)}$ تابع توزیع تجمعی و $P_{(x)}$ تابع چگالی احتمال است.

$$F_{(x)} = \int_{-\infty}^{x} P_{(x)} dx \qquad (۳-۱۹)$$

رابطه ۳-۱۹ برای توزیع های گسسته و یا کیفی به صورت مجموع احتمال حالت ها از ابتدا تا حالت متغیر x تعریف می شود. معمولا برای نمایش مشخصات یک توزیع از نمودار تابع چگالی احتمال و تابع توزیع تجمعی آن ها به ازای مقدار مختلف متغیر x استفاده می شود.

۳.۸ توزیع تجربی[3]

تابع چگالی احتمال یک رابطه تئوری است و ممکن است توزیع احتمالاتی متغیر مربوط به یک پدیده از آن رابطه پیروی نکند در این صورت توزیع اصلی متغیر آن پدیده یک توزیع غیرپارامتریک بوده و تابع چگالی احتمال تعریف شده صرفا یک تقریب از آن توزیع است. توزیع واقعی آن پدیده با تقسیم بازه عددی متغیر به بازه های مساوی و محاسبه چگالی احتمال متناظر با هر بازه به صورت یک اسکالر که نسبت تعداد نمونه های بازه به کل نمونه های جمع

[1] Probability density function (PDF)
[2] Cumulative distribution function (CDF)
[3] Empirical distribution

آوری شده از مقادیر آن متغیر است محاسبه می شود. در نهایت با کنار هم قرار دادن این مقادیر چگالی به ازای بازه های مختلف توزیع تجربی، نمونه ای، متغیر تعریف می شـــود. به نمودار متناظر با این بازه ها هیستوگرام[1] توزیع گفته می شود.

[1] Histogram

۳.۹ مسائل

۱. در مورد مسئله مونتی هال[1] تحقیق کنید.

۲. رابطه ۱۹-۳ را برای توزیع یک متغیر کمی گسسته بنویسید.

۳. در مورد توزیع دو جمله ای[2] تحقیق کرده و ارتباط آن با توزیع برنولی را شرح دهید.

۴. رابطه ۱۵-۳ بیانگر توزیع گوسی یک بعدی است، رابطه این توزیع را برای متغیرهای چند بعدی توسعه دهید.

[1] Monty Hall problem
[2] Binomial distribution

فصل ۴
داده کاوی[1]

۴.۱ مقدمه

در این فصل به معرفی انواع دادگانی که برای روش های یادگیری ژرف به کار برده می شوند می پردازیم. همچنین دسته بندی و ساختارهای مختلف آن ها را مطرح می کنیم. در ادامه چند نمونه از روش هایی که برای پردازش و ذخیره دادگان استفاده می شود را بررسی کرده و در نهایت در مورد مفاهیم تئوری اطلاعات بحث می کنیم. در انتهای فصل نیز مسائل مربوط برای درک بهتر مفاهیم و مطالب طرح شده در طول فصل تعریف شده است.

۴.۲ دسته بندی دادگان

به طور کلی به هر نوع اطلاعات و دانش داده[2] گفته می شود. در تعاریف مربوط به یادگیری ژرف و علوم داده[3] به مجموعه ای از نمونه های ثبت شده برای دسته ای متغیرها، مجموعه دادگان[4] گفته می شود. با توجه به گستره وسیع دادگان می توان آن ها را از نظر محتوا، حجم، ساختار و... به دسته های مختلفی تقسیم بندی کرد. یکی از مهم ترین تقسیم بندی هایی که در حوزه علوم داده به دادگان نسبت داده می شود، تقسیم بندی آن ها به دو نوع دادگان ساختار یافته[5] و غیر ساختار یافته[6] است. دادگان ساختار یافته مجموعه دادگانی است که در آن ها الگوی مشخصی، رابطه[7]، برای ساختار داده تعریف شده و نمونه های مجموعه دادگان از آن الگو پیروی کرده و بر اساس آن تعریف می شوند. در مقابل دادگان ساختار یافته، دادگان غیر ساختار یافته تعریف می شود که بر خلاف دادگان ساختار یافته، این دادگان الگوی ساختار مشخصی ندارند. در ادامه به بررسی روش های ذخیره سازی و پردازش هر دو دسته دادگان ساختار یافته و غیر ساختار یافته پرداخته و برای هر کدام مثال هایی مطرح می کنیم.

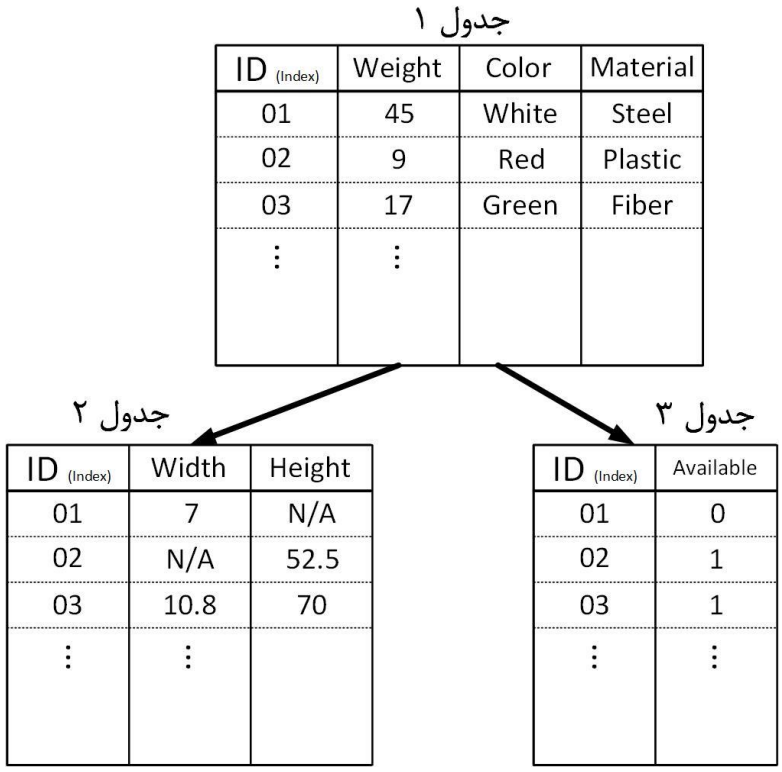
---

[1] Data mining
[2] Data
[3] Data science
[4] Data set
[5] Structured data
[6] Unstructured data
[7] Relation

۴.۲.۱ دادگان ساختار یافته

با توجه به تعریف الگوی ثابت در دادگان ساختار یافته می توانیم یک کلی برای این نوع مجموعه دادگان تعریف کرده و بر اساس آن نمونه های مجموعه دادگان را در یک پایگاه داده[۱] ذخیره کنیم. با توجه به ثابت بودن الگوی تعریف شده به ازای همه نمونه های مجموعه دادگان استفاده از پایگاه های داده رابطه ای[۲]، SQL[۳]، برای این دادگان مرسوم تر است. پایگاه های داده رابطه ای از حداقل یک جدول[۴] تشکیل شده اند که مقادیر مربوط به هر نمونه مانند ماتریس در آن جدول ذخیره می شوند، هر یک از سطر ها در جدول یک نمونه و هر ستون متغیرهای تعریف شده هستند. شکل ۱-۴ جداول یک پایگاه داده فرضی را نمایش می دهد، همان طور که در جدوال شکل ۱-۴ نشان داده شده است در پایگاه های داده رابطه ای، رابطه بین جداول به واسطه یک ستون مشترک، ستون نمایه[۵]، تعریف می شود؛ ستون نمایه می تواند هر یک از ستون های جدول باشد به طوری که مقادیر آن برای هر سطر، نمونه، منحصر به فرد است. ذخیره سازی اطلاعات در پایگاه های داده رابطه ای به دو روش سطرگرا[۶] و ستون گرا[۷] صورت می گیرد؛ در روش سطرگرا، هر سطر، نمونه ، از جدول به طور مجزا ذخیره می شود در حالی که در روش ستون گرا، هر یک از ستون های جدول به صورت مستقل ذخیره می شوند؛ با توجه به مجزا بودن ستون های یک جدول در روش ستون گرا، برای مرتبط کردن ستون های مختلف جدول، ستون نمایه نیز در هر ستون به صورت کلید-مقدار[۸] ذخیره می شود که در آن کلید مربوط به نمایه و مقدار مربوط به مقادیر متناظر با نمایه در یک ستون است.

[1] Database
[2] Relational database
[3] Structured query language
[4] Table
[5] Index
[6] Row-oriented
[7] Column-oriented
[8] Key-Value



شکل ۱-۴: جداول یک پایگاه داده رابطه ای

در مجموعه دادگانی که در روش های یادگیری ژرف به کار می روند برای همه نمونه های مجموعه دادگان متغیرهای یکسانی تعریف می شود، بنابراین اگر مجموعه دادگان به صورت ساختار یافته باشند باید هر نمونه، سطر، از جدول به ازای ستون های تعریف شده دارای مقدار باشد. در صورتی که در بعضی از نمونه ها مقادیر مربوط به یک یا چند ستون تعریف نشده باشد[1] باید یکی از راهکارهای ذیل را برای آن نمونه های به کار ببریم؛ برای مثال در جداول شکل۱-۴ مقادیری که با نماد N/A[2] مشخص شده اند، تعریف نشده هستند.

- نمونه های دارای مقادیر متغیر تعریف نشده را حذف کنیم، در صورتی که تعداد نمونه های مجموعه دادگان به حد کافی باشد استفاده از این روش معمولا مناسب ترین روش است.
- سطر، متغیر، مربوط که در آن مقادیر تعریف نشده قرار گرفته اند را حذف کنیم، این روش در صورتی که سطر مورد نظر به عنوان خروجی مدل یادگیری ژرف تعریف شده

[1] Missing value
[2] Not applicable

باشد امکان پذیر نبوده و فقط در صورتی که ستون از متغیر، ویژگی، های ورودی مدل باشد اجرا می شود. استفاده از این روش در صورتی که مقادیر مربوط به ستون دارای پراکندگی مناسبی باشند، واریانس بزرگی داشته باشند، ممکن عملکرد مدل را تضعیف کند.

- برای متغیرهایی که در برخی از نمونه های مجموعه دادگان مقادیر آن ها تعریف نشده است مقادیر تقریبی در نظر می گیریم؛ در این روش برخلاف دو روش پیشین هیچ یک از بخش های مجموعه دادگان حذف نمی شود. این مقدار تقریبی می تواند مقداری باشد که بیشترین تکرار را در بین نمونه هایی که مقدار آن تعریف شده است دارد[1]؛ در حوزه داده کاوی الگوریتم های مختلفی مانند Apriori، FP-Growth و... برای پیدا کردن مواردی که بیشترین تکرار را در مجموعه دادگان دارند ارائه شده است که بررسی آن ها خارج از حوزه مطالب این کتاب است. همچنین اگر متغیر مورد نظر کمی پیوسته باشد مقدار تقریبی که برای آن در نظر می گیریم می تواند امید ریاضی آن متغیر باشد که با استفاده از سایر نمونه هایی که مقدار آن متغیر در آن ها تعریف شده است محاسبه می شود. همچنین می توان با استفاده از معیارهای شباهت مانند معیار Cosine similarity مقادیر متغیر مربوط به نمونه ای که بیشترین شباهت را در بین نمونه های مجموعه دادگان به نمونه مورد نظر دارد برای متغیر تعریف نشده نمونه در نظر بگیریم؛ در این روش شباهت براساس متغیر هایی، ویژگی هایی، که در نمونه مورد نظر تعریف شده است محاسبه می شود. رابطه ۱-۴ رابطه معیار Cosine similarity است، که در آن $\boldsymbol{a}, \boldsymbol{b}$ دو نمونه، سطر، از مجموعه دادگان هستند که به صورت دو بردار شامل متغیرهایی که مقادیر آن ها در هر دو نمونه تعریف شده است در نظر گرفته می شوند، N نیز تعداد متغیرهای مشترک بین دو بردار، نمونه، است.

$$Cosine-Similarity(\mathbf{a},\mathbf{b}) = \frac{\mathbf{a}.\mathbf{b}}{\|\mathbf{a}\|_2 . \|\mathbf{b}\|_2} = \frac{\sum_{i=1}^{N} a_i.b_i}{\sqrt{\sum_{i=1}^{N} a_i^2}\sqrt{\sum_{i=1}^{N} b_i^2}} \qquad (۴-۱)$$

دادگانی که در پایگاه های داده غیر رابطه ای[2]، Non-SQL، ذخیره سازی می شوند نیز جز دسته دادگان ساختار یافته محسوب می شوند، در این پایگاه های داده هر نمونه ممکن است الگوی متفاوتی داشته و تعداد متغیرهای با نمونه های دیگر متفوت باشد. با این توضیحات با توجه به لزوم تعریف متغیرهای یکسان در همه نمونه های یک مجموعه

[1] Frequent item
[2] Non-relational database

دادگانی که در یک مدل یادگیری ژرف به کار برده می شـــوند، در صـــورتی که مجموعه دادگان در یک پایگاه داده غیر رابطه ای ذخیره شـده باشـند باید الگوی تعریف شـده برای نمونه های مختلف پیش از اسـتفاده از دادگان در مدل یادگیری ژرف با هم یکسـان سـازی شوند.

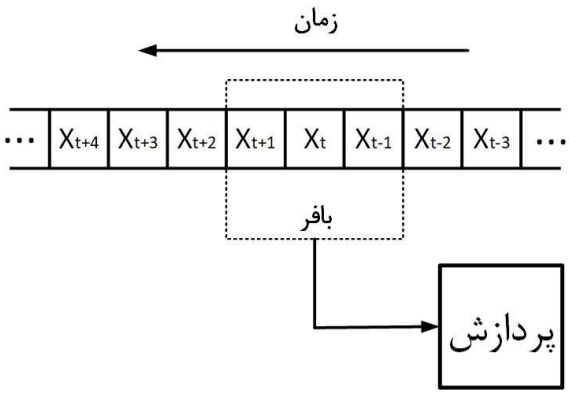
۴.۲.۲ دادگان غیرساختار یافته

در مقابل دادگان سـاختار یافته دادگان غیر سـاختار یافته به صـورت دادگانی تعریف می شـوند که در آن ها الگوی مشـخصـی برای نمونه های مجموعه تعریف نمی شـود؛ به عبارت دیگر در دادگان ســاختار یافته نمی توان مجموعه متغیرهای یک نمونه را مانند شــکل ۱-۴ دســته بندی کرد. دادگان تصویری، متن، سیگنال های زمانی مانند صوت و... مثال هایی از دادگان غیر ساختار یافته هستند. ممکن است هر یک از نمونه های مجموعه دادگان غیر ساختار یافته دارای برچسب[1] های ساختار یافته باشند که به عنوان خروجی مطلوب یک مدل یادگیری ژرف به ازای هر نمونه ورودی در نظر گرفته می شود.

۴.۳ پردازش جریان[2]

در تعاریف بخش گذشته فرض کردیم که مجموعه دادگان در یک پایگاه داده ذخیره شده باشـند، در صـورتی که در مواردی با توجه حجم و سـرعت بالای تولید دادگان ممکن اسـت امکان ذخیره همه دادگان و ســپس پردازش آن ها وجود نداشــته و یا هدف ایجاد یک مدل پردازش برخط[3] باشــد. در این صــورت صــرفا تعداد محدودی از نمونه های دادگان در بازه های زمانی متوالی، یکسـان، ذخیره[4] و به صـورت برخط مطابق شـکل ۲-۴ پردازش می شـوند. به این روش پردازش برخط، پردازش جریان اطلاق می شــود. در این روش پردازش می تواند به صــورت الگوریتم های مختلفی مانند نمونه برداری، محاســبه ویژگی های آماری، آموزش یک مدل یادگیری ژرف یا استنتاج[5] از آن تعریف شود.

[1] Label
[2] Stream processing
[3] Online
[4] Buffer
[5] Inference



شکل ۲-۴: پردازش جریان مقادیر متغیر X

۴.۴ پردازش دادگان حجیم[1]

در مواردی اگر حجم دادگان ذخیره شده بالا باشد به دلیل محدودیت های منابع محاسباتی امکان پردازش آن ها با روش های مرسوم وجود ندارد. در این صورت دادگان در منابع محاسباتی جداگانه توزیع شده[2] و در هر منبع بخشی از محاسبات مربوط به پردازش تعریف شده انجام می شود. الگوریتم نگاشت-کاهش[3] امکان پردازش دادگان توزیع شده در منابع محاسباتی مجزا[4] را فراهم می کند. مراحل اجرای الگوریتم نگاشت-کاهش را با یک مثال بررسی می کنیم؛ فرض کنید یک مجموعه دادگان حجیم که شامل نمونه هایی از کلاس های مختلف است را در اختیار داشته باشیم، در هر منبع محاسباتی بخشی از نمونه ها به صورت جداولی مشابه شکل ۱-۴ به طور مجزا ذخیره شده است، هدف از پردازش این دادگان محاسبه تعداد نمونه های هر کلاس در کل مجموعه دادگان است که برای محاسبه آن الگوریتم نگاشت-کاهش را به ترتیب زیر اجرا می کنیم:

۱. **نگاشت[5]:** این مرحله هر یک از منابع محاسباتی که بخشی از دادگان در آن قرار دارد به طور جداگانه اجرا می شود. این الگوریتم به ازای هر نمونه ای که در یک منبع ذخیره شده است یک کلید-مقدار تولید می کند. با توجه به تعریف مسئله در اینجا کلید معادل کلاس نمونه و مقدار برابر یک، {۱:کلاس}، در نظر گرفته می شود.

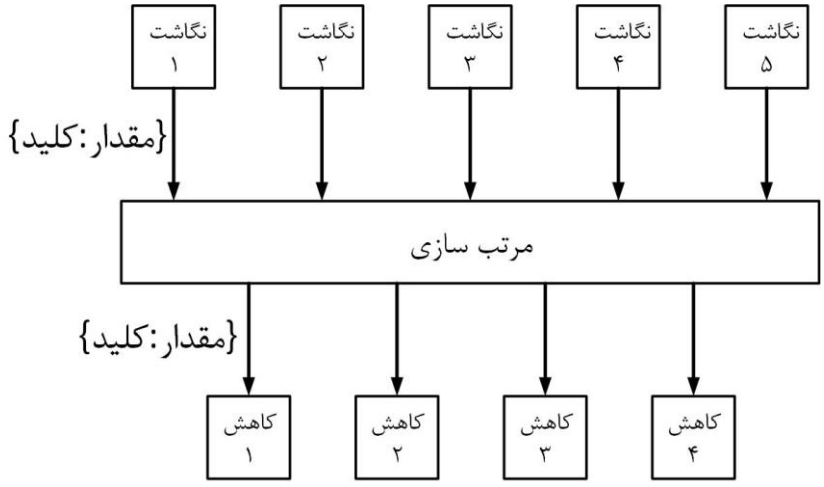
۲. **مرتب سازی[6]:** پس از اتمام اجرای الگوریتم نگاشت در همه منابع به ازای هر یک از نمونه های کل مجموعه دادگان یک کلید-مقدار تولید شده است. در این مرحله از

[1] Big data analysis
[2] Distributed
[3] Map-Reduce
[4] Node
[5] Mapping
[6] Sorting

روند الگوریتم کلید-مقدار ها بر اساس کلید مرتب سازی شده و کلید های یکسان به منبع کاهش یکسان ارسال می شوند.

۳. **کاهش**[1]: مرحله کاهش نیز همچون مرحله نگاشت از منابع محاسباتی مجزا تشکیل شده است که ورودی هر یک از آن ها همه کلید-مقدارهایی است دارای کلید مشابه هستند. در این صورت همه کلید-مقدارهای متناظر با نمونه های مربوط به یک کلاس در یک منبع کاهش مشترک قرار می گیرند. با ذخیره کلید-مقدارها با کلید مشترک در هر منبع کاهش، هر یک از این منابع کاهش مقدارهای همه کلید-مقدارهای ورودی را با هم جمع کرده و مقدار مجموع را به عنوان خروجی منبع ارائه می کنند که همان مجموع تعداد نمونه های یک کلاس است.

الگوریتم نگاشت-کاهش را می توان به انواع مسائل گوناگون تعمیم داد. شکل ۳-۴، شمای روند اجرای الگوریتم نگاشت-کاهش است.



شکل ۳-۴: شمای الگوریتم نگاشت-کاهش

۴.۵ رایانش موازی[2]

با توسعه و پیشرفت سخت افزارهای پردازشی که دارای تعداد هسته[3] های پردازشی بالایی هستند مانند واحدهای پردازش گرافیکی[4] و واحدهای پردازش تانسور[5] پیاده سازی و اجرای الگوریتم های مختلف به صورت موازی به ویژه در مواردی که یک الگوریتم حجم

[1] Reduce
[2] Parallel computing
[3] Core
[4] Graphics processing units (GPUs)
[5] Tensor processing units (TPUs)

محاسباتی بالایی دارد، مانند مدل های یادگیری ژرف، مورد توجه قرار گرفته است تا زمان مورد نیاز اجرا کاهش یابد. رایانش موازی در بخش هایی از الگوریتم که نسبت هبه طور مستقل اجرا می شـــوند به کار گرفته می شـــود. برای مثال الگوریتم نگاشـــت-کاهش بخش قبل را در نظر بگیرید در این الگوریتم، مراحل نگاشـــت منابع مختلف می توانند به صـــورت موازی به طور همزمان اجرا شوند در حالی که یک مرحله نگاشت را با یک مرحله کاهش نمی توان همزمان به صورت موازی اجرا کرد.

٤.٦ تئوری اطلاعات[1]

تئوری اطلاعات شـــامل مباحث گســـترده ای مانند محاســـبات مربوط به عدم قطعیت[2] دادگان، میزان فشرده سازی[3] اطلاعات، ظرفیت یک کانال ارتباطی[4] و... است که در رشته های مختلفی مانند علوم داده، علوم کامپیوتر، مهندسی برق و مخابرات و... به کار برده می شوند. در ادامه این بخش به بررسی موضوعاتی از تئوری اطلاعات می پردازیم که در یادگیری ژرف کاربرد دارند.

۴.۶.۱ آنتروپی[5]

آنتروپی مهم ترین مبحث تئوری اطلاعات است که هدف آن سنجش میزان عدم قطعیت در یک متغیر اسـت. مقدار آنتروپی همواره بزرگتر یا مسـاوی صـفر بوده و هر چه قدر مقدار آن بزرگ تر باشـــد میزان عدم قطعیت متغیر نیز بالاتر اســـت. برای هر یک از متغیرهای مجموعه دادگانی که متشـکل از نمونه های مسـتقل باشـد می توان میزان عدم قطعیت را مطابق رابطه ۲-۴ محاســـبه کرد. در رابطه ۲-۴، $H_{(x)}$ میزان آنتروپی متغیر $\boldsymbol{x}$ و $P_{(x)}$ احتمال مقدار x برای متغیر $\boldsymbol{x}$ اســـت. همچنین عدد مبنای لگاریتم، b، در رابطه ۲-۴ واحد آنتروپی را مشـــخص می کند؛ در صـــورتی که این مبنا برابر ۲، e و ۱۰ باشـد واحد آنتروپی به ترتیب بیت[6]، نات[7] و بان[8] خواهد بود.

$$H_{(\mathbf{x})} = -\sum_{\mathbf{x}} P_{(x)} \log_b P_{(x)} = -\mathrm{E}(\log_b P_{(x)}) \qquad (۴-۲)$$

[1] Information theory
[2] Uncertainty
[3] Compression
[4] Channel capacity
[5] Entropy
[6] Bit
[7] Nat
[8] Ban

۴.۶.۲ آنتروپی مشترک[1] و آنتروپی شرطی[2]

با توجه به ماهیت احتمالاتی رابطه آنتروپی می توان آن را به صورت آنتروپی مشترک و آنتروپی شرطی برای دو یا چند متغیر مختلف نیز تعریف کرد. آنتروپی مشترک میزان عدم قطعیت برای مقادیر مشترک دو متغیر $\boldsymbol{y},\boldsymbol{x}$و آنتروپی شرطی عدم قطعیت مربوط به متغیر $\boldsymbol{x}$ را در حضور متغیر $\boldsymbol{y}$ بررسی می کند. روابط ۳-۴ و ۴-۴ به ترتیب رابطه آنتروپی مشترک و شرطی دو متغیر $\boldsymbol{y},\boldsymbol{x}$ است، در این روابط y مقدار متغیر $\boldsymbol{y}$ است.

$$H_{(\mathbf{x},\mathbf{y})} = -\sum_{\mathbf{x},\mathbf{y}} P_{(x,y)} \log_b P_{(x,y)} = -\mathrm{E}_{\mathbf{x},\mathbf{y}}(\log_b P_{(x,y)}) \quad (۴-۳)$$

$$H_{(\mathbf{x}|\mathbf{y})} = -\sum_{\mathbf{y}} P_{(y)} \sum_{\mathbf{x}} P_{(x|y)} \log_b P_{(x|y)} = -\mathrm{E}_{\mathbf{y}}(H_{(\mathbf{x}|y)}) \quad (۴-۴)$$

با توجه به روابط ۳-۴ و ۴-۴ برای آنتروپی شرطی می توانیم رابطه ۵-۴ را تعریف کنیم.

$$H_{(\mathbf{x}|\mathbf{y})} = H_{(\mathbf{x},\mathbf{y})} - H_{(\mathbf{y})} \quad (۴-۵)$$

۴.۶.۳ اطلاعات متقابل[3]

اطلاعات متقابل یکی از مباحث مطرح در تئوری اطلاعات است که هدف محاسبه میزان وابستگی متقابل دو متغیر است. هر چه مقدار اطلاعات متقابل دو متغیر بزرگ تر باشد با بررسی یکی از آن ها می توان اطلاعات بیشتری از رفتار دیگری به دست آورد. اطلاعات متقابل دو متغیر $\boldsymbol{y},\boldsymbol{x}$ مطابق رابطه ۶-۴ محاسبه می شود. با محاسبه اطلاعات متقابل هر یک از متغیرهای ورودی مختلف با متغیرهای خروجی یک مدل می توان از آن به عنوان یک روش انتخاب ویژگی[4] استفاده کرده و متغیرهای، ویژگی های، مناسب را از بین مجموعه متغیرهای دادگان به عنوان ورودی مدل انتخاب کرد. در رابطه ۶-۴، $I_{(x;y)}$ اطلاعات متقابل دو متغیر $\boldsymbol{x}$ و $\boldsymbol{y}$ است.

$$I_{(\mathbf{x};\mathbf{y})} = \sum_{\mathbf{x},\mathbf{y}} P_{(x,y)} \log_b \frac{P_{(x,y)}}{P_{(x)} P_{(y)}} \quad (۴-۶)$$

برای اطلاعات متقابل دو متغیر می توانیم رابطه ۷-۴ و ۸-۴ را تعریف کنیم.

$$I_{(\mathbf{x};\mathbf{y})} = H_{(\mathbf{x})} - H_{(\mathbf{x}|\mathbf{y})} \quad (۴-۷)$$

---

[1] Joint entropy
[2] Conditional entropy
[3] Mutual information
[4] Feature selection

$$I_{(\mathbf{x};\mathbf{y})} = I_{(\mathbf{y};\mathbf{x})} \qquad (۴-۸)$$

۴.۷ مسائل

۱. در صورتی که یک داده غیر ساختار یافته مانند یک تصویر را در جدول یک پایگاه داده رابطه ای ذخیره کنیم می توانیم آن را داده ساختار یافته بنامیم؟ توضیح دهید.
۲. عملیات ضرب دو ماتریس را با استفاده از الگوریتم نگاشت-کاهش پیاده سازی کنید.
۳. مقدار آنتروپی پرتاب یک تاس را محاسبه کنید.
۴. در مورد واحدهای بیت، نات، بان مربوط به آنتروپی و اطلاعات متقابل که با تغییر مبنای لگاریتم در روابط حاصل می شود تحقیق و آن ها را با هم مقایسه کنید.

فصل ۵
پرداش سیگنال[1]

۵.۱ مقدمه

سیگنال تابعی است که ورودی آن زمان[2]، فضا[3] و... و خروجی آن اطلاعات است. مقادیر مربوط به تابع سیگنال، ورودی و خروجی تابع، دادگان غیر ساختار یافته ای هستند که به طور گسترده به عنوان مجموعه دادگان آموزشی مدل های یادگیری ژرف استفاده می شوند. سیگنال هایی که در حوزه یادگیری ژرف بررسی می شوند، سیگنال های گسسته[4]، دیجیتال[5] هستند که عموما از یک سیگنال پیوسته[6]، آنالوگ[7]، نمونه برداری می شوند. در این فصل ابتدا به بررسی حوزه های تعریف سیگنال پرداخته و سپس مباحث و چالش های موجود در مبحث نمونه برداری سیگنال ها و تبدیل های تعریف شده برای حوزه های مختلف را بررسی کرده و نحوه پردازش و پیش پردازش سیگنال در حوزه های مختلف را معرفی می کنیم. در انتهای فصل نیز مسائل مربوط به آن برای تعمیق بیشتر مطالب مطرح شده است.

۵.۲ حوزه[8] تعریف سیگنال

یک سیگنال می تواند در حوزه های مختلفی تعریف شود، منظور از حوزه تعریف سیگنال متغیری است که به عنوان ورودی تابع سیگنال در نظر گرفته می شود. حوزه زمان و فضا از مهم ترین حوزه های تعریف سیگنال هستند. سیگنال های حوزه زمان یک سری زمانی از مقادیر یک یا چند پارامتر متغیر با زمان، دامنه[9]، هستند، در حالی که سیگنال های حوزه فضا به صورت پارامتری که مقادیر آن با تغییر موقعیت در راستای محور های مختلف فضا تغییر می

---

[1] Signal processing
[2] Time
[3] Space
[4] Discrete
[5] Digital
[6] Continues
[7] Analog
[8] Domain
[9] Amplitude

کند تعریف می شـوند. صـوت، سـیگنال های رادیویی، سـیگنال الکتروانسـفالوگرافی[۱] و الکتروکاردیوگرافی[۲] نمونه هایی از سیگنال های حوزه زمان و انواع تصاویر سیاه و سفید، رنگی و سه بعدی نمونه هایی از سیگنال های حوزه فضا هستند، تصویر سیاه و سفید یک سیگنال حوزه فضای دو بعدی x,y است، به طوری که به هر یک از نقاط مختلف در این فضا مقداری نسبت داده شده و در نهایت به صورت مجموعه ای از مقادیر متناظر با نقاط گسسته تعریف شده در فضای دو بعدی در یک ماتریس ذخیره می شود، هر یک از درایه های این ماتریس معادل یک پیکسل[۳] از تصویر است. تصویر رنگی نیز یک سیگنال فضای سه بعدی x,y,z است که به صورت یک تانسور سه بعدی ذخیره می شود، بعد سوم، z، شامل سه کانال رنگی قرمز، سبز و آبی[۴] مربوط به هر پیکسل است به طوری که رنگ مربوط به هر پیکسل، درایه، که یک نقطه در فضای دو بعدی x,y است با کنار قرار دادن مقادیر بعد z تعریف می شود، برخی از فرمت های تصویر ممکن است از چهار کانال رنگی[۵] استفاده کنند. تصاویر سه بعدی اگر به صورت تک رنگ تعریف شده باشند، شامل مجموعه ای از واحد های سه بعدی گسسته هستند که به آن ها واکسل[۶] اطلاق می شود، مکان این واکسل ها در فضای سه بعدی x,y,z تعریف شده و به صورت یک تانسور سه بعدی تعریف می شوند در صورتی که این تصاویر سه بعدی رنگی باشند بعد چهارم تانسور که مربوط به کانال های رنگی هر واکسل است نیز تعریف شده و مقدار هر واکسل در موقعیت فضای سه بعدی به صورت مجموع مقادیر کانال های رنگی آن تعریف می شود. توالی زمانی تصاویر، ویدئو، نیز سیگنالی است که در هر دو حوزه زمان و فضا تعریف می شود سیگنال ویدئو های سیاه و سفید به صورت تانسور سه بعدی ذخیره می شوند که دو بعد مربوط به موقعیت پیکسل ها در فضای دو بعدی x,y و بعد سوم مربوط به زمان است در حالی که برای سیگنال ویدئوهای رنگی یک تانسور چهار بعدی تعریف می شود که دو بعد مربوط به موقعیت پیکسل ها در فضای دو بعدی x,y یک بعد مربوط به کانال های رنگی و بعد دیگر مربوط به زمان است.

۵.۳ تبدیل حوزه سیگنال به حوزه فرکانس و بالعکس

هر سیگنالی که در حوزه زمان یا فضا تعریف شده باشد دارای محتوای فرکانسی است که می تواند در محدوده باندهای فرکانسی مختلف باشد. محتوای فرکانسی یک سیگنال می تواند مانند سیگنال سینوسی[۷] فقط شامل یک مقدار و یا مانند اکثر سیگنال های طبیعی از مقادیر

[1] Electroencephalography (EEG)
[2] Electrocardiography (ECG)
[3] Pixel
[4] Red-Green-Blue (RGB)
[5] تصاویر CMYK
[6] Voxel
[7] Sinusoidal signal

فرکانسی مختلف در باندهای متفاوت باشد. با اعمال تبدیل های مناسب می توان سیگنال را از حوزه زمان یا فضا به حوزه فرکانس انتقال داده و محتوای فرکانسی و دامنه[1] متناظر با هر مقدار فرکانسی را محاسبه کرد. همچنین می توان با اعمال معکوس تبدیل سیگنال را از حوزه فرکانس به زمان یا فضا انتقال دهیم. در ادامه دو تبدیل حوزه زمان، فضا به فرکانس و معکوس آن ها را که کاربرد گسترده ای در مباحث پردازش سیگنال دارند بررسی می کنیم.

۵.۳.۱ تبدیل فوریه[2]

تبدیل فوریه یک سیگنال را به صورت مجموع سیگنال های سینوسی با فرکانس های مختلف تعریف کرده و به هر فرکانس مقدار دامنه متناظر با آن را نسبت می دهد، به عبارت دیگر تبدیل فوریه تابعی بر اساس سیگنال تعریف می کند که ورودی آن مقدار فرکانس و خروجی آ« دامنه مربوط به آن فرکانس است. رابطه تبدیل فوریه برای یک فرکانس دارای ورودی یک بعدی مانند سیگنال زمان مطابق رابطه ۱-۵ است. در رابطه ۱-۵، $f_{(t)}$ تابع حوزه زمان سیگنال ، t, ω به ترتیب فرکانس و زمان و j متغیر مربوط بخش موهومی[3] است. با توجه به اینکه تبدیل فوریه

$$F_{(\omega)} = \int_{-\infty}^{+\infty} f_{(t)} e^{-j\omega t} dt = \int_{-\infty}^{+\infty} f_{(t)} \cos(\omega t) dt - j \int_{-\infty}^{+\infty} f_{(t)} \sin(\omega t) dt \qquad (۵-۱)$$

با توجه به اینکه مقادیر یک سیگنال به صورت گسسته ذخیره می شوند، لازم است تبدیل فوریه رابطه ۱-۵ را به صورت گسسته[4] مطابق رابطه ۲-۵ تعریف کنیم.

$$F_{(\omega)} = \sum_{-\infty}^{+\infty} f_{(n)} e^{-j\omega n} \qquad (۵-۲)$$

تبدیل فوریه برای یک سیگنال با ورودی دو بعدی مانند سیگنال حوزه فضا، تصویر، به صورت رابطه ۳-۵ تعریف می شود. در رابطه ۳-۵ نیز x, y مربوط به ورودی حوزه فضا و u, v فرکانس های متناظر با هر یک از ابعاد فضا است.

$$F_{(u,v)} = \int \int_{-\infty}^{+\infty} f_{(x,y)} e^{-j(ux+vy)} dx dy \qquad (۵-۳)$$

رابطه ۳-۵ را می توانیم تعمیم داده تبدیل فوریه را برای سیگنال های دارای سه بعد و بیشتر نیز تعریف کنیم.

---

[1] Amplitude
[2] Fourier transform
[3] Imaginary
[4] Discrete Fourier transform

تبدیل فوریه یک روش تقریبی بوده و سیگنال را به صورت مجموع سیگنال های سینوسی مدل می کند بنابراین استفاده از آن در مواردی که سیگنال اولیه رفتار تناوبی نداشته باشد باعث ایجاد مقدار خطای تقریب بزرگی شده در نتیجه محتوای فرکانسی محاسبه شده نیز دقیق نخواهد بود. به دلیل وجود خطای تقریب بین سیگنال اولیه و سیگنال تقریبی حاصل، اعمال تبدیل فوریه باعث می شود بخشی از اطلاعات سیگنال اولیه از بین برود، برای مشاهده مقدار اطلاعات از بین رفته می توانیم پس از اعمال تبدیل فوریه، معکوس آن را برای رابطه حاصل اعمال کرده و نتیجه را با سیگنال اولیه مقایسه کنیم. معکوس تبدیل فوریه که سیگنال را از حوزه فرکانس به حوزه تعریف اولیه آن انتقال می دهد برای یک سیگنال با ورودی یک بعدی مانند سیگنال زمانی مطابق رابطه ۴-۵ است. در رابطه ۴-۵، ω_{max} بیشترین فرکانسی است که در تبدیل فوریه برای سیگنال اولیه در نظر گرفته شده است که در حالت تئوریکال مقدار آن بینهایت، ∞+، است.

$$f_{(t)} = \frac{1}{\pi} \int_{0}^{\omega_{\max}} F_{(\omega)} e^{j\omega t} d\omega \qquad (۵-۴)$$

برای پیاده سازی تبدیل فوریه معمولا از روش تبدیل فوریه سریع[1] استفاده می شود؛ تبدیل فوریه سریع یک الگوریتم پیشنهادی برای پیاده سازی تبدیل فوریه است که سرعت اجرای بالاتری نسبت به پیاده سازی معادله پایه تبدیل که در روابط ۱-۵ و ۳-۵ تعریف شده است، دارد. از تبدیل فوریه می توان برای استخراج ویژگی به عنوان ورودی مدل های یادگیری ژرف استفاده کرد، برای مثال می توان تعدادی از مقادیر فرکانسی و دامنه متناظر با آن ها را که از اعمال تبدیل فوریه به یک سیگنال محاسبه شده است به صورت یک بردار ورودی تعریف کرد، همچنین می توان بردار ورودی شامل میانگین یا بیشینه مقادیر دامنه در بازه های فرکانسی مختلف تعریف کرد، علاوه بر این موارد می توان بردای شامل تعداد ثابتی مقادیر فرکانس تعریف کرد که دامنه متناظر با آن ها بیشترین مقدار را دارد.

۵.۳.۱.۱ تبدیل فوریه زمان کوتاه[2]

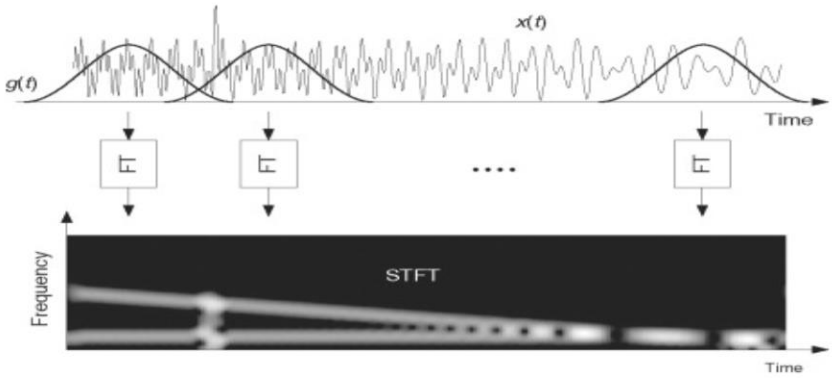
تبدیل فوریه زمان کوتاه با اعمال تبدیل فوریه به بازه های زمانی، پنجره[3]، سیگنال تعریف می شود. با توجه به تعریف پنجره های مجزا در این روش معمولا مقادیر حاصل بین دو پنجره کنار هم دارای ناپیوستگی هستند، برای جلوگیری از این ناپیوستگی در مقادیر حاصل، پیش از اعمال تبدیل فوریه یک تابع پنجره[4] در بخشی سیگنال که داخل پنجره قرار دارد ضرب می شود تا

[1] Fast Fourier transform
[2] Short-time Fourier transform (STFT)
[3] Window
[4] Window function

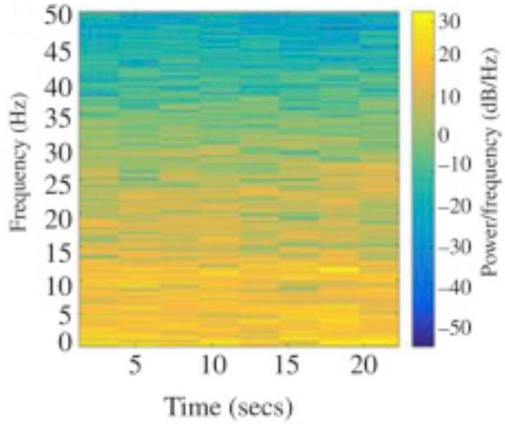
باعث پیوستگی مقادیر خروجی پنجره های مختلف شود، تابع گوسی[1] و تابع هن[2] معمولا بیشترین کاربرد را به عنوان تابع پنجره دارند، مقدار حاصل از ضرب تابع پنجره در قسمت هایی از سیگنال که در خارج از پنجره هستند صفر است. تعریف پنجره هایی که با یکدیگر دارای هم پوشانی زمانی هستند باعث می شود اطلاعات بیشتری از سیگنال استخراج شود به همین دلیل اغلب پنجره های تعریف شده در تبدیل فوریه زمان کوتاه دارای هم پوشانی زمانی هستند که در این صورت استفاده از تابع پنجره و ایجاد پیوستگی بین مقادیر پنجره های کنار هم اهمیت بیشتری پیدا می کند. رابطه تبدیل فوریه زمان کوتاه مطابق رابطه ۵-۵ است. در رابطه ۵-۵، $g_{(.)}$ تابع پنجره و u اندازه پنجره است. خروجی حاصل از تبدیل فوریه زمان کوتاه را می توان به صورت طیف نگاره[3] فرکانسی سیگنال نمایش داد. شکل ۱-۵ روند اعمال تابع پنجره، تبدیل فوریه زمان کوتاه و طیف نگاره فرکانسی حاصل را برای یک سیگنال با یک بعدی نشان می دهد. طیف نگاره فرکانسی سیگنال در یک دستگاه دو بعدی تعریف می شود، یکی از بعدها مربوط به زمان و بعد دیگر مقادیر مختلف فرکانس است، برای هر نقطه تعریف شده در این فضای دو بعدی توان[4] سیگنال در آن نقطه نسبت داده می شود که همان توان دوم مقدار حاصل از رابطه ۵-۵ است. مجموع همه مقادیر توان برابر با انرژی[5] سیگنال است. واحد سیگنال شکل ۲-۵ نمونه ای طیف نگاره تبدیل فوریه زمان کوتاه است که در آن توان سیگنال در نقاط مختلف نمایش داده شده است. از طیف نگاره سیگنال می توان به عنوان ورودی یک مدل یادگیری ژرف استفاده کرد، پردازش این ورودی همانند پردازش ورودی تصویر برای مدل است.

$$STFT_{(u,\omega)} = \int_{-\infty}^{+\infty} f_{(t)} g_{(t-u)} e^{-j\omega t} dt \qquad (۵-۵)$$

[1] Gaussian function
[2] Hann function
[3] Spectrogram
[4] Power
[5] Energy



شکل ۱-۵: روند اعمال تبدیل فوریه زمان کوتاه[۱۴۶]



شکل ۲-۵: طیف نگاره یک سیگنال یک بعدی[۱۴۵]

۵.۳.۲ تبدیل موجک[1]

یکی از معایب تبدیل فوریه زمان کوتاه وابستگی محتوای فرکانسی محاسبه شده از سیگنال به اندازه پنجره اســـت؛ بدین ترتیب در صـــورتی که ابعاد پنجره کوچکتر از دوره[2] مربوط به فرکانس بخشی از سیگنال باشد، آن فرکانس با اعمال پنجره تبدیل فوریه زمان کوتاه شناسائی نمی شـــود. از آن جایی که در تبدیل فوریه زمان کوتاه اندازه پنجره همواره ثلبت اســـت این احتمال وجود دارد که بخشی از فرکانس های پایین سیگنال شناسائی نشوند. تبدیل موجک برای رفع این عیب مربوط به تبدیل فوریه زمان کوتاه ارائه شده است. در تبدیل موجک به جای تابع پنجره از یک تابع موجک مادر[3] اســتفاده می شــود رابطه تبدیل موجک مطابق رابطه ۶-۵

[1] Wavelet transform
[2] Period
[3] Mother wavelet

است. در رابطه ۵-۶، $\psi^*_{(.)}$ مزدوج مختلط تابع موجک مادر $\psi_{(.)}$ ، u اندازه پنجره و s مقیاس[1] موجک است. موجک مادر هر تابعی است که دو شرط قرینگی نسبت مبدا و نرمال بودن را که در روابط ۵-۷ و ۵-۸ نشان داده شده است، داشته باشد. از توابع موجک مادر مرسوم می توان به تابع گوسی و تابع کلاه مکزیکی[2] اشاره کرد.

$$WT_{(u,s)} = \int_{-\infty}^{+\infty} f_{(t)} \frac{1}{\sqrt{s}} \psi^*_{(\frac{t-u}{s})} dt \qquad (۵-۶)$$

$$\int_{-\infty}^{+\infty} \psi_{(t)} dt = 0 \qquad (۵-۷)$$

$$\left\| \psi_{(t)} \right\|^2 = \int_{-\infty}^{+\infty} \psi_{(t)} \psi^*_{(t)} dt = 1 \qquad (۵-۸)$$

با تغییر مقیاس موجک، توابع موجکی موسوم به تابع موجک دختر[3] از تابع موجک مادر اولیه حاصل می شوند که با استفاده از آن ها در رابطه ۵-۶ می توان توان محدوده های فرکانسی مختلف را شناسائی کرده و بدین ترتیب مشکل مربوط به ابعاد ثابت پنجره در تبدیل فوریه زمان کوتاه را رفع کرد. هر چه مقدار مقیاس بزرگ تر باشد فرکانس های پایین تری شناسایی می شوند. خروجی تبدیل موجک را نیز می توان به صورت طیف نگاره ای از ضرایب تبدیل موجک، مقادیر رابطه ۵-۶، ارائه کرده و به عنوان ورودی تصویر در یک مدل یادگیری ژرف استفاده کرد. با توجه به رابطه ۵-۶ و تعریف انتگرال ضرب دو تابع موجک و فرکانس خروجی این رابطه که همان ضرایب موجک است، معیاری برای تعیین میزان شباهت فرکانس با تابع موجک است؛ بدین صورت که هرچه میزان هم پوشانی و شباهت بین دو تابع بیشتر باشد مقدار انتگرال در آن گام، مقدار قدر مطلق، عددی بزرگ تر بوده و نشان دهنده تطابق بیشتر تابع سیگنال با تابع موجک و شامل شدن محدوده فرکانسی تعیین شده توسط تابع موجک در آن بخش از تابع سیگنال می شود.

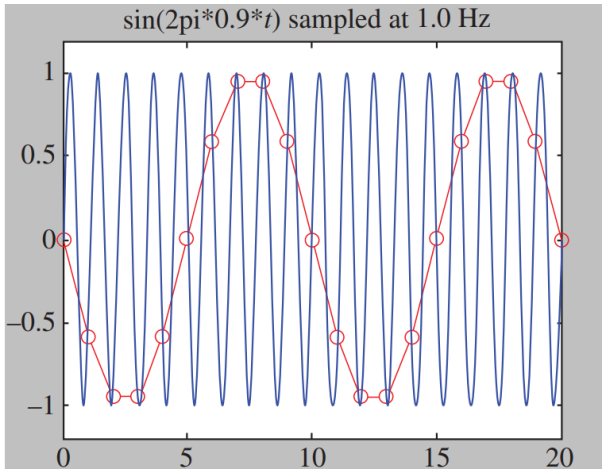
۵.۴ فرکانس نمونه برداری[4] سیگنال

همان طور که در ابتدای فصل اشاره کردیم سیگنال های مورد استفاده در یادگیری ژرف سیگنال گسسته هستند که با نرخ، فرکانس[5]، مشخص از یک سیگنال پیوسته نمونه برداری می

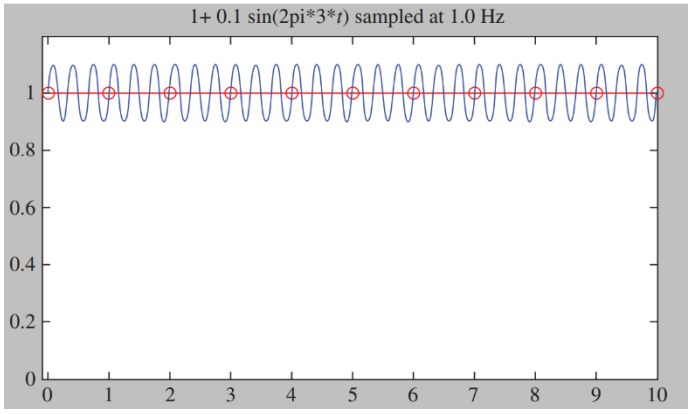
[1] Scale
[2] Mexican hat function
[3] Daughter wavelet
[4] Sampling
[5] Frequency

شـوند. در سـیگنال های حوزه زمان هر چه فرکانس نمونه برداری بیشـتر باشـد تعداد مقادیر گسسته ای، نقاط، که از سیگنال پیوسته که در یک بازه زمانی ثبت می شوند افزایش می یابد در حالی که افزایش فرکانس نمونه برداری در سیگنال های حوزه فضا به صورت افزایش وضوح[1] تصـویر نمایان می شـود. طبق قانون شـنون-نایکوئیسـت[2] فرکانس نمونه برداری باید از دو برابر بیشترین فرکانس سیگنال بیشتر باشد. در صورتی که سیگنال نمونه برداری کمتر از فرکانس سـیگنال باشـد باعث ایجاد پدیده الایزینگ[3] می شـود. در صـورتی که سـیگنال نمونه برداری کوچک تر از دو برابر فرکانس سـیگنال بوده و همچنین فرکانس سـیگنال مضـربی از فرکانس نمونه برداری باشد، پدیده نوسانات پنهان[4] ایجاد می شود. پدیده الایزینگ و نوسانات پنهان به ترتیب در دو شکل ۳-۵ و ۴-۵ با فرض نمونه برداری از یک سیگنال سینوسی نشان داده شده است. پدیده الایزنیگ در تصاویر به صورت پله پله شدن تصویر و پدیده نوسانات پنهان به صورت پیکسل های مشابه کنار هم و عدم ثبت جزئیات پدیدار می شوند. همان طور که در شکل های ۳-۵ تا ۵-۵ نیز مشـاهده می شـود، سـیگنال حاصـل از نمونه برداری با درون یابی خطی بین مقادیر ثبت شـده از نمودار پیوسـته اولیه حاصـل می شـود. برای درون یابی بین این مقادیر می توان از روش های دیگری مانند درون یابی درجه دو و بالاتر و... نیز اسـتفاده کرد. با توجه به توضـیحات فوق می توانیم چنین برداشـت کنیم که طبق قانون شـنون-نایکوئیسـت حداکثر فرکانس قابل شناسائی از مقادیر نمونه برداری شده از یک سیگنال برابر با نصف فرکانس نمونه برداری آن بوده و در صورتی که قانون شنون-نایکوئسیت در نمونه برداری رعایت شده باشد این مقدار حداکثری که از مقادیر شـناسـائی می شـود همان حداکثر مقدار فرکانس موجود در سیگنال اولیه ای تعریف شده برای نمونه برداری است.

[1] Resolution
[2] Shannon-Nyquist theorem
[3] Aliasing
[4] Hidden oscillations

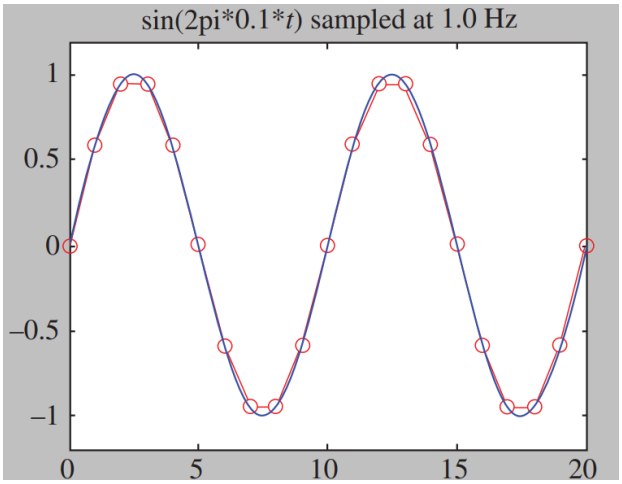


شکل ۳-۵: پدیده الایزنیگ[۱۶۳]



شکل ۴-۵: پدیده نوسانات پنهان[۱۶۳]

شکل ۵-۵ نمایش رعایت قانون شنون-نایکوئیست در نمونه برداری از یک سیگنال سینوسی است.



شکل ۵-۵: نمونه برداری بر اساس قانون شنون-نایکوئیست[۱۶۳]

۵.٤.۱ پدیده تشدید[1] در نمونه برداری

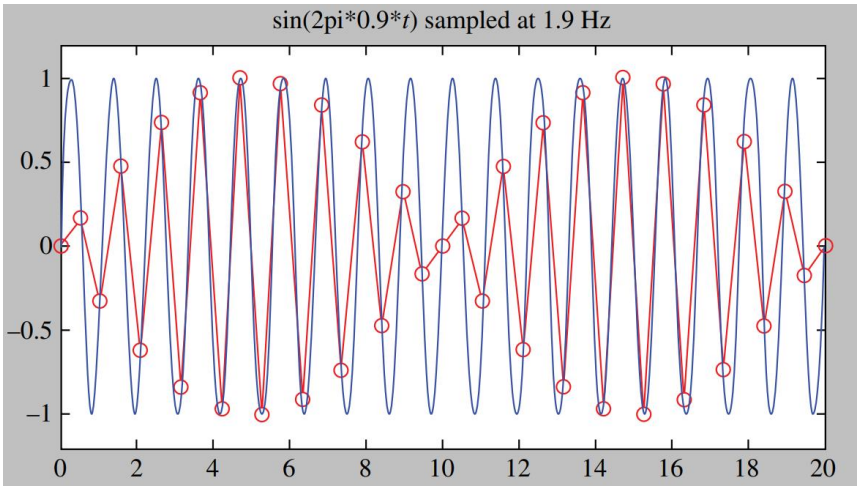
در صورتی که فرکانس نمونه برداری به اندازه بسیار کوچکی از دو برابر فرکانس سیگنال بزرگ تر باشد، پدیده تشدید ایجاد می شود؛ در حالت کلی پدیده تشدید در مواردی که دو سیگنال با فرکانس بسیار نزدیک به هم، با هم جمع می شوند بروز می کند. نحوه ایجاد پدیده تشدید در نمونه برداری مطابق رابطه ۹-۵ اثبات می شود. شکل ۶-۵ نمایش پدیده تشدید در نمونه برداری است. در رابطه ۹-۵، f_s فرکانس سیگنال، f_n فرکانس نمونه برداری، δ یک مقدار فرکانسی کوچک، T دوره سیگنال سینوسی و k متغیر با مقادیر صحیح مثبت است.

$$f_s = \frac{f_n}{2} - \delta$$

$$\sin(2\pi f_s kT) = \sin(2\pi(\frac{f_n}{2} - \delta)kT) = \qquad (۵-۹)$$

$$\sin(2\pi(\frac{f_n}{2})kT)\cos(2\pi\delta kT) - \cos(2\pi(\frac{f_n}{2})kT)\sin(2\pi\delta kT)$$

[1] Beat phenomenon



شکل ۶-۵: پدیده تشدید در نمونه برداری[۱۶۳]

۵.۵ ضرب پیچشی[1]

در مراحل مختلف پردازش یک سیگنال، از فیلترهای فرکانسی مختلفی برای جداسازی محتوای نا مطلوب از سیگنال استفاده می شود. یک فیلتر همانند سیگنال یک تابع است. در فضای فرکانسی اعمال فیلتر به سیگنال با ضرب تابع حوزه فرکانس فیلتر در تابع حوزه فرکانس سیگنال تعریف می شود، در حالی که اعمال فیلتر در حوزه زمان یا فضا معادل تعریف یک ضرب پیچشی بین تابع حوزه زمان یا فضای فیلتر و تابع حوزه زمان یا فضای سیگنال است. رابطه ۱۰-۵، نمایش رابطه بین اعمال فیلتر در دو حوزه زمان و فرکانس برای سیگنال با ورودی یک بعدی، زمان، $f_{(t)}$ است. در رابطه ۱۰-۵، $g_{(t)}$ تابع حوزه زمان فیلتر، $G_{(\omega)}$ و $F_{(\omega)}$ به ترتیب توابع حوزه فرکانس فیلتر و سیگنال و * نماد ضرب پیچشی است.

$$f_{(t)} * g_{(t)} \leftrightarrow F_{(\omega)}.G_{(\omega)} \qquad (۵-۱۰)$$

تبدیل فضای زمان به فرکانس و بالعکس در رابطه ۱۰-۵ به ترتیب با اعمال تبدیل فوریه و معکوس تبدیل فوریه اجرا می شود. ضرب پیچشی بین دو تابع سیگنال و فیلتر در حوزه زمان مطابق رابطه ۱۱-۵ تعریف می شود.

$$f_{(t)} * g_{(t)} = \int_{-\infty}^{+\infty} f_{(\tau)} g_{(t-\tau)} d\tau \qquad (۵-۱۱)$$

[1] Convolutional product

همان طور که پیش تر اشاره کردیم می توان با تبدیل فضای سیگنال به فضای فرکانس، ویژگی هایی را به عنوان ورودی در یک مدل یادگیری ژرف از آن استخراج کنیم. در سیگنال های دو بعدی، تصویر، علاوه بر ویژگی های اشاره شده می توانیم با اعمال فیلتر مشتق گیر[1] لبه های تصویر را که نواحی دارای سیگنال بالا هستند شناسائی کرده و از آن ویژگی هایی مانند HOG[2]و SIFT[3] استخراج کرده و از آن ها به عنوان ورودی مدل استفاده کنیم. اما همواره تبدیل حوزه سیگنال به فرکانس باعث از بین رفتن بخشی از اطلاعات می شود، برای رفع این معضل مدل های ژرف پیچشی ارائه شده اند که در آن ها فیلترهای مختلفی در حوزه تعریف سیگنال، زمان یا فضا، بر روی ورودی اعمال شده و بدین ترتیب اطلاعات ورودی حفظ می شود، حجم محاسبات مربوط به این مدل ها معمولا بیشتر از مدل هایی است که ورودی آن ها ویژگی های حوزه فرکانسی است، اما در عوض دقت عملکردی آن ها در مقایسه با مدل های دارای ورودی ویژگی فرکانسی بسیار بالاتر است.

[1] Derivative filters
[2] Histogram of oriented gradients (HOG)
[3] Scale-invariant feature transform (SIFT)

۵.۶ مسائل

۱- رابطه مربوط به تبدیل فوریه زمان کوتاه معکوس را محاسبه کنید.

۲- رابطه مربوط به تبدیل موجک معکوس را محاسبه کنید

۳- روابط موجک و تبدیل فوریه زمان کوتاه را برای یک سیگنال با ورودی دو بعدی، تصویر، تعریف کنید.

۴- در مورد نحوه محاسبه ویژگی HOG و SIFT در تصاویر تحقیق کنید.

۵- رابطه ۹-۵ مربوط به پدیده تشدید در نمونه برداری را با رابطه پدیده تشدید حاصل از جمع دو سیگنال با فرکانس نزدیک به هم مقایسه کنید.

۶- روابط تبدیل های فوریه معکوس، فوریه زمان کوتاه و موجک را در حالت گسسته بنویسید.

۷- رابطه ضرب پیچشی گسسته را بر اساس رابطه ۱۱-۵ توسعه دهید.

فصل ۶
یادگیری ماشین[1]

۶.۱ مقدمه

یادگیری ماشین یکی از حوزه های مطرح در هوش مصنوعی بوده و در برگیرنده مطالب مربوط به یادگیری ژرف است. هدف کلی مباحث یادگیری ماشین ارائه راهکارهایی برای حل مسائلی چون رگرسیون[2]، طبقه بندی[3] و خوشه بندی[4] و... است. در این فصل ابتدا به بررسی مجموعه دادگان مناسب و عملیات پیش پردازشی که بر روی دادگان اعمال می شود تا برای مدل های یادگیری ماشین قابل استفاده باشند پرداخته و سپس مسئله ذکر شده در حوزه یادگیری ماشین را معرفی می کنیم. در کنار معرفی هر مسئله روابط و مدل های تعریف شده برای حل آن ها را نیز بررسی می کنیم. در انتهای فصل نیز مسائل مربوط به منظور تحقیق بیشتر در مباحث این حوزه معرفی شده است.

۶.۲ پیش پردازش دادگان

مجموعه دادگان مورد استفاده در مدل های پایه یادگیری ژرف معمولا دادگان ساختار یافته ای هستند که در جداولی مانند جداول شکل ۱-۴ ذخیره شده اند. هر یک از ستون های این جداول یک ویژگی[5] مربوط به نمونه های، سطر، مجموعه دادگان هستند. مقادیر ویژگی هایی که در مدل های یادگیری ماشین استفاده می شوند باید به ازای همه نمونه ها تعریف شده باشند، از این رو اولین مرحله پیش پردازش دادگان حذف مقادیر تعریف نشده به یکی از روش هایی است که در ۱-۴-۲ فصل ۴ معرفی شده اند. همچنین همه ویژگی های مجموعه دادگان در مدل های یادگیری ماشین باید کمی باشند، بنابراین در صورتی که یک ویژگی به صورت کیفی تعریف شده باشد به هر یک از حالات آن یک مقدار عددی نسبت داده می شود برای مثال ویژگی رنگ را که شامل سه حالت قرمز، سبز و آبی می باشد را در نظر بگیرید برای اینکه این ویژگی دارای مقادیر کمی باشد

---

[1] Machine learning (ML)
[2] Regression
[3] Classification
[4] Clustering
[5] Feature

در ستون مربوط به آن حالت قرمز را معادل عدد یک، سبز دو و آبی را سه در نظر گرفته و مقادیر عددی را جایگزین مقادیر کیفی می کنیم، بدین ترتیب ویژگی های کیفی تبدیل به ویژگی های کمی می شوند. مقادیر عددی مختلف در مدل های یادگیری ماشین معمولا در بازه اعداد صفر تا یک و یا منفی یک با یک تعریف می شوند لذا باید مقادیر ذخیره شده در جداول مجموعه دادگان را نرمال سازی کرده و آن ها را در بازه اعداد بین صفر تا یک و یا منفی یک تا یک نگاشت کنیم. نرمال سازی مجموعه مقادیر مربوط به یک ویژگی، ستون، بین بازه عددی a تا b مطابق رابطه ۱-۶ تعریف می شود. در رابطه ۱-۶، $\acute{x}$ مقدار متناظر با مقدار x است که در بازه a تا b نرمال سازی شده است، x_{min} و x_{max} نیز به ترتیب کمترین و بیشترین مقدار بین همه مقادیر تعریف شده به ازای نمونه های مختلف برای ویژگی مورد نظر هستند.

$$x' = (b-a)\frac{x - x_{\min}}{x_{\max} - x_{\min}} + a \qquad (۶-۱)$$

۶.۳ تبدیل و انتخاب ویژگی[1]

به طور کلی می توان مدل های مطرح شده در حوزه یادگیری ماشین را به مدل های دارای آموزش با سرپرست[2] و مدل های دارای آموزش بدون سرپرست[3] دسته بندی کرد. منظور از آموزش تنظیم مقادیر پارامترهای مختلف مدل به منظور بهینه کردن خطای مدل است. در مدل هایی که دارای آموزش بدون سرپرست هستند تعدادی از ویژگی های مجموعه دادگان به عنوان ورودی مدل تعریف شده و خروجی مدل بر اساس مقادیر متناظر با ویژگی های ورودی به ازای نمونه های مختلف محاسبه می شود، در این مدل ها مقدار مطلوبی برای خروجی تعریف نشده و تنظیم پارامترها صرفا بر اساس مقادیر ویژگی های ورودی به ازای نمونه های مختلف تعریف شده در مجموعه دادگان انجام می شود. در مقابل، برای مدل های دارای آموزش با سرپرست علاوه بر ورودی مقدار مطلوب[4] خروجی نیز تعریف می شود؛ بنابراین در مدل های دارای آموزش با سرپرست هدف آموزش مدل به گونه ای است که در آن مدل به ازای مقادیر ورودی هر نمونه، مقدار مطلوب تعریف شده به ازای آن نمونه را تولید کند. مقدار مطلوب همانند ویژگی های ورودی یک یا چند ستون از جدول مجموعه است.

بنابر توضیحات فوق الذکر انتخاب ویژگی های مناسب به منظور بهبود عملکرد مدل برای مسئله تعریف شده از بین همه ستون هایی، ویژگی ها، که امکان تعریف به عنوان ورودی را دارند، یکی از مهم ترین مراحل پیش پردازش در مدل های یادگیری ماشین محسوب می شود. در هر دو روش

[1] Feature selection
[2] Supervised learning
[3] Unsupervised learning
[4] Desired value - Label

آموزش با سرپرست و بدون سرپرست برای عملکرد مناسب مدل حائز اهمیت است. روش های انتخاب ویژگی نیز همانند روش های آموزش به دو دسته کلی روش های با سرپرست و بدون سرپرست تقسیم می شوند. روش های انتخاب ویژگی بدون سرپرست روش هایی هستند که در آن ها الگوریتم و معیار انتخاب ویژگی مستقل از عملکرد مدل است. در این مدل ها صرفا مقایسه ای بین ویژگی های مختلف تعریف شده صورت گرفته و از بین آن ها بهترین ویژگی ها بر اساس معیار الگوریتم، به عنوان ورودی مدل انتخاب می شوند. روش تحلیل مولفه های اصلی[1] یکی از این روش های انتخاب ویژگی است که در ادامه به بررسی آن خواهیم پرداخت. در مقابل این روش ها، روش های انتخاب ویژگی با سرپرست تعریف می شوند که در آن ها عملکرد مدل نیز در روند انتخاب ویژگی ها دخیل است. روش های انتخاب ویژگی با سرپرست به سه دسته کلی فیلتر[2]، بسته ای[3] و جا نمائی شده[4] تقسیم می شوند که در ادامه هر کدام را معرفی می کنیم:

فیلتر: روش هایی هستند که تعدادی از ویژگی ها را با توجه به رابطه آن ها با مقدار مطلوب مدل انتخاب می کنند، انتخاب ویژگی هایی دارای بیشترین اطلاعات متقابل[5] با مقدار مطلوب خروجی هستند یک روش انتخاب ویژگی فیلتر محسوب می شود.

بسته ای: در این روش انتخاب ویژگی مدل های مختلفی که هر کدام دارای دسته ای از ویژگی های تعریف شده هستند تعریف شده و از بین آن ها ویژگی های مدلی که عملکرد بهتری دارد به عنوان ویژگی های انتخابی در نظر گرفته می شوند. این روش در واقع یک روش جستجو بین ویژگی های مختلف محسوب می شود. الگوریتم حذف ویژگی بازگشتی[6] یکی از این روش های انتخاب ویژگی است.

جا نمائی: این روش انتخاب ویژگی به عنوان بخشی از مدل تعریف می شود و در طول روند آموزش مدل ویژگی های مناسب را از بین ویژگی های مختلف انتخاب می کند. درخت های تصمیم[7] یکی از مدل های که دارای این نوع روش انتخاب ویژگی هستند که بررسی آن ها خارج از مطالب این کتاب است. از ترتیب قرارگیری ویژگی ها در یک درخت تصمیم می توان اهمیت ویژگی های مختلف را محاسبه و از آن به عنوان یک روش انتخاب ویژگی برای سایر مدل ها استفاده کرد.

از روش های انتخاب ویژگی به طور عمده به عنوان یک راهکار برای کاهش ابعاد[8] ورودی استفاده می شود تا حجم محاسبات مدل که یکی از چالش های عمده در مدل های یادگیری ماشین و یادگیری ژرف است کاهش یابد.

---

[1] Principle components analysis (PCA)
[2] Filter
[3] Wrapper
[4] Embedded
[5] Mutual information
[6] Recursive feature elimination (RFE)
[7] Decision trees
[8] Dimension reduction

۶.۳.۱ تحلیل مولفه های اصلی

روش تحلیل مولفه های اصلی یک روش تبدیل ویژگی خطی است که از آن برای انتخاب ویژگی به صورت بدون سرپرست استفاده می شود. در این روش ویژگی هایی که دارای وابستگی خطی به سایر ویژگی هستند به عنوان ویژگی زائد تشخیص داده شده و از مجموعه ویژگی های تعریف شده حذف می شوند. بدین ترتیب علاوه بر کاهش ابعاد، فقط ویژگی هایی مستقل به عنوان ورودی انتخاب می شوند این ویژگی ها همان محورهایی هستند که فضای ویژگی را افراز[1] می کنند. برای محاسبه میزان استقلال و یا وابستگی بین ویژگی ها لازم است مقادیر ویژه مربوط به ماتریس کواریانس[2] ویژگی ها محاسبه شود. ویژگی هایی مستقل هستند که مقدار ویژه متناظر با آن ها بزرگ تر باشد. ماتریس کواریانس مجموعه دادگان مطابق رابطه ۲-۶ محاسبه می شود. در رابطه ۲-۶، m تعداد نمونه های مجموعه دادگان، n تعداد کل ویژگی های مجموعه دادگان، $Data$ و $\sigma_{(x_a,x_b)}$ کواریانس بین دو ویژگی، ستون، x_a و x_b است.

$$Cov(Data_{m\times n}) = \begin{bmatrix} \sigma_{(x_1,x_1)} & \sigma_{(x_1,x_2)} & \cdots & \sigma_{(x_1,x_n)} \\ \sigma_{(x_2,x_1)} & \sigma_{(x_2,x_2)} & \cdots & \vdots \\ \vdots & \vdots & \ddots & \vdots \\ \sigma_{(x_n,x_1)} & \cdots & \cdots & \sigma_{(x_n,x_n)} \end{bmatrix}_{n\times n} \qquad (۶-۲)$$

کواریانس بین دو ویژگی، ستون، x_a و x_b مطابق رابطه ۳-۶ محاسبه می شود. در رابطه ۳-۶، x_{a_i} مقادیر مختلف ویژگی x_a به ازای نمونه های مجموعه دادگان و x_{b_i} مقادیر مختلف ویژگی x_b به ازای نمونه های مجموعه دادگان است.

$$\sigma_{(x_a,x_b)} = \frac{1}{m-1}\sum_{i=1}^{m}(x_{a_i} - \overline{x}_a).(x_{b_i} - \overline{x}_b) \qquad (۶-۳)$$

پس از محاسبه ماتریس کواریانس رابطه ۲-۶ با استفاده از رابطه ۲۳-۲ مقادیر ویژه ماتریس کواریانس را محاسبه می کنیم. مقادیر ویژه ماتریس کواریانس شامل n مقدار اسکالر متناظر با هر یک از ویژگی های تعریف شده در مجموعه دادگان است. با توجه به اینکه مقادیر ویژه حاصل از رابطه ۲۳-۲ در یک ماتریس قطری $n \times n$ تعریف می شوند، هر یک از درایه های قطر اصلی این ماتریس، مقادیر ویژه متناظر با ویژگی است که واریانس آن در درایه معادل با آن بر روی قطر اصلی ماتریس کواریانس رابطه ۲-۶ تعریف شده است. ویژگی هایی که مقادیر ویژه بزرگ تری دارند از استقلال خطی بیشتری برخوردار هستند.

[1] Span
[2] Convivence matrix

۶.۳.۲ تحلیل مولفه های اصلی مبتنی بر کرنل[1]

روش تحلیل مولفه های اصلی مبتنی بر کرنل برای ویژگی های استخراج شده حاصل از اعمال روش تحلیل مولفه های اصلی به مجموعه دادگان به کار برده می شود. بدین صورت که پس از اعمال روش تحلیل مولفه های اصلی با استفاده از ویژگی های استخراج شده، ویژگی های جدیدی با اعمال یک کرنل به آن ها اعمال می شود. این ویژگی حاصل در کنار ویژگی هایی که به عنوان ویژگی مستقل از طریق روش تحلیل مولفه های اصلی استخراج شده اند، ویژگی های ورودی مدل را تشکیل می دهند. افزودن ویژگی همانند نگاشت فضای ویژگی ها به فضائی با ابعاد بالاتر است. تابع گوسی یکی از مرسوم ترین توابعی است که به عنوان تابع کرنل استفاده می شود.

۶.۴ مسائل مطرح در یادگیری ماشین

در این بخش به تعریف مسئله های رگرسیون، طبقه بندی و خوشه بندی که از مسائل مهم مطرح در حوزه یادگیری ماشین هستند پرداخته سپس مدل های ارائه شده برای هر کدام را بررسی می کنیم.

۶.۴.۱ رگرسیون

هدف از تعریف مسئله رگرسیون برازش منحنی[2] بر نمونه های مجموعه دادگان و ارائه رابطه ای برای درون یابی[3] و برون یابی[4] بر اساس آن نمونه ها است. برای تعریف مسئله رگرسیون، باید مقادیر ورودی و مقادیر مطلوب خروجی در نمونه های مختلف مجموعه دادگان مشخص شده و برای هر خروجی رابطه کلی مانند مطابق رابطه ۴-۶ توسعه داده شود. در این رابطه مقادیر $a, b, c, \dots, d$ پارامترهای اسکالر مدل هستند که مقادیر آن ها با آموزش مدل تنظیم می شود، x_i مقدار ویژگی، بعد، ورودی i-ام و $\hat{y}$ خروجی متناظر به ازای مقادیر ورودی است توسط مدل، رابطه، رگرسیون ایجاد می شود. با توجه به تعریف ویژگی ورودی و مقدار مطلوب آموزش رابطه رگرسیون یک آموزش با سرپرست محسوب می شود.

$$\hat{y}_{(\mathbf{x})} = ax_1 + bx_2 + \dots + cx_n + d \qquad (۶-۴)$$

در طی روند آموزش پارامترهای رابطه، مدل، رگرسیون به گونه ای تنظیم می شوند تا مقدار خروجی محاسبه شده توسط رابطه ۴-۶ بیشترین شباهت را با مقدار خروجی مطلوب به ازای

[1] Kernel-PCA
[2] Curve fitting
[3] Interpolation
[4] Extrapolation

مقادیر ورودی داشته باشد؛ به بیان دیگر مقدار احتمال شرطی خروجی رابطه ۴-۶ نسبت به ورودی و مقادیر پارامترها به ازای همه نمونه های مجموعه دادگان بیشینه شود، رابطه این احتمال شرطی مطابق رابطه ۵-۶ است. در رابطه ۵-۶، w مجموعه پارامترهای مدل رگرسیون و $\hat{y}_{i(.)}$ خروجی متناظر با بردار ورودی $\boldsymbol{x}_i$ در نمونه i-ام مجموعه دادگان متشکل از m نمونه است که توسط مدل رگرسیون محاسبه شده است. بردار ورودی $\boldsymbol{x}_i$ شامل مقادیر ورودی ویژگی های مختلف، x_{1_i} تا x_{n_i}، در نمونه i-ام است.

$$\arg\max(\prod_{i=1}^{m} P_{(\hat{y}_i|w,\mathbf{x}_i)}) \qquad (۶-۵)$$

برای مقادیر تعریف شده در مجموعه دادگان، آغشته بودن به نویز با توزیع احتمالاتی گوسی با میانگین صفر و واریانس σ را در نظر می گیریم با این فرض رابطه ۵-۶ معادل کمینه کردن فاصله اقلیدسی بین مقدار خروجی مدل رابطه ۴-۶ و مقدار خروجی مطلوب متناظر با آن به ازای نمونه های مختلف است. بدین ترتیب رابطه ۵-۶ را به صورت رابطه ۶-۶ توسعه می دهیم. در رابطه ۶-۶، $\hat{y}_i$ خروجی مدل، رابطه، رگرسیون به ازای ورودی های نمونه i-ام و y_i مقدار مطلوب نمونه i-ام است.

$$\arg\min(\sum_{i=1}^{m}(y_i-\hat{y}_i)^2) \qquad (۶-۶)$$

برای کمینه کردن رابطه ۶-۶ مشتق آن را نسبت به هر یک از پارامترهای مجموعه w برابر با صفر قرار داده بدین ترتیب مقدار بهینه پارامترهای رابطه رگرسیون را محاسبه می کنیم. برای سهولت محاسبه مقادیر مختلف پارامتر ها می توانیم رابطه ۴-۶ را به صورت ماتریس مطابق رابطه ۷-۶ تعریف کرده و مقادیر خروجی مدل را به ازای همه نمونه ها با تعریف یک ضرب ماتریسی مطابق رابطه ۷-۶ محاسبه می کنیم. در رابطه ۷-۶، W ماتریسی است که شامل همه پارامترهای $a, b, c, \dots, d$، X ماتریس نمونه های ورودی است که در آن ستون ها ویژگی های هر نمونه است.

$$\hat{Y}_{m\times 1} = X_{m\times n+1} \times W^{T}_{1\times n+1} = \begin{bmatrix} x_{1_1} & x_{2_1} & \cdots & x_{n_1} & 1 \\ x_{1_2} & x_{2_2} & \cdots & x_{n_2} & 1 \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ x_{1_m} & x_{1_m} & \cdots & x_{n_m} & 1 \end{bmatrix}_{m\times n+1} \times \begin{bmatrix} a \\ b \\ c \\ \vdots \\ d \end{bmatrix}_{n+1\times 1} \qquad (۶-۷)$$

در ماتریس ورودی رابطه ۶-۷، یک ستون با مقادیر یک به ویژگی های نمونه ها اضافه شده است این ماتریس برای ضرب در پارامتر d در نظر گرفته شده است، همان طور که در رابطه ۶-۴ هم نشان داده شده است در پارامتر d هیچ یک از مقادیر ویژگی ها ضرب نمی شود، به این پارامتر، بایاس[1] اطلاق می شود که نقشی مانند عرض از مبدا در معادله یک خط را دارد. معادل رابطه ۶-۶ برای رابطه ۶-۷، رابطه ۶-۸ خواهد بود. در رابطه ۶-۸، Y ماتریس است که در آن مقادیر مطلوب همه نمونه های مجموعه دادگان تعریف شده است.

$$(Y-\hat{Y})^2 \qquad (۶-۸)$$

برای محاسبه مقادیر بهینه ماتریس W از ماتریس حاصل از رابطه ۶-۸ نسبت به ماتریس W مشتق گرفته و آن را برابر با صفر قرار می دهیم. بدین ترتیب ماتریس مقادیر بهینه W مطابق رابطه ۶-۹ محاسبه می شود.

$$W=(X^T X)^{-1} X^T Y \qquad (۶-۹)$$

رابطه ۶-۴ که رابطه کلی یک مدل رگرسیون است در واقع یک نگاشت خطی بین مقادیر ورودی خروجی تعریف می کند از این رو به آن رگرسیون خطی اطلاق می شود. در صورتی که بین ویژگی های ورودی و خروجی در یک مجموعه دادگان رابطه خطی برقرار نباشد این رابطه عملکرد مطلوبی نداشته و مقدار خطای کمینه آن، رابطه ۶-۶، به ازای نمونه های مجموعه دادگان مقدار بزرگی خواهد بود. برای رفع این معضل به رابطه ۶-۴ یک تابع غیرخطی مطابق رابطه ۶-۱۰ اضافه می شود تا نگاشت غیر خطی بین ورودی و خروجی در توسعه رابطه رگرسیون لحاظ شود. انتخاب تابع غیرخطی مناسب از چالش های این مدل ها است. توابع تانژانت هیپربولیک، سیگموئید و تابع گوسی از مرسوم ترین توابعی هستند که به عنوان تابع غیرخطی در مدل های رگرسیون استفاده می شوند. در رابطه ۶-۱۰، $f_{(.)}$ تابع غیرخطی است.

$$\hat{y}_{(\mathbf{x})} = f(ax_1 + bx_2 + \ldots + cx_n + d) \qquad (۶-۱۰)$$

رابطه ۶-۱۰ یک مدل رگرسیون غیرخطی است، علاوه بر رابطه ۶-۱۰ می توان مدل رگرسیون غیرخطی را به صورت رابطه ۶-۱۱ نیز تعریف کرد. در رابطه ۶-۱۱، $f_{(.)}$، $g_{(.)}$ و $h_{(.)}$ توابع غیرخطی هستند.

$$\hat{y}_{(\mathbf{x})} = af(x_1) + bg(x_2) + \ldots + ch(x_{n-1}, x_{n-1}) + \ldots + d \qquad (۶-۱۱)$$

تعریف یک مدل شامل ترکیبات چند جمله ای درجه m از ویژگی های ورودی یک مدل رگرسیون غیرخطی مطابق رابطه ۶-۱۲ مثالی از رابطه ۶-۱۱ است، در این مدل ها انتخاب تعداد و

[1] Bias

درجه جملات به طوری که مدل رگرسیون بهترین عملکرد را داشته باشند، عمده چالش مطرح در این مدل ها است. در رابطه ۱۲-۶، $a, b, c, d, k, u, \ldots$ پارامترهای اسکالر مدل هستند.

$$\hat{y}_{(\mathbf{x})} = ax_1^{\ m} + bx_2^{\ m} + \ldots + cx_n^{\ m} + kx_1^{\ m-1}.x_2^{\ 1} + \ldots + ux_{n-1}^{\ 1}.x_n^{\ m-1} + d \quad (۶-۱۲)$$

پس از آموزش و محاسبه مقادیر بهینه پارامترهای مدل رگرسیون در نهایت مدلی، رابطه ای، حاصل می شود که با استفاده از آن می توان مقدار متغیر کمی پیوسته y متناظر با هر بردار ورودی $\boldsymbol{x}$، چه ورودی هایی که مقادیر آن ها در مجموعه دادگان تعریف شده و چه ورودی هایی که به عنوان نمونه در مجموعه دادگان تعریف نشده اند را محاسبه کنیم. عملکرد این مدل برای ورودی هایی که مقادیر آن ها در بازه بین مقادیر نمونه های آموزشی است، درون یابی، معمولا مطلوب است ولی به ازای ورودی های خارج از بازه عددی نمونه های آموزشی، به خصوص مقادیری که فاصله نسبتا زیادی با بازه نمونه های آموزشی دارند، در مواردی عملکرد مدل چندان مطلوب نیست. یکی از مثال های کاربردی برای مسئله رگرسیون توسعه یک مدل رگرسیون برای محاسبه قیمت خانه با ورودی هایی مانند متراژ، تعداد اتاق و... است، برای آموزش چنین مدلی مجموعه دادگان متشکل از تعدادی خانه که قیمت، متراژ، تعداد اتاق و... هر یک از آن ها مشخص شده باشد مورد نیاز است.

۶.۴.۲ طبقه بندی

هدف مسئله رگرسیون ارائه یک رابطه کلی برای محاسبه مقدار یک ویژگی کمی پیوسته به ازای مقادیر مختلف ورودی است در حالی که در بعضی از متغیرها در مجموعه دادگان به صورت متغیرهای کیفی تعریف می شوند و مدل رگرسیون قابلیت محاسبه حالت های مختلف تعریف شده برای متغیر کیفی را ندارد. هدف از طرح مسئله طبقه بندی ارائه مدلی برای محاسبه حالات مختلف تعریف شده برای یک متغیر کیفی با توجه به مقادیر ورودی است. برای آموزش مدل طبقه بند[1] نیز مستلزم مجموعه دادگانی متشکل از نمونه هایی است که در آن ها مقادیر ورودی و مقدار مطلوب تعریف شود، با این اوصاف آموزش مدل طبقه بند نیز آموزش با سرپرست محسوب می شود. مقادیر مطلوب برای دادگان مدل طبقه بند همان حالت های متغیر کیفی است. رابطه ۱۳-۶ حالت کلی یک مدل طبقه بند را برای متغیر y که شامل دو حالت، کلاس، صفر و یک است نشان می دهد. در رابطه ۱۳-۶ مقادیر $a, b, c, \ldots, d$ پارامترهای اسکالر مدل هستند که مقادیر آن ها با آموزش مدل تنظیم می شود، x_i مقدار ویژگی، بعد، ورودی i-ام و $\hat{y}$ خروجی متناظر به ازای مقادیر ورودی است.

[1] Classifier

$$\hat{y}_{(\mathbf{x})} = \begin{cases} 1, & if \quad 0 \le ax_1 + bx_2 + ... + cx_n + d \\ 0, & if \quad 0 > ax_1 + bx_2 + ... + cx_n + d \end{cases} \qquad (۶-۱۳)$$

برای آموزش پارامترهای یک مدل طبقه بند از معیار آنتروپی متقاطع[1] استفاده می شود که در فصل ۸ آن را بررسی خواهیم کرد.

۶.۴.۲.۱ مدل طبقه بند برای بیش از دو کلاس[2]

مدل طبقه بند رابطه ۱۳-۶ برای حالتی کاربرد دارد که برای متغیر خروجی صرفا دو کلاس[3]، حالت، صفر و یک تعریف شده باشد در صورتی که ممکن است برای یک متغیر بیش از دو حالت تعریف شده باشد، برای مواجهه با این مشکل در مدل های طبقه بند دو رویکرد کلی به شرح ذیل ارائه می شود:

- **یک کلاس در برابر بقیه[4]:** در این روش هر بار یکی از حالت ها، کلاس ها را معادل کلاس یک و سایر حالت ها را معادل کلاس صفر در نظر می گیرند، در این حالت برای تشخیص کلاس متغیر خروجی صورت $n-1$ مدل طبقه بند مجزا برای n کلاس تعریف شده باید آموزش داده شود. به وجود آمدن نواحی در فضای ویژگی های ورودی که متعلق به هیچ یک از کلاس ها نیست از معایب و حجم محاسبات کمتر نسبت به روش بعدی از مزایای این روش است.
- **یک کلاس در برابر یک کلاس[5]:** در این روش برای آموزش هر مدل طبقه بند از بین کلاس های تعریف شده دو کلاس را انتخاب و یک مدل طبقه بند بین آن دو تعریف می کنیم. در این روش لازم تا برای n کلاس تعریف شده $\binom{n}{2}$ مدل طبقه بند آموزش داده شود. حجم محاسبات بیشتر در مقایسه با روش قبل از معایب و تشکیل نواحی مجهول محدودتر نسبت به روش قبل از مزایای آن است.
- **استفاده از مقدار حاصل از طبقه بند به جای استفاده از علامت آن:** مطابق رابطه ۱۳-۶ صرفا علامت خروجی رابطه برای تعیین کلاس مقادیر ورودی استفاده می شود، این رویکرد که در دو روش پیشین استفاده شده است باعث ایجاد نواحی مجهول می شود برای رفع معصل برای هر کلاس یک رابطه طبقه بند خطی مطابق رابطه ۱۴-۶ نسبت داده می شود. در رابطه ۱۴-۶، $a, b, c, ..., d$ پارامترهای مدل، x_i مقدار ویژگی ورودی i-ام، n تعداد کلاس ها، m تعداد بعد، ویژگی ورودی است.

[1] Cross entropy
[2] Categorical classification
[3] Binary classification
[4] One versus the rest
[5] One versus one

$$y_{j(\mathbf{x})} = ax_1 + bx_2 + \ldots + cx_m + d \quad (۶-۱۴)$$

در این روش یک نمونه به ازای مقادیر ورودی آن متعلق به کلاس k است اگر رابطه ۶-۱۵ برای آن برقرار باشد:

$$y_{k(\mathbf{x})} > y_{j(\mathbf{x})}, j = 1,\ldots,n, j \neq k \quad (۶-۱۵)$$

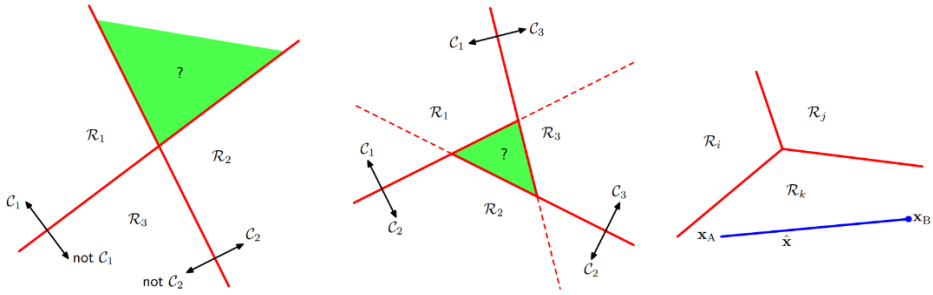
پارامترهای هر مدل طبقه بند با حل معادله رابطه ۶-۱۶ به ازای نمونه های ورودی مختلف آموزش داده می شوند، رابطه ۶-۱۶ همان مرز بین دو کلاس است.

$$y_{k(\mathbf{x})} = y_{j(\mathbf{x})}, j = 1,\ldots,n, j \neq k$$

$$y_{k(\mathbf{x})} - y_{j(\mathbf{x})} = 0$$

$$(a_k x_1 + b_k x_2 + \ldots + c_k x_m + d_k) - (a_j x_1 + b_j x_2 + \ldots + c_j x_m + d_j) = 0 \quad (۶-۱۶)$$

شکل ۶-۱ مقایسه سه روش طبقه بندی فوق الذکر است. در شکل ۶-۱، C_i کلاس i-ام و R_i ناحیه مربوط به کلاس i-ام است. همان طور که در شکل ۶-۱ ج نیز نشان داده شده است، در یک طبقه بند خطی در صورتی که دو نقطه x_A و x_B متعلق به کلاس i باشند، هر نقطه بر روی خط متصل کننده آن ها، $\hat{x}$، نیز متعلق به کلاس i است.



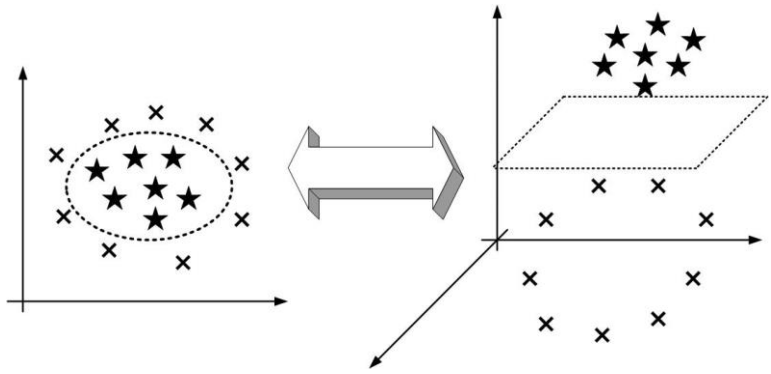
ج) استفاده از مقدار حاصل از رابطه ب) یک کلاس در برابر یک کلاس الف) یک کلاس در برابر بقیه

شکل ۶-۱: مقایسه راهکارهای ارائه شده برای مدل طبقه بند دارای بیش از یک کلاس[۱]

۶.۴.۲.۲ مدل طبقه بند غیرخطی

مدل طبقه بند رابطه ۶-۱۳ و مدل های شکل ۶-۱ طبقه بندهای خطی هستند که یک مرز خطی به عنوان جدا کننده بین کلاس های مختلف تعریف می کنند، این مدل ها در صورتی که مجموعه دادگان بر اساس ویژگی های ورودی به صورت خطی تفکیک پذیر باشند عملکرد مناسبی دارند، اما در مواردی مجموعه دادگان به صورت خطی جدائی پذیر نیستند. در چنین مواردی از راهکاری مشابه با راهکار مطرح شده در روش تحلیل مولفه های اصلی مبتنی بر کرنل استفاده می شود؛ بدین صورت که با تعریف یک تابع کرنل یک بعد به ابعاد، ویژگی ها، ورودی اضافه کرده و فضای ویژگی را به ابعاد بالاتر نگاشت می کنیم، در این فضای نگاشت شده نمونه های مجموعه

دادگان به صورت خطی تفکیک پذیر هستند بنابراین می توانیم در این فضا یک مدل طبقه بند خطی مشابه رابطه ۱۳-۶ برای آن ها تعریف کنیم. نگاشت طبقه بند خطی در فضای با بعد بالاتر بر روی فضای اصلی، بعد پایین تر، یک طبقه بند غیر خطی است که در شکل ۲-۶ رابطه بین این دو فضا نمایش داده شده است. این روش به طور گسترده در طبقه بندهایی مانند ماشین های بردار پشتیبان[1] به کار گرفته می شود که بررسی آن ها خارج از مطالب این کتاب است. انتخاب تابع کرنل مناسب در این مدل ها که امکان تفکیک پذیری در فضای حاصل را برای دادگان فراهم کند همواره یکی از چالش های مطرح در این مدل های طبقه بند است.



شکل ۲-۶: انتقال فضای ویژگی به ابعاد بالاتر برای ایجاد تفکیک پذیری

۶.۴.۲.۳ رگرسیون لجستیک[2]

رگرسیون لجستیک یکی از مدل های طبقه بندهای غیرخطی مرسوم است. رابطه کلی این طبقه بند مطابق رابطه ۱۷-۶ تعریف می شود. در رابطه ۱۷-۶، $a, b, c, \dots, d$ پارامترهای مدل، x_i مقدار ویژگی، بعد، ورودی i-ام و P_j احتمال تعلق نمونه دارای بردار ورودی $\boldsymbol{x}$ به کلاس j است که مقداری بین صفر تا یک دارد.

$$P_{j(\mathbf{x})} = \frac{e^{ax_1+bx_2+\dots+cx_n+d}}{1+e^{ax_1+bx_2+\dots+cx_n+d}} \qquad (۶-۱۷)$$

در صورتی که فضای تعریف مسئله شامل دو کلاس j, k ناسازگار باشد می توانیم برای آن صرفا یک مدل رگرسیون لجستیک مطابق رابطه ۱۷-۶ با تعریف مقادیر مطلوب صفر و یک برای آن آموزش دهیم؛ در این صورت مقدار یک مقدار مطلوب تعلق به کلاس j و مقدار صفر مقدار مطلوب تعلق به کلاس k است. در این حالت پس از آموزش مدل مقدار حاصل از آن همواره عددی بین صفر تا یک خواهد بود که با اعمال حدآستانه، معمولا مقدار ۰.۵، می توانیم این مقدار خروجی را به

[1] Support vector machines (SVMs)
[2] Logistic regression

یکی از دو کلاس j,k نسبت دهیم. برای حالتی که مسئله شامل بیش از دو کلاس ناسازگار باشد به تعداد کلاس ها یک مدل رگرسیون لجستیک آموزش داده و از بین آن ها مدل کلاسس که دارای مقدار حداکثری است را به عنوان تعلق به آن کلاس در نظر می گیریم، این روند همان رویکرد سوم در مواجهه با چند کلاس است که در آن به جای علامت از مقدار مدل طبقه بند استفاده می شود. مقادیر مطلوب برای حالت بیش از دو کلاس به صورت یک بردار با اندازه ای برابر با تعداد کلاس ها تعریف می شود که شامل ابعادی با مقادیر صفر یا یک است، به این بردار مقادیر مطلوب بردار One-Hot اطلاق می شود. رابطه ۱۸-۶ یک بردار One-Hot است که برای نمونه j-ام مجموعه دادگان مربوط به یک مسئله طبقه بندی شامل سه کلاس تعریف شده است، این بردار یک مقدار مطلوب، برچسب، تعلق به کلاس دوم از بین سه کلاس است، ابعاد مختلف این بردار به ترتیب کلاس ها تعریف می شود؛ بدین معنی که بعد i-ام بردار One-Hot مشخص کننده تعلق یا عدم تعلق به کلاس i تعریف شده برای مدل است.

$$OneHot_j = [0,1,0] \qquad (۶-۱۸)$$

۶.۴.۲.۴ طبقه بند بیز ساده[1]

طبقه بند بیز ساده یکی دیگر از مدل های طبقه بند محسوب می شود. رابطه کلی این این طبقه بند مطابق رابطه ۱۹-۶ است. در رابطه ۱۹-۶، $P_{(y_j|\mathbf{x})}$ احتمال تعلق نمونه دارای بردار ورودی $\boldsymbol{x}$ به کلاس y_j که به آن احتمال پسین[2]، $P_{(\mathbf{x})}$ احتمال بردار ورودی $\boldsymbol{x}$ که به آن احتمال شاهد[3]، $P_{(y_j)}$ احتمال وقوع کلاس y_j که به آن احتمال پیشین[4] و $P_{(\mathbf{x}|y_j)}$ احتمال شرطی بردار ورودی $\boldsymbol{x}$ به ازای کلاس y_j است که به آن درست نمائی[5] گفته می شود.

$$P_{(y_j|\mathbf{x})} = \frac{P_{(y_j)}P_{(\mathbf{x}|y_j)}}{P_{(\mathbf{x})}} \qquad (۶-۱۹)$$

نحوه نسبت دادن یک نمونه به یک کلاس بر اساس مقدار خروجی در این مدل مشابه مدل رگرسیون لجستیک است، در صورتی که توزیع های احتمالاتی به کار برده شده در مدل بیز ساده پارامتریک باشند می توان برای آموزش پارامترهای آن ها از مقادیر مطلوبی مشابه مدل رگرسیون لجستیک استفاده کرد. مدل سازی توزیع احتمالاتی شاهد که همان مخرج رابطه ۱۹-۶ است با توجه به وابستگی آن به فضای ویژگی و عدم در دسترس بودن همه نمونه های قابل تعریف در این

[1] Naïve Bayes classifier
[2] Posterior probability
[3] Evidence probability
[4] Prior probability
[5] Likelihood

فضا همواره یکی از چالش های این مدل است.

از مدل های طبقه بندی که دارای خروجی احتمالی هستند، مانند طبقه بند رگرسیون لجستیک و بیز ساده می توان برای حالتی که فقط یک کلاس تعریف شده باشد نیز استفاده کرد؛ در این مسائل برای عدم تحقق کلاس مورد نظر، کلاس خاصی تعریف نشده و فقط شرایط یک کلاس مورد بررسی قرار می گیرد. همچنین با تعریف چند مدل با خروجی احتمالی به صورت موازی در کنار هم می توان مدلی برای مسئله طبقه بندی دارای کلاس های سازگار که امکان تحقق همزمان کلاس ها در آن ها تعریف شده است، نیز توسعه داد.

۶.۴.۳ مدل خود همبسته[1]

یکی دیگر از مسائل مطرح در یادگیری ماشین مسئله خود همبستگی[2] است. هدف از این مسئله پیش بینی مقدار یک متغیر وابسته در یک دینامیک با استفاده مقادیر دینامیک های قبلی آن است. مدل تعریف شده برای این مسئله مشابه مدل رگرسیون است با این تفاوت که در مدل خود همبسته، ورودی دینامیک های قبل از دینامیک زمانی خروجی است. در این مدل مقدار مطلوب نیز برابر مقدار دینامیک خروجی است. رابطه کلی یک مدل خود همبسته مطابق رابطه ۶-۲۰ است. از مدل خود همبسته برای پیش بینی مقادیر سری های زمانی، سیگنال های زمانی، مختلف مانند پیش بینی وضعیت آب و هوا با توجه به وضعیت فعلی و گذشته آن استفاده می شود. در رابطه ۶-۲۰، x_t مقدار متغیر در دینامیک t، a, b, c, d پارامترهای مدل و m, n مقادیر اسکالری هستند که به ترتیب دینامیک خروجی و ورودی را مشخص می کنند. در صورتی که رفتار سیگنال تعریف شده غیرخطی باشد به مدل خود همبسته نیز مانند روابط ۶-۱۰ و ۶-۱۱ می توان نگاشت غیرخطی اضافه کرد. برای هر دو نوع دادگان ساختار یافته و غیرساختار یافته می توان مدل خود همبسته تعریف کرد. آموزش پارامترهای مدل خود همبسته به همانند مدل رگرسیون و بر اساس فاصله اقلیدسی بین خروجی و مقدار مطلوب به ازای نمونه های مختلف اجرا می شود.

$$\hat{x}_{t+n} = ax_t + bx_{t-1} + \ldots + cx_{t-m} + d \qquad (۶-۲۰)$$

۶.۴.۴ خوشه بندی[3]

یکی دیگر از مسائل مطرح در مباحث یادگیری ماشین، مسئله خوشه بندی است. هدف از خوشه بندی تقسیم نمونه های مجموعه دادگان بر اساس مقادیر ویژگی های مختلف آن ها است. تقسیم نمونه ها به خوشه های مختلف در الگوریتم های خوشه بندی صرفا براساس مقادیر ویژگی های، ورودی ها، آن ها بوده و مقدار مطلوبی برای تعلق نمونه ها به خوشه ها تعریف نمی شود از این نظر این الگوریتم ها، الگورتیم های دارای آموزش بدون سرپرست محسوب می شوند. از بین

[1] Autoregressive models
[2] Autoregression
[3] Clustering

الگوریتم های مختلف خوشــه بندی در اینجا صــرفا به بررســی روش خوشــه بندی K-means می پردازیم. روند اجرای این الگوریتم بدین شرح است:

۱. ابتدا تعداد خوشـــه هایی، K، که می خواهیم نمونه های مختلف را به آن ها نســـبت دهیم مشخص می کنیم.
۲. به تعداد خوشـه ها در فضـای ویژگی های مسـئله نقاط تصـادفی تعریف می کنیم این نقاط را مراکز دسته می نامیم.
۳. فاصله همه نمونه ها را از همه مراکز دسـته ها محاسـبه می کنیم و به هر دسـته تعداد n نمونه ای که نسـبت به سـایر نمونه ها به مرکز آن دسـته نزدیک تر هسـتند نسـبت می دهیم. n از تقســیم تعداد کل نمونه ها بر تعداد دســته ها محاســبه می شــود، $n = \frac{N}{K}$، N تعداد کل نمونه های مجموعه دادگان است.
۴. نقطه میانگین نمونه های قرار گرفته در هر دسـته را محاسـبه کرده و مختصـات آن را در هر دسـته به نقطه تصـادفی اولیه که به عنوان مرکز دسته تعریف کردیم نسبت می دهیم.
۵. مرحله ۳ را به ازای مراکز دسته جدید تکرار می کنیم.
۶. مجددا میانگین نمونه های هر دسته را محاسبه کرده و مختصات آن را به مرکز دسته نسبت می دهیم، مقادیر مربوط به مختصات مرکز دسته را به روزرسانی می کنیم.
۷. مراحل ۵ و ۶ را تا زمانی تکرار می کنیم که تغییرات مقادیر مختصات مربوط به مراکز دســته ها در هر بار تکرار این مراحل نســبت به مقدار مرحله قبل کمتر از حد معینی باشد.

۶.۵ دسته بندی مدل های یادگیری ماشین

مدل های تعریف شـده در حوزه یادگیری ماشـین را می توان به انواع مختلف دسـته بندی کرد که در اینجا به دو مورد از مرسوم ترین آن ها اشاره می کنیم:

۶.۵.۱ دسته بندی مدل ها براساس نحوه آموزش

مدل های یادگیری ماشین براساس نحوه آموزش به سه دسته مدل های دارای آموزش با نظارت ، با سـرپرسـت ، مدل های دارای آموزش بدون نظارت ، بدون سـرپرسـت، و مدل های دارای آموزش نیمه نظارتی[1]. همان طور ک پیش تر نیز اشاره کردیم در طی روند آموزش مدل های دارای آموزش با نظارت برای مقادیر خروجی مدل به ازای مقادیر ورودی نمونه های مختلف مجموعه دادگان، یک مقدار مطلوب تعریف شـده و پارامترهای مدل براسـاس اختلاف بین مقدار مطلوب و مقدار خروجی

[1] Semi-supervised

که توسط یک معیار، تابع هزینه[1]، تعریف شده سنجیده می شود، آموزش داده می شوند در حالی که در مدل های دارای آموزش بدون نظارت این مقدار مطلوب برای خروجی مدل تعریف نشده و آموزش صرفا براساس مقادیر ورودی صورت می گیرد. در کنار این دو روش آموزش، مدل های دارای آموزش نیمه نظارتی تعریف می شوند که در آن ها برای بخشی از دادگان مقادیر مطلوب در نظر گرفته شده و بخشی دیگر از دادگان فاقد مقادیر مطلوب هستند.

۶.۵.۲ مدل های مولد[2] و متمایز کننده[3]

یکی دیگر از دسته بندی مدل های یادگیری ماشین تقسیم آن ها به دو دسته مدل های مولد و متمایزکننده است. مدل های مولد مدل هایی هستند که طی روند آموزش ویژگی هایی از دادگان آموزشی را تقلید می کنند، در برخی مدل ها از این ویژگی ها برای تولید نمونه های مشابه با نمونه های موجود در دادگان آموزشی، نمونه های دارای توزیع احتمالاتی یکسان با نمونه های دادگان آموزشی، استفاده می شود. ویژگی هایی که از نمونه های مجموعه دادگان به مدل آموزش داده می شود معمولا ویژگی های آماری مانند توزیع احتمالاتی و... از مجموعه دادگان هستند، ماشین های بولتزمن[4] و شبکه های مولد متخاصم[5] و طبقه بند بیز ساده نمونه هایی از مدل های مولد هستند. در مقابل مدل های مولد، مدل های متمایزکننده تعریف می شوند، در مدل های متمایزکننده هدف صرفا ایجاد یک خروجی به ازای مقادیر ورودی است، مدل رگرسیون لجستیک و شبکه های عصبی پرسپترون چند لایه[6] نمونه هایی از مدل های متمایزکننده هستند.

[1] Cost function
[2] Generative
[3] Discriminative
[4] Boltzmann machine
[5] Generative adversarial networks (GANs)
[6] Multi-layer perceptron

۶.۶ مسائل

۱. رابطه ۶-۶ را از رابطه ۵-۶ اثبات کنید.

۲. در صورتی که ماتریس X^TX رابطه ۹-۶ معکوس پذیر نباشد می توانیم همچنان از آن برای محاسبه پارامترهای مدل رگرسیون استفاده کنیم؟ توضیح دهید.

۳. در مورد توابع کرنل مرسوم برای ایجاد تفکیک پذیری در ابعاد بالاتر مدل های طبقه بند تحقیق کرده و آن ها را با هم مقایسه کنید.

۴. در مورد روش های محاسبه توزیع های احتمالاتی به کاربرده شده در مدل طبقه بند بیز ساده تحقیق کنید.

۵. آیا می توان یک مدل مولد را به متمایز کننده و یا بالعکس تبدیل کرد؟ توضیح دهید.

فصل ۷
شبکه های عصبی مصنوعی[1]

۷.۱ مقدمه

شبکه های عصبی مصنوعی که به اختصار به آن ها شبکه های عصبی[2] نیز اطلاق می شود، پایه اصلی مدل های طرح شده در حوزه یادگیری ژرف محسوب می شوند. سنگ بنای شبکه های عصبی مدل پرسپترون[3] است، در واقع شبکه های عصبی در راستای رفع معضل مربوط به عدم توانایی مدل پرسپترون در حل مسئله XOR توسعه پیدا کرده اند.

در این فصل ابتدا به بررسی مدل پرسپترون و پیش زمینه تعریف شبکه های عصبی پرداخته و سپس مفاهیم و تعاریف مطرح در شبکه عصبی های عصبی را معرفی می کنیم همچنین در ادامه انواع مختلف شبکه‌های عصبی، مشکلات و چالش‌های موجود در آن ها را مطرح می کنیم. در انتهای فصل نیز مسائل مربوط به شبکه های عصبی برای مطالعه و تحقیق بیشتر در این حوزه تعریف شده است.

۷.۲ مدل پرسپترون

مدل پرسپترون یک مدل یادگیری ماشین است که می توان آن را برای مسائل رگرسیون و طبقه بندی و یا یک مدل خود همبسته به کار برد. رابطه کلی مدل پرسپترون برای یک مدل رگرسیون، طبقه بند و خود همبسته به ترتیب مشابه روابط ۴-۶، ۱۱-۶ و ۲۰-۶ است. تفاوت اصلی مدل پرسپترون با سایر مدل های رگرسیون، طبقه بند و خود همبسته که در فصل ۶ معرفی کردیم، نحوه آموزش آن است؛ آموزش این مدل بر پایه پس انتشار خطا[4] به صورت یک روش مبتنی بر گرادیان[5] تعریف می شود، در صورتی که آموزش بخش عمده ای از مدل های مطرح شده در فصل ۶، بر اساس محاسبه نقطه بهینه[6] تابع هزینه مدل و در یک مرحله مطابق روابطی مانند رابطه

---

[1] Artificial neural networks (ANNs)
[2] Neural networks (NNs)
[3] Perceptron
[4] Error back propagation
[5] Gradient based methods
[6] Optimum

تعریف می شد. آموزش یک مدل پرسپترون به صورت تکراری[1] تعریف می شود. روش پس انتشار خطا به روش گرادیان نزولی یکی از روش های مرسوم آموزش مدل های پرسپترون است. روند این روش آموزش برای یک مدل رگرسیون مطابق رابطه ۱-۷ به صورت ذیل تعریف می کنیم. در رابطه ۱-۷، o خروجی اسکالر مدل، $f_{(.)}$ یک تابع خطی یا غیرخطی، w_i پارامتر، وزن[2]، مربوط به بعد x_i از بردار ورودی $\boldsymbol{x}$ و b اسکالر ترم بایاس[3] است.

$$o_{(\mathbf{x})} = f\left(w_1x_1 + w_2x_2 + ... + w_nx_n + b\right) \tag{۷-۱}$$

۱. ابتدا مطابق رابطه ۱-۶، نمونه های مختلف را بین بازه عددی صفر تا یک و یا منفی یک تا یک نرمال سازی می کنیم.

۲. برای هر یک از پارامترهای w_i یک مقدار تصادفی در بازه منفی یک تا یک با توزیع احتمالاتی یکنواخت[4]، احتمال انتخاب مقادیر مختلف یکسان است، انتخاب می کنیم. برای ترم بایاس نیز یک مقدار اولیه تعریف می کنیم، مقدار اولیه ترم بایاس معمولا برابر با یک است.

۳. مقادیر ورودی به ازای ویژگی های مختلف، بردار ورودی، یک نمونه از مجموعه دادگان آموزشی را در رابطه ۱-۷ قرار داده و با استفاده از مقادیر تصادفی مرحله ۱ مقدار خروجی o را محاسبه می کنیم.

۴. آموزش مدل پرسپترون یک آموزش با سرپرست محسوب می شود، از این رو هر نمونه از مجموعه دادگان آموزشی دارای بردار ورودی و بردار، اسکالر، مقدار مطلوب است. مقدار محاسبه شده مرحله ۳ را با مقدار مطلوب متناظر با نمونه ورودی با استفاده از یک تابع هزینه، تابع ضرر، تعریف شده برای مدل مقایسه کرده و مقدار تابع هزینه را به ازای ورودی محاسبه می کنیم. برای این مدل رگرسیون فرضی تابع هزینه را مطابق رابطه ۲-۷، به صورت مربع خطای خروجی تعریف می کنیم. در رابطه ۲-۷، E تابع هزینه، e بردار، اسکالر، خطا، d بردار، اسکالر، مقدار مطلوب، o بردار، اسکالر، خروجی مدل است. دلیل استفاده از ضریب $\frac{1}{2}$ تسهیل مشتق گیری از رابطه تابع هزینه است.

$$E_{(o)} = \frac{1}{2}(e)^2 = \frac{1}{2}(d-o)^2 \tag{۷-۲}$$

[1] Iterative
[2] Weight
[3] Bias
[4] Uniform distribution

۵. مقدار مشتق تابع هزینه نسبت به هر یک از پارامترهای w_i مدل را مطابق رابطه ۳-۷ با استفاده از مشتقات زنجیره ای محاسبه می کنیم. در رابطه ۳-۷، $\acute{f}_{(.)}$ مشتق تابع رابطه ۱-۷ بوده و net مطابق رابطه ۴-۷ برابر با مجموع ورودی های وزن دار تعریف می شود.

$$\frac{\partial E}{\partial w_i} = \underset{e}{\frac{\partial E}{\partial e}} \underset{-1}{\frac{\partial e}{\partial o}} \underset{f'_{(net)}}{\frac{\partial o}{\partial net}} \underset{x_i}{\frac{\partial net}{\partial w_i}} \qquad (۷-۳)$$

$$net = w_1 x_1 + w_2 x_2 + \ldots + w_n x_n + b \qquad (۷-۴)$$

۶. هر یک از مقادیر مربوط به پارامترهای w_i را مطابق رابطه ۵-۷ به روزرسانی[1] می کنیم. در رابطه ۵-۷، k مقدار اسکالر نشان دهنده مرحله به روزرسانی و η نرخ یادگیری[2]، نرخ آموزش، است که به صورت یک عدد اسکالر بین صفر تا یک تعریف شده و میزان تاثیر مقدار مشتق تابع هزینه نسبت به پارامتر مورد نظر را در به روزرسانی تعیین می کند.

$$w_{i(k+1)} = w_{i(k)} - \eta \frac{\partial E}{\partial w_{i\ (k)}} \quad , k = 1, \ldots N \qquad (۷-۵)$$

۷. مراحل ۳ تا ۶ را به ازای همه نمونه های موجود در مجموعه دادگان آموزشی تکرار کرده، $k = 1, \ldots N$، و بدین ترتیب یک گام آموزشی[3] در مدل پرسپترون اجرا می کنیم. یک گام آموزشی در مدل پرسپترون را می توانیم مجموعه ای شامل مراحل۳ تا ۶ الگورتیم به ازای همه نمونه های موجود در دادگان آموزشی تعریف کنیم. مقدار تابع هزینه یک گام آموزشی به صورت میانگین مقادیر تابع هزینه به ازای همه نمونه های مجموعه دادگان مطابق رابطه ۶-۷ تعریف می شود. در رابطه ۶-۷، N تعداد نمونه های مجموعه دادگان است.

$$E_{Epoch} = \frac{1}{2N} \sum_{i=1}^{N} (e_i)^2 = \frac{1}{2N} \sum_{i=1}^{N} (d_i - o_i)^2 \qquad (۷-۶)$$

۸. گام های آموزشی مدل را تا جایی ادامه می دهیم که مقدار تابع هزینه در گام های آموزشی متوالی به یک مقدار کوچک قابل قبولی همگرا

[1] Update
[2] Learning rate
[3] Epoch

شود.

روابط ۳-۷ و ۵-۷ را می توانیم به صـورت برداری مطابق روابط ۷-۷ و ۸-۷ تعریف کرده و همه پارامترهای مدل را در یک مرحله با اســـتفاده از آن به روزرســـانی کنیم. در روابط ۷-۷ و ۸-۷، $\boldsymbol{w}$ بردار وزن ها شامل همه پارامترهای وزن، w_i، و $\boldsymbol{x}$ بردار ورودی شامل همه مقادیر مربوط به ویژگی های ورودی، x_i است.

$$\frac{\partial E}{\partial \mathbf{w}} = \underset{e}{\frac{\partial E}{\partial e}} \underset{-1}{\frac{\partial e}{\partial o}} \underset{f'_{(net)}}{\frac{\partial o}{\partial net}} \underset{\mathbf{x}}{\frac{\partial net}{\partial \mathbf{w}}} \tag{۷-۷}$$

$$\mathbf{w}_{(k+1)} = \mathbf{w}_{(k)} - \eta \frac{\partial E}{\partial \mathbf{w}_{(k)}}, k = 1, \ldots N \tag{۷-۸}$$

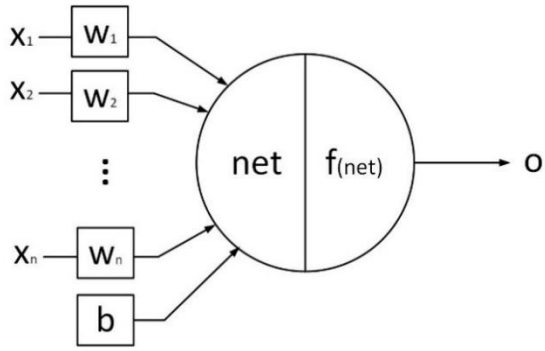
در صـورتی که بخواهیم مقدار مربوط به ترم بایاس را نیز آموزش دهیم، روابط مشـتقات زنجیره ای و به روزرســـانی مقدار آن به ازای هر نمونه را به ترتیب مطابق روابط ۹-۷ و ۱۰-۷ تعریف می کنیم:

$$\frac{\partial E}{\partial b} = \underset{e}{\frac{\partial E}{\partial e}} \underset{-1}{\frac{\partial e}{\partial o}} \underset{f'_{(net)}}{\frac{\partial o}{\partial net}} \underset{1}{\frac{\partial net}{\partial b}} \tag{۷-۹}$$

$$b_{(k+1)} = b_{(k)} - \eta \frac{\partial E}{\partial b_{(k)}}, k = 1, \ldots N \tag{۷-۱۰}$$

در مراحل فوق به محاســبه خروجی مدل به ازای یک نمونه ورودی که در مرحله ۳ شـرح داده شـده اسـت، مرحله پیشـرو[1] و به مرحله ۷ که مربوط به به روزرسـانی پارامترهای مدل با اسـتفاده از رابطه ۸-۷ است پس انتشار خطا به روش گرادیان نزولی گفته می شود. مدل پرسپترون مطرح شده در این بخش را می توانیم به صورت شکل ۱-۷ نمایش دهیم.

[1] Feed forward

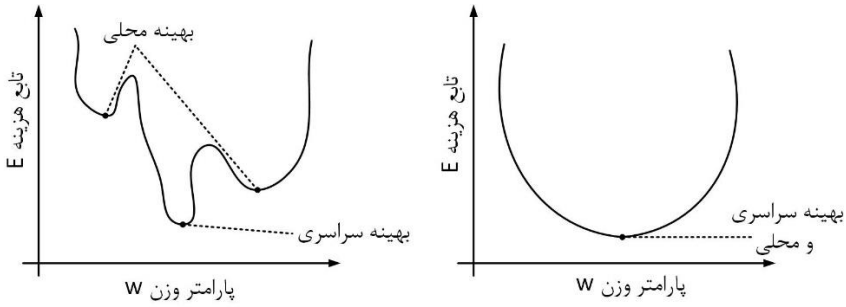


شکل ۷-۱: مدل پرسپترون (نورون عصبی)

۷.۲.۱ توجیه استفاده از مدل پرسپترون

با مقایسه مدل پرسپترون و رابطه تحلیلی ۹-۶ در آموزش یک مدل رگرسیون می توانیم چنین استنباط کنیم که آموزش مدل پرسپترون نسبت به راه حل تحلیلی رابطه ۹-۶ مستلزم حجم محاسبات بوده و زمان اجرای بیشتری دارد. علاوه بر این روش پس انتشار خطا به پارامتری هایی مانند تعداد گام های آموزشی، مقدار نرخ آموزش و... وابسته بوده و مقادیر حاصل از آن لزوما با مقادیر حاصل رابطه ۹-۶ برای پارامترهای مدل که همان مقادیر بهینه برای این پارامترها است برابر نیست؛ به بیان دیگر روش پس انتشار خطا در مدل پرسپترون لزوما به جواب بهینه نمی رسد در حالی که رابطه ۹-۶ همواره مقادیر بهینه پارامترهای مدل را محاسبه می کند. با این اوصاف تعریف مدل پرسپترون و استفاده از روش پس انتشار خطا برای آموزش به جای رابطه های تحلیلی مانند رابطه ۹-۶ چندان منطقی به نظر نمی رسد. اما باید در نظر داشته باشیم که مسئله مربوط به رگرسیون که مقادیر بهینه آن با استفاده از رابطه ۹-۶ محاسبه شد یک مسئله بهینه سازی محدب[1] است در حالی که بسیاری از مسائل مطرح غیرمحدب[2] بوده و تابع هزینه آن ها به ازای مقادیر مختلف پارامترها دارای چندین نقطه کمینه محلی[3] است، از این رو امکان تعریف یک رابطه تحلیلی برای رفتار تابع هزینه بر اساس مقادیر مختلف پارامترهای مدل و بهینه سازی آن مشابه رابطه ۹-۶ برای این مسائل وجود نداشته و یا بسیار دشوار است به همین علت باید پارامترهای مدل تعریف شده برای این مسائل را با روش های تکراری مانند پس انتشار خطا آموزش دهیم. مسئله محدب مسئله ای است که نقطه کمینه، بهینه، سراسری و محلی آن با هم برابر باشند در حالی که مسئله غیرمحدب می تواند دارای چندین نقطه بهینه محلی باشد که بر نقطه بهینه سراسری منطبق نیستند. شکل ۷-۲ مقایسه رفتار محدب و غیر محدب مقادیر تابع هزینه به ازای مقادیر پارامتر w یک مدل فرضی فقط از یک پارامتر اسکالر w تشکیل شده باشد شکل ۷-۲ است.

[1] Convex optimization
[2] Non-convex
[3] Local minimum



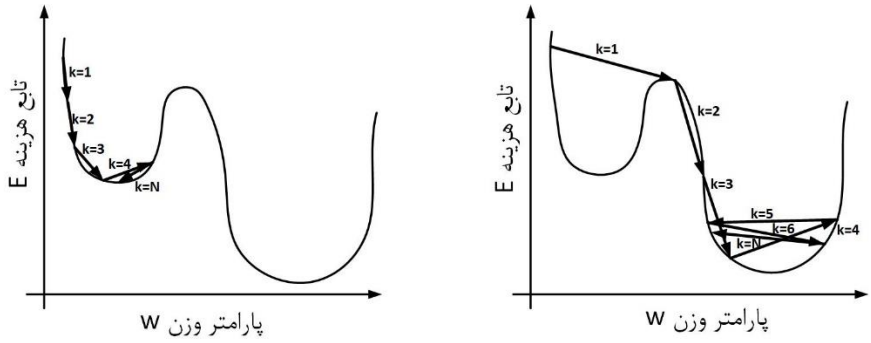
الف) غیرمحدب **ب) محدب**

شکل ۲-۷: مقایسه تابع هزینه محدب و غیرمحدب

۷.۲.۲ بررسی همگرائی[1] و انتخاب نرخ آموزش

طی روند آموزش یک مدل پرپســـپترون مقادیر پارامتر های مدل لزوما به نقطه ای که به ازای آن ها مقدار بهینه سراسری تابع هزینه حاصل می شود، همگرا نمی شوند؛ در صورتی که مقدار نرخ آموزش در روابط به روزرســانی مقدار کوچکی انتخاب شــود علاوه بر اینکه روند آموزش و همگرائی مدل به کندی پیش می رود، ممکن اســـت در نهایت مقادیر پارامترهای مدل به یک نقطه بهینه محلی ختم شوند و مقادیر نقطه بهینه سراسری برای پارامترهای مدل هرگز حاصل نشوند. از سوی دیگر انتخاب مقادیر بزرگ برای نرخ آموزش ممکن اســت باعث ناپایداری روند آموزش و بروز پدیده زیگ زاگ[2] شود؛ در صورت بروز پدیده زیگ زاگ مقادیر پارامترهای مدل در هر مرحله به روزرسانی به میزان قابل توجهی تغییر کرده و در نهایت با ادامه گام های آموزشی به نقطه بهینه خاصی همگرا نمی شوند. تغییرات نوسانی مقدار تابع هزینه در گام های آموزشی متوالی نشانه ناپایدار شدن روند آموزش اســت. با این توضــیحات انتخاب نرخ آموزش مناســب همواره یکی از چالش های مطرح در مدل های پرسپترون است، نرخ آموزش باید به اندازه ای بزرگ انتخاب شود که از همگرائی به نقطه بهینه محلی جلوگیری کرده و باعث بهبود ســـرعت همگرائی و آموزش مدل شـــود و از طرفی این مقدار نباید به اندازه ای بزرگ انتخاب شــود که باعث ناپایداری و بروز پدیده زیگ زاگ شــود. شــکل ۳-۷ مقایسه نرخ آموزش کوچک و بزرگ برای نرخ آموزش است.

[1] Convergence
[2] Zigzag phenomenon



ب) نرخ آموزش بزرگ **الف) نرخ آموزش کوچک**

شکل ۳-۷: مقایسه تاثیر مقدار نرخ آموزش

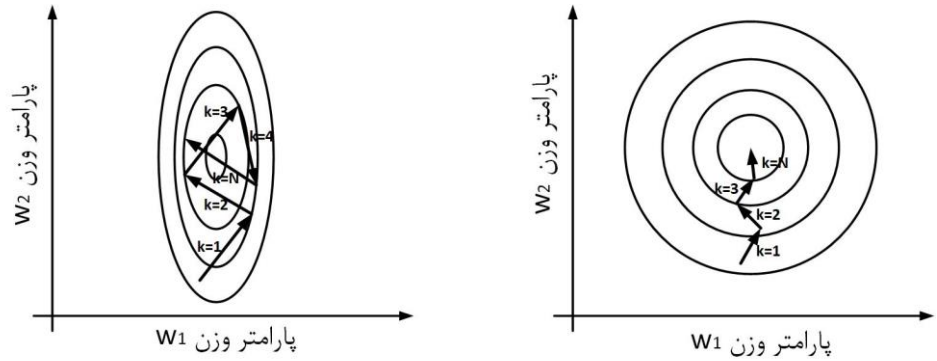
معضـلات همگرائی به نقطه بهینه محلی و سـرعت کم همگرائی معمولا در گام های آموزشـی اولیه مدل و مشـکل ناپایداری مقادیر و عدم همگرائی معمولا در گام های نهائی آموزش مدل بروز پیدا می کنند، از این رو در برخی مدل ها نرخ آموزش را تطبیقی[1] در نظر می گیرند؛ بدین صـورت که در ابتدا مقدار بزرگی برای آن در نظر می گیرند و با ادامه روند آموزش در هر گام آموزشـی به میزان مشـخصـی مقدار آن را کاهش می دهند. به این مقدار تعیین شـده برای کاهش نرخ آموزش، نرخ زوال[2] اطلاق می شـود. کاهش نرخ آموزش می تواند به ازای هر یک یا چند گام آموزشی اعمال شود، همچنین می توان حد پایین مشخصی برای آن تعیین کرد تا پس از رسیدن به آن مقدار نرخ آموزشـی کاهش پیدا نکند. مقادیر نرخ آموزش بسـیار کوچک ممکن اسـت باعث ایجاد اعدادی با تعداد رقم اعشـار بالا در روابط مدل شـود، در این صـورت اگر تعداد ارقام این اعداد در حافظه رایانه تعریف نشده باشند مقادیر غیر عددی، NaN[3]، در روابط ظاهر شده و روند آموزش مدل را با اختلال مواجه می کند، از این رو تعریف حد پایین برای نرخ آموزش تطبیقی ضروری است.

۷.۲.۳ لزوم نرمال سازی

در معرفی روند آموزش مدل پرسـپترون اشـاره کردیم که ابتدا مقادیر مجموعه دادگان را در بازه عددی منفی یک تا یک و یا صفر تا یک نرمال می کنیم، این مرحله از پیش پردازش علاوه بر اینکه بازه تعریف مقادیر دادگان را به بازه مقادیر پارامترهای مدل انتقال می دهد در مواردی که بازه مقادیر ویژگی های مختلف با هم متفاوت باشد، با تبدیل بازه آن ها به یک بازه یکسان از بروز پدیده زیگ زاگ جلوگیری می کند؛ فرض کنید یک مدل پرسـپترون دارای دو پارامتر وزن w_1, w_2 برای دو ویژگی ورودی x_1, x_2 باشـد، به طوری که بازه مقادیر این دو ویژگی با هم متفاوت اسـت، در این صـورت با توجه به ظاهر شـدن مقدار مربوط به ویژگی های ورودی در رابطه ۳-۷ بازه مربوط به

[1] Adaptive learning rate
[2] Decay rate
[3] Not a number

تغییرات مقادیر این پارامترهای در مراحل به روزرسانی نیز با هم متفاوت بوده و در نتیجه بازه مقادیر تابع هزینه نسبت به مقادیر پارامترهای مدل در هر بعد متفاوت خواهد بود، این تفاوت بازه ابعاد مختلف باعث می شود بردار به روزرسانی مقادیر پارامترهای مدل در هر مرحله از نظر اندازه و بعد با بردار مرحله قبل اختلاف قابل توجهی داشته و باعث ناپایداری روند آموزش و بروز پدیده زیگ زاگ شود. شکل ۴-۷ تفاوت استفاده و عدم استفاده از نرمال سازی برای مقادیر ویژگی های ورودی مدل فرضی فوق الذکر را نشان می دهد.



ب) استفاه از نرمال سازی **الف) عدم استفاده از نرمال سازی**

شکل ۴-۷: تاثیر نرمال سازی مقادیر ورودی

۷.۲.۴ ارزیابی[1] مدل

هدف کلی در مسائل یادگیری ماشین تعریف یک مدل عمومی برای همه مقادیر تعریف شده برای مسئله و آموزش آن با استفاده از مقادیر تعدادی نمونه تعریف شده در فضای آن مسئله است؛ پس از آموزش، مدل باید این قابلیت را داشته باشد که به ازای هر مقدار ورودی که در بازه مسئله تعریف شده است یک مقدار خروجی مناسب ارائه کند، البته ممکن است عملکرد مدل به ازای ورودی های خارج از بازه نمونه های آموزشی نسبت به ورودی های داخل بازه نمونه های آموزشی ضعیف تر باشد. با این توضیحات ارزیابی و بررسی عملکرد مدل پس از آموزش به ازای ورودی هایی که به عنوان نمونه آموزشی تعریف نمی شوند در همه مدل های یادگیری ماشین که برای مسائل مختلف ارائه می شوند ضروری است. بدین منظور پس از نرمال سازی مقادیر نمونه های مجموعه دادگان بخشی از آن ها را به عنوان دادگان ارزیابی در نظر گرفته و از نمونه های آموزشی جدا می کنند. بدین ترتیب نمونه های مجموعه دادگان به دو بخش آموزش و ارزیابی تقسیم می شوند. تعداد نمونه های بخش ارزیابی معمولا کمتر از نمونه های آموزش انتخاب می شود. بدین ترتیب

[1] Validation

پس از اتمام هر گام آموزشی و محاسبه مقدار تابع هزینه گام آموزشی، رابطه ۶-۷، یک گام ارزیابی[1] نیز تعریف می شود، روند گام ارزیابی مشابه گام آموزشی است با این تفاوت که این گام فاقد مرحله به روزرسانی، مراحل ۵ و ۶، بوده و صرفا مقدار خروجی مدل به ازای مقادیر نمونه های ارزیابی محاسبه شده، روند پیشرو، و بر اساس آن ها مقدار تابع هرینه گام ارزیابی با استفاده از رابطه ای مشابه رابطه ۶-۷ محاسبه می شود. هر گام آموزشی و ارزیابی در کنار هم یک گام در روند آموزش مدل محسوب می شوند. رابطه تابع هزینه هر دو گام آموزشی و ارزیابی یکسان است. در تعریف گام آموزشی و ارزیابی باید همواره توجه کنیم که در طول روند آموزش نمونه های آموزش و ارزیابی از هم تفکیک شده و به هیچ وجه با هم جایگزین نشوند، به عبارت دیگر نباید از یک نمونه آموزشی به عنوان نمونه ارزیابی و بالعکس استفاده کرد. با توجه به تشابه روابط گام آموزش و ارزیابی در این کتاب بیشتر به بررسی گام آموزشی مدل های مختلف می پردازیم ولی در عمل به ازای هر گام آموزشی یک گام ارزیابی نیز تعریف می شود که در پیاده سازی مدل باید آن را در نظر بگیریم.

۷.۳ تعریف یک شبکه عصبی

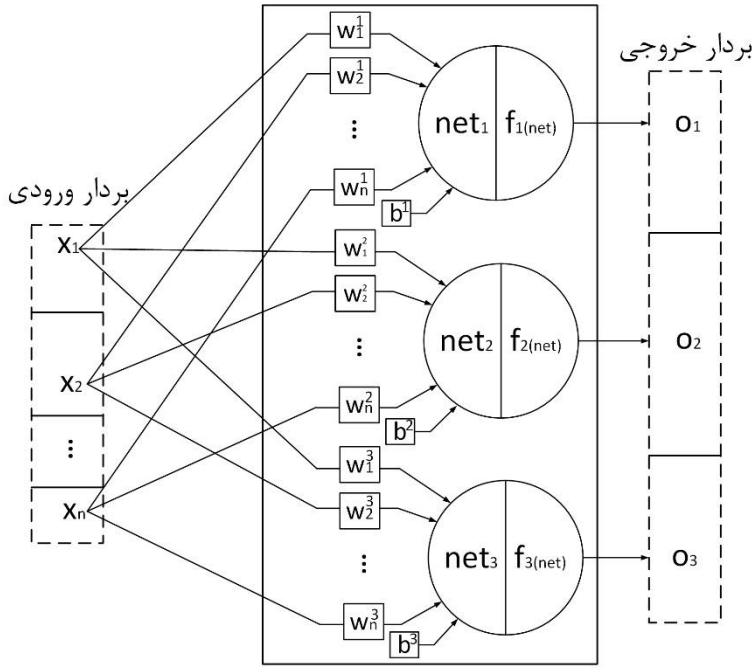
مدل پرسپترونی که در بخش ۲-۷ معرفی کردیم را در نظر بگیرید، این مدل را به صورت شکل ۱-۷ نشان می دهیم. این مدل در ساختار شبکه های عصبی به عنوان یک نورون[2] عصبی شناخته می شود. در این صورت به هر یک از وزن های w_i آن وزن های سیناپسی[3] نورون، به تابع $f_{(.)}$ تابع فعال ساز[4] نورون و به مقدار اسکالر o خروجی نورون اطلاق می شود. رابطه ۴-۷ را می توانیم برای یک نورون با تعریف یک ضرب ماتریسی بین بردار، ماتریس، ورودی $\boldsymbol{x}$ شامل همه مقادیر x_i و بردار، ماتریس، وزن های $\boldsymbol{w}$ شامل همه وزن های w_i به صورت رابطه ۱۱-۷ نشان دهیم.

$$net_{1\times1} = \mathbf{x}^T_{n\times1} \times \mathbf{w}_{n\times1} = \begin{bmatrix} x_1 & x_1 & \cdots & x_n \end{bmatrix} \times \begin{bmatrix} w_1 \\ w_2 \\ \vdots \\ w_n \end{bmatrix} \qquad (۷-۱۱)$$

همان طور که در فصل ۶ نیز اشاره کردیم، ممکن است یک مدل رگرسیون یا خود همبسته دارای چند خروجی باشد و یا یک مسئله طبقه بندی شامل چند کلاس باشد. در چنین مواردی برای هر خروجی یا هر کلاس یک مدل جداگانه تعریف شده و این مدل ها به صورت موازی کنار هم قرار می گیرند. مدل پرسپترون تعریف شده برای یک مسئله رگرسیون دارای سه خروجی

[1] Validation epoch
[2] Neuron
[3] Synaptic weights
[4] Activation function

مطابق شکل ۵-۷ خواهد بود. همان طور که در شکل ۵-۷ نیز نشان داده شده است بردار ورودی تعریف شده برای هر سه مدل یکسان بوده، و تعداد پارامترهای وزن، w_i، نیز در هر یک از آن ها برابر با اندازه بردار ورودی است، بدین ترتیب در هر مدل به هر بعد از بردار ورودی یک پارامتر اسکالر وزن اختصاص داده می شود. کل پارامترهای وزن سه مدل را می توانیم به صورت یک ماتریس وزن، $W_{n\times 3}$، تعریف کنیم. ساختاری که در شکل ۵-۷ نمایش داده شده، یک شبکه عصبی متشکل از یک لایه دارای سه نورون است. رابطه پیشرو این ساختار به فرم ماتریسی این شبکه مطابق روابط ۱۲-۷ و ۱۳-۷ است. در رابطه ۳۱-۷، $\boldsymbol{o}$ بردار خروجی لایه، شبکه، و $\boldsymbol{b}$ بردار بایاس است.



شکل ۵-۷: ساختار متشکل از چند مدل پرسپترون (لایه عصبی)

$$\mathbf{net}_{1\times 3} = \mathbf{x}^{T}_{n\times 1} \times W_{n\times 3} + \mathbf{b}_{1\times 3} \qquad (۷-۱۲)$$

$$\mathbf{o}_{1\times 3} = f_{(\mathbf{net}_{1\times 3})} \qquad (۷-۱۳)$$

بردار خطا و مقدار تابع هزینه این شبکه مطابق رابطه ۱۴-۷ محاسبه می شود.

$$E_{(\mathbf{o})} = \frac{1}{2}\left\|\mathbf{e}_{1\times 3}\right\|_2 = \frac{1}{2}\left\|\mathbf{d}_{1\times 3} - \mathbf{o}_{1\times 3}\right\|_2 \qquad (۷-۱۴)$$

برای به روزرسانی ماتریس وزن های شبکه روابط ۷-۱۵ و ۷-۱۶ را داریم.

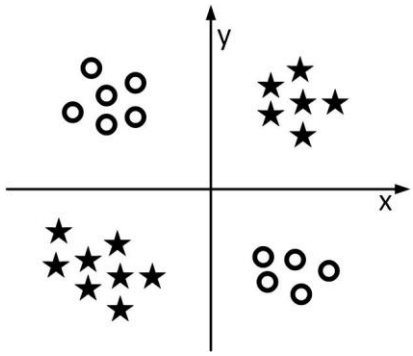
$$\underset{n\times 3}{\frac{\partial E}{\partial W}} = \underset{\mathbf{e}_{1\times 3}}{\frac{\partial E}{\partial \mathbf{e}}} \underset{-1}{\frac{\partial \mathbf{e}}{\partial \mathbf{o}}} \underset{f'_{(\mathbf{net})_{3\times 3}}}{\frac{\partial \mathbf{o}}{\partial \mathbf{net}}} \underset{\mathbf{x}_{n\times 1}}{\frac{\partial \mathbf{net}}{\partial W}} \qquad (۷-۱۵)$$

$$W_{(k+1)} = W_{(k)} - \eta \frac{\partial E}{\partial W_{(k)}} \quad , k = 1,...N \qquad (۷-۱۶)$$

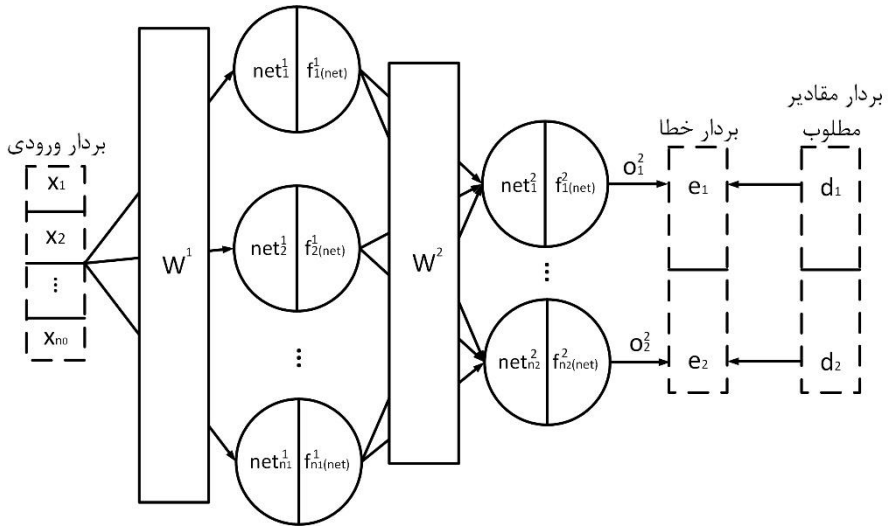
بین جملات مختلف رابطه مشتقات زنجیره ای، ۷-۱۵، ضرب ماتریسی تعریف می شود، در صورت عدم تطبیق ابعاد دو جمله یکی از ترانهاده یکی از دو جمله در روابط استفاده می شود. در رابطه ۷-۱۵ منظور از $f'_{(.)}$ ژاکوبین تابع فعال ساز است. با توجه به تعریف روابط پیشرو به صورت برداری برای تابع فعال ساز در رابطه ۷-۱۳، این تابع دارای ورودی و خروجی برداری است، از این رو برای آن برای آن به جای مشتق، ژاکوبین تعریف می شود. ژاکوبین تابع فعال ساز رابطه ۷-۱۳ مطابق رابطه ۷-۱۷ تعریف می شود، با توجه به عدم اتصال بین نورون های یک لایه در این رابطه همه درایه ها به جز درایه های قطر اصلی مقادیر صفر دارند. در رابطه ۷-۱۷، $J_{(.)}$ ژاکوبین و $\acute{f}_{n(net_m)}$ مشتق تابع فعال ساز نورون n-ام نسبت به ورودی net نورون m است.

$$f'_{(\mathbf{net})_{3\times 3}} = J(f_{(\mathbf{net}_{1\times 3})}) = \begin{bmatrix} f'_{1(net_1)} & f'_{1(net_2)} & f'_{1(net_3)} \\ f'_{2(net_1)} & f'_{2(net_2)} & f'_{2(net_3)} \\ f'_{3(net_1)} & f'_{3(net_2)} & f'_{3(net_3)} \end{bmatrix} \qquad (۷-۱۷)$$

ساختار شبکه عصبی پرسپترون متشکل از یک لایه که در شکل ۷-۵ نمایش داده شده است قادر به حل مسئله طبقه بندی XOR که از دو کلاس تشکیل شده است نیست، شکل ۷-۶ نمایش این مسئله طبقه بندی است. از این رو برای مدتی طولانی پس از معرفی آن، این مدل چندان مورد توجه قرار نگرفت تا اینکه با افزودن یک لایه به صورت سری به آن این معضل رفع شد. بدین ترتیب یک ساختار شبکه عصبی پرسپترون دو لایه مطابق شکل ۷-۷ که متشکل از یک لایه پنهان و یک لایه خروجی است معرفی شد.



شکل ۶-۷: مسئله طبقه بندی XOR



شکل ۷-۷: ساختار شبکه عصبی پرسپترون دو لایه فعال

در حالت کلی ساختار شبکه های عصبی مانند ساختار شکل ۷-۷ پرسپترون چند لایه[1] گفته می شود. همچنین به طور کلی در ساختار این شبکه ها همه ابعاد بردار ورودی یک لایه به همه نورون های آن لایه متصل است، از این رو به آن ها شبکه های دارای اتصالات تماما متصل و یا به اختصار شبکه های تماما متصل[2] گفته می شود، در صورتی که برخی از این اتصالات حذف شوند در این صورت شبکه حاصل دارای اتصالات محلی[3] است. در نمادها و روابط یک شبکه عصبی عدد بالاوند[4] نشان دهنده ترتیب لایه از سمت ورودی و عدد زیروند[5] مربوط به ترتیب نورون های لایه از بالا به پایین است؛ برای مثال $f_{n(.)}^{m}$ نماد تابع فعال ساز نورون n از لایه m ساختار شبکه عصبی

[1] Multi-layer perceptron (MLP)
[2] Fully connected neural network
[3] Locally connected neural network
[4] Superscribe
[5] Subscribe

است. در صورتی که یک نماد فقط دارای بالاوند باشد مربوط همه مقادیر تعریف شده برای یک لایه است. برای ساختار یک شبکه عصبی پرسپترون دو لایه مطابق شکل ۷-۷ روابط پیشرو مطابق روابط ذیل است:

$$\mathbf{net}^1_{1\times n_1} = \mathbf{x}^T_{n_0\times 1}\times W^1_{n_0\times n_1} + \mathbf{b}^1_{1\times n_1} \qquad (۷\text{-}۱۸)$$

$$\mathbf{o}^1_{1\times n_1} = f_{(\mathbf{net}^1_{1\times n_1})} \qquad (۷\text{-}۱۹)$$

$$\mathbf{net}^2_{1\times n_2} = \mathbf{o}^1_{1\times n_1}\times W^2_{n_1\times n_2} + \mathbf{b}^2_{1\times n_2} \qquad (۷\text{-}۲۰)$$

$$\mathbf{o}^2_{1\times n_2} = f_{(\mathbf{net}^2_{1\times n_2})} \qquad (۷\text{-}۲۱)$$

روابط ۷-۱۸ و ۷-۱۹ روابط پیشرو لایه اول و روابط ۷-۲۰ و ۷-۲۱ روابط پیشرو لایه دوم هستند. در روابط فوق n_2, n_1, n_0 به ترتیب بعد ورودی، تعداد نورون های لایه اول و تعداد نورون های لایه دوم، بعد خروجی هستند، همان طور که در این روابط نیز نشان داده شده است خروجی لایه اول به عنوان ورودی لایه دوم تعریف می شود. بردار خطا و مقدار تابع هزینه این شبکه مطابق رابطه ۷-۲۲ محاسبه می شود.

$$E_{(\mathbf{o}^2)} = \frac{1}{2}\left\|\mathbf{e}_{1\times n_2}\right\|_2 = \frac{1}{2}\left\|\mathbf{d}_{1\times n_2} - \mathbf{o}_{1\times n_2}\right\|_2 \qquad (۷\text{-}۲۲)$$

برای به روزرسانی مقادیر وزن های لایه اول و دوم این شبکه روابط ذیل برقرار است:

$$\frac{\partial E}{\partial W^2_{n_1\times n_2}} = \underbrace{\frac{\partial E}{\partial \mathbf{e}}}_{\mathbf{e}_{1\times n_2}}\underbrace{\frac{\partial \mathbf{e}}{\partial \mathbf{o}^2}}_{-1}\underbrace{\frac{\partial \mathbf{o}^2}{\partial \mathbf{net}^2}}_{f'_{(\mathbf{net}^2)_{n_2\times n_2}}}\underbrace{\frac{\partial \mathbf{net}^2}{\partial W^2}}_{\mathbf{o}^1_{1\times n_1}} \qquad (۷\text{-}۲۳)$$

$$\frac{\partial E}{\partial W^1_{n_0\times n_1}} = \underbrace{\frac{\partial E}{\partial \mathbf{e}}}_{\mathbf{e}_{1\times n_2}}\underbrace{\frac{\partial \mathbf{e}}{\partial \mathbf{o}^2}}_{-1}\underbrace{\frac{\partial \mathbf{o}^2}{\partial \mathbf{net}^2}}_{f'_{(\mathbf{net})_{n_2\times n_2}}}\underbrace{\frac{\partial \mathbf{net}^2}{\partial \mathbf{o}^1}}_{W^2_{n_1\times n_2}}\underbrace{\frac{\partial \mathbf{o}^1}{\partial \mathbf{net}^1}}_{f'_{(\mathbf{net}^1)_{n_1\times n_1}}}\underbrace{\frac{\partial \mathbf{net}^1}{\partial W^1}}_{\mathbf{x}_{1\times n_0}} \qquad (۷\text{-}۲۴)$$

$$W^2_{(k+1)} = W^2_{(k)} - \eta\left.\frac{\partial E}{\partial W^2}\right|_{(k)}, k = 1,\ldots N \qquad (۷\text{-}۲۵)$$

$$W^1_{(k+1)} = W^1_{(k)} - \eta \frac{\partial E}{\partial W^1}_{(k)} \quad , k = 1,...N \qquad (۷-۲۶)$$

با توجه به ظاهر شدن مقدار وزن لایه دوم در گام k در روابط مشتق زنجیره ای مربوط به لایه اول در پیاده سازی الگوریتم آموزشی ساختار باید ابتدا وزن های لایه اول و سپس لایه دوم به روزرسانی شود، به طور کلی همه مقادیر استفاده شده در جملات مشتقات زنجیره ای مربوط به یک پارامتر باید با گام مقدار فعلی آن پارامتر هماهنگ باشند. با افزایش تعداد لایه های یک شبکه عصبی، عملکرد ساختار حاصل در مدل کردن رفتار غیرخطی مقادیر ورودی و خروجی بهبود پیدا می کند اما افزایش لایه ها به بیش از سه لایه باعث افزایش تعداد جملات مشتقات زنجیره ای، به خصوص ظاهر شدن تعداد بالاتر مشتق تابع فعال ساز، می شود که حاصل آن کوچک شدن مقدار حاصل از ضرب این جملات، مقدار تغییرات وزن های در هر گام به روزرسانی، می شود، این پدیده که به آن محوشدگی گرادیان ها[1] گفته می شود باعث می شود مقادیر پارامترهای لایه های اولیه آموزش مناسبی نداشته باشند. در فصل های آینده با رویکردها و راهکارهای ارائه شده برای رفع این مشکلات و تعریف ساختارهای دارای تعداد لایه بالا خواهیم پرداخت. به طور کلی به ساختار شبکه ای که بیش از دو لایه داشته باشد ساختار شبکه ژرف[2] اطلاق می شود.

۷.۴ بیش برازش[3] و کم برازش[4]

انتخاب پارامترهای مربوط به ساختار و آموزش یک شبکه عصبی مانند تعداد نورون ها، نوع توابع فعال ساز، تعداد گام آموزشی، نرخ آموزش و... از چالش های مطرح در این مدل ها است. انتخاب نامناسب این پارامترها باعث اختلال در روند آموزش و عملکرد ساختار می شود. یکی از عمده مشکلاتی که در طی روند آموزش یک شبکه، یا هر مدل یادگیری ماشین دیگر که دارای آموزش تکراری است، بروز پیدا می کند مشکل بیش برازش و کم برازش است. بیش برازش زمانی اتفاق می افتد که مقدار تابع هزینه به ازای نمونه های ارزیابی به اندازه قابل توجهی از مقدار تابع هزینه حاصل از نمونه های آموزشی در یک گام مشترک، k، بزرگ تر باشد. بدین ترتیب مدل حاصل به ازای نمونه های آموزشی عملکرد مناسب و به ازای نمونه های ارزیابی عملکرد ضعیفی دارد، این مدل را نمی توان یک مدل عمومی به طوری که از آن بتوان برای محاسبه خروجی مقادیری که در نمونه های آمورشی تعریف نشده است، تلقی کرد. از عمده دلایل بروز پدیده بیش برازش، بالا بودن تعداد پارامترهای آموزشی ساختار، بالا بودن تعداد گام های آموزشی اشاره کرد. در حالت کلی برای برطرف کردن مشکل بیش برازش می توان از راهکارهای ذیل استفاده کرد:

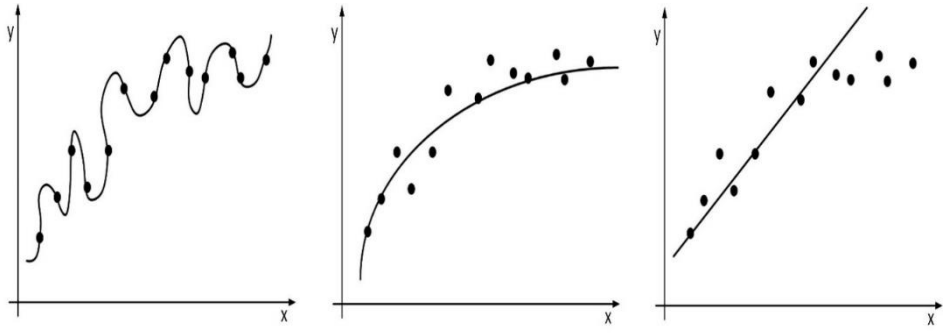
[1] Gradient vanishing
[2] Deep neural network
[3] Overfitting
[4] Underfitting

- کاهش تعداد گام های آموزشی؛ تعداد گام های آموزشی را تا جایی افزایش می دهیم تا اختلاف مقدار تابع هزینه گام آموزش و گام ارزیابی در گام های متوالی همواره نزولی باشد، روند صعودی این مقدار نشانه بروز پدیده بیش برازش است.
- کاهش تعداد پارامترهای مدل
- به هم زدن ترتیب نمونه ها در مجموعه دادگان قبل از تقسیم آن ها به مجموعه نمونه های آموزش و ارزیابی[1]؛ در صورتی که مقادیر مربوط به یک یا چند ویژگی ورودی در نمونه های مجموعه دادگان به ترتیب، از مقدار کوچک به بزرگ، تعریف شده باشند با تقسیم نمونه ها به دو مجموعه آموزش و ارزیابی، همواره بخشی از بازه عددی آن ویژگی فقط در نمونه های آموزش و بخش دیگر بازه در نمونه های ارزیابی تعریف شده و مدل آموزش داده شده با این دادگان به ازای نمونه های ارزیابی عملکرد مناسبی نخواهد داشت. برای رفع این معضل با به هم زدن ترتیب نمونه ها توزیع مقادیر نمونه ها در هر دو مجموعه آموزش و ارزیابی به طور مساوی از سراسر بازه عددی آن ویژگی خواهد بود.
- استفاده از روش حذف تصادفی[2]؛ در این روش در هر گام آموزشی تعدادی از نورون ها از یک لایه، صرفا لایه های میانی؛ این روش برای لایه خروجی قابل استفاده نیست، به صورت تصادفی حذف شده و روابط پیشرو و آموزش صرفا برای نورون های باقی مانده تعریف می شود، برای پیاده سازی این روش مقدار خروجی و مقدار مشتقات زنجیره ای وزن های نورون حذف شده را به ترتیب در روابط پیشرو و آموزش صفر در نظر می گیریم، بدین ترتیب در هر گام آموزشی وزن های برخی از نورون ها، نورون های حذف شده، ثابت می مانند. این روش یک حالت فرضی است که صرفا در گام آموزش تعریف می شود و ساختار اصلی شبکه تغییری نمی کند، بدین ترتیب با آغاز گام ارزیابی این نورون ها دوباره در روابط ساختار تعریف می شوند. ساختار نهائی بعد از اتمام روند آموزش نیز متشکل از همه نورون هایی که در ابتدا تعریف شده اند، خواهد بود.

در مقابل پدیده بیش برازش، پدیده کم برازش تعریف می شود، در پدیده کم برازش تابع هزینه هر دو گام آموزش و ارزیابی مقدار بزرگی داشته و با ادامه روند آموزش ساختار تغییر چندانی نمی کند. در این صورت مدل به ازای نمونه های هر دو مجموعه آموزش و ارزیابی عملکرد ضعیفی داشته و آموزش پارامترهای آن باعث بهبود عملکرد آن نمی شود. کم بودن تعداد پارامترهای مدل از عمده دلایل بروز پدیده کم برازش است؛ برای مثال فرض کنید یک مدل رگرسیون خطی برای مجموعه دادگانی که رفتار غیرخطی، برای مثال درجه دو، دارند تعریف کنیم در این صورت مقادیر

[1] Shuffling
[2] Drop out

پارامترهای مدل به ازای هر مقداری عملکرد مناسبی نخواهند داشت. برای رفع این معضل می توان تعداد پارامترهای مدل را با افزایش نورون های لایه پنهان و یا افزایش تعداد لایه ها، افزایش دهیم. شکل ۸-۷ مقایسه سه حالت بیش برازش، آموزش مناسب و کم برازش برای یک مسئله رگرسیون است. همان طور که پیش تر نیز اشاره کردیم در حالت بیش برازش در یک گام متشرک مقدار تابع هزینه گام ارزیابی به مقدار قابل توجهی بزرگ تر از تابع هزینه گام آموزش بوده و با ادامه روند آموزش این اختلاف بیشتر می شود، تابع هزینه گام آموزش نزولی و گام ارزیابی صعودی می شود، در حالتی که ساختار آموزش مناسبی داشته باشد روند تغییرات مقدار تابع هزینه هر دو گام آموزشی و ارزیابی در گام های متوالی نزولی بوده و مقدار تابع هزینه گام ارزیابی همواره کمی بزرگ تر از تابع هزینه گام آموزش است. در حالت کم برازش نیز مقدار تابع هزینه هر دو گام دارای مقادیر بزرگی بوده و در برخی موارد این مقادیر دچار نوسان می شود. همچنین در بعضی از موارد کم برازش مقدار تابع هزینه گام ارزیابی کوچک تر از تابع هزینه گام آموزش است.



ج) کم برازش **ب) آموزش مناسب** **الف) بیش برازش**

شکل ۸-۷: مقایسه بیش برازش، آموزش مناسب و کم برازش

۷.۵ مقادیر اولیه و توابع فعال ساز

در معرفی روند آموزش یک مدل پرسپترون اشاره کردیم که مقادیر اولیه وزن ها به طور تصادفی با توزیعی یکنواخت از بازه منفی یک تا یک انتخاب می شوند، فرض کنید در یک شبکه عصبی دولایه ای که دارای خروجی یک بعدی، اسکالر، است، همه این وزن های تصادفی اولیه در لایه دوم شبکه مقادیر مثبت داشته باشند و تابع فعال ساز لایه اول نیز تابع سیگموئید باشد، در این صورت خروجی لایه اول نیز همواره مقدار مثبتی بین صفر تا یک خواهد داشت. در روند آموزش این ساختار فرضی مقدار همه درایه های رابطه ۲۳-۷ که مقدار تغییرات وزن لایه دوم است با توجه به علامت خطای اسکالر همواره افزایشی یا کاهشی خواهد بود، این حالت باعث بروز پدیده زیگ زاگ می شود زیرا همه وزن های این لایه در هر مرحله به روزرسانی یا افزایش و یا کاهش می یابند، در صورتی که در طی یک روند آموزش مناسب برخی از این مقادیر باید افزایش و برخی

کاهش یابند. این حللت برای لایه اول زمانی اتفاق می افتد که علاوه بر شـــرایط فوق للذکر مقادیر ورودی نیز بین صفر تا یک نرمال سازی شده باشند. با توجه به شرایط وقوع این حالت برای رفع آن می توانیم موارد ذیل را اعمال کنیم:

- نرمال سازی مقادیر ورودی بین منفی یک تا یک
- استفاده از مقادیر مثبت و منفی در وزن های اولیه لایه های مختلف
- استفاده از تابع فعال سازی که خروجی آن بازه شامل بازه اعداد منفی نیز باشد.

تعداد نورون ها و تابع فعال ساز در لایه خروجی ساختار بر اساس مسئله تعریف می شود؛ معمولا در مسئله رگرسیون و خود همبستگی از توابع خطی در این لایه استفاده می شود، تعداد نورون ها نیز برابر با تعداد خروجی های مدل، بعد بردار مقادیر مطلوب، است، البته استفاده از توابع غیرخطی مانند تانژانت هیپربولیک نیز در برخی از این مسائل عملکرد مناسبی دارد.س در مسائل طبقه بندی معمولا از تابع سیگموئید در لایه خروجی استفاده می شود این تابع عملکردی مشابه رگرسیون لجستیک داشته و یک خروجی احتمالی بین صفر تا یک دارد، تعداد این نورون ها برابر با تعداد کلاس های تعریف شده، در صورتی که دو کلاس ناسازگار باشند یک نورون، است. برای تعریف توابع فعال ساز لایه های پنهان قید خاصی مطرح نمی شود، در این نورون ها استفاده از تابع تانژلنت هیپربولیک به جای سیگموئید از آنجایی که دارای خروجی منفی یک تا یک هستند از پدیده زیگ زاگ جلوگیری می کند، علاوه بر این توابع مقدار مشتق بزرگ تری نسبت به تابع سیگموئید به ازای مقادیر ورودی یکسان داشته و از این جهت از بروز پدیده محوشدگی گرادیان ها نیز جلوگیری می کند. بنابراین استفاده از تانژانت هیپربولیک به جای سیگموئید تقریبا در همه موارد باعث بهبود عملکرد ساختار می شود. در صورتی که بخواهیم از تابع تانژانت هیپربولیک در لایه خروجی یک شبکه طبقه بند استفاده کنیم باید مقادیر خروجی ساختار را که در بازه منفی یک تا یک هستند با استفاده از رابطه ۱-۶ به بازه صفر تا یک نگاشت کنیم تا مقادیر به صورت احتمالاتی بین صفر تا یک تعریف شوند، سپس این مقادیر نگاشت شده را به عنوان ورودی تابع هزینه در نظر بگیریم. تابع ReLU نیز یکی دیگر از توابع فعال ساز است که به طور گسترده در ساختارهای پیچشی استفاده می شود این تابع در مقایسه با تابع تانژانت هیپربولیک نگاشت غیرخطی کمتری دارد ولی به دلیل جلوگیری از پدیده محوشدگی گرادیان ها و دلایل دیگری که در فصل ۱۲ بررسی خواهیم کرد، در ساختارهایی که دارای تعداد لایه بالایی هستند استفاده می شود. به طور کلی در ساختارهای دارای تعداد لایه کمتر تابع تانژانت هیپربولیک نسبت به توابع سیگموئید و ReLU عملکرد بهتری دارد. روابط ذیل به ترتیب رابطه تابع فعال ساز سیگموئید و تانژانت هیپربویلک هستند. یک لایه از شبکه عصبی نگاشت کننده فضا محسوب شده و توابع فعال ساز همه نورون های آن از یک نوع تعریف می شود.

$$f_{(x)} = Sigmoid(x) = \frac{1}{1+e^{-x}} \qquad (۷-۲۷)$$

$$f_{(x)} = \mathrm{Tanh}(x) = \frac{e^{x} - e^{-x}}{e^{x} - e^{-x}} \qquad (۷-۲۸)$$

برای تعیین مقادیر یک شبکه عصبی اولیه علاوه بر روش انتخاب مقادیر تصادفی، می توانیم از روش های دیگری نیز استفاده کنیم؛ برای مثال این مقادیر را می توانیم از یک الگوریتم بهینه سازی دیگر مانند الگوریتم ژنتیک[1] و یا سایر الگوریتم های بهینه سازی پردازش تکاملی استفاده کنیم، این الگوریتم ها از همان نمونه های آموزشی ساختار تعریف شده در مجموعه دادگان برای محاسبه مقادیر وزن ها استفاده می کنند. استفاده از این الگوریتم ها حجم محاسباتی بالایی داشته و در مواردی نیز منجر به مقادیر اولیه مناسب نمی شود. در کنار این روش ها دو روش ذیل نیز برای محاسبه وزن های اولیه ارائه شده است که به نسبت بار محاسباتی کمتری داشته و از طرفی استفاده از آن ها عملکرد بهتری نسبت به روش انتخاب مقادیر تصادفی دارد.

۷.۵.۱ روش Xavier

این روش برای تعیین وزن های اولیه نورون هایی به کار می رود که دارای توابع فعال سازی که در نقطه صفر مشتق پذیر هستند تعریف می شود، در این روش مقادیر وزن ها نمونه هایی از یک توزیع گوسی با مقدار میانگین صفر و واریانسی مطابق رابطه ۲۹-۷ است. در رابطه ۹-۲۹، N تعداد نورون های لایه، $f_{(.)}$ تابع فعال ساز، $\acute{f}_{(.)}$ مشتق تابع فعال ساز و σ^2 واریانس توزیع گوسی است

$$\sigma^2 = \frac{1}{N.(f'_{(0)})^2.(1+ {f_{(0)}}^2)} \qquad (۷-۲۹)$$

رابطه ۲۹-۷ برای تابع فعال ساز سیگموئید و تانژانت هیپربولیک به ترتیب مطابق روابط ۳۰-۷ و ۳۱-۷ تعریف می شود.

$$\sigma^2 \approx \frac{3.6^2}{N} \qquad (۷-۳۰)$$

$$\sigma^2 \approx \frac{1}{N} \qquad (۷-۳۱)$$

۷.۵.۲ روش He

این روش برای تعیین وزن های اولیه نورون هایی به کار می رود که دارای توابع فعال سازی هستند که در نقطه صفر مشتق پذیر نیستند، برای مثال تابع فعال ساز ReLU. در این روش نیز مقادیر وزن های اولیه به صورت نمونه هایی از توزیع گوسی با میانگین صفر و واریانسی مطابق

[1] Genetic algorithm (GA)

رابطه ۷-۳۲ تعریف می شوند.

$$\sigma^2 \approx \frac{2}{N} \qquad (۷-۳۲)$$

۷.۶ انواع شبکه های عصبی

ساختار معرفی شده در این فصل، یک ساختار پایه و ابتدایی برای شبکه های عصبی است. در فصل های آینده به بررسی انواع مختلف ساختارهای شبکه های عصبی ژرف و کاربرد آن ها خواهیم پرداخت. در این بخش نیز چند مورد دیگر از ساختار شبکه های عصبی را بررسی می کنیم که همانند ساختاری که پیش تر معرفی کردیم، می توان از آن ها برای مسائل و دادگان مختلف از آن ها استفاده کرد. عملکرد هر یک از این ساختارها به نوع دادگان و سایر پارامترهای دخیل در تعریف ساختار مانند تعداد نورون، نرخ آموزش و... وابسته بوده و نمی توان به طور کلی ساختاری را بهتر از بقیه در نظر گرفت.

۷.۶.۱ شبکه های عصبی مبتنی بر کرنل

در ساختار نورون های شبکه ها به جای استفاده از پارامترهای وزن و توابع فعال ساز از کرنل استفاده می شود. کرنل های مبتنی بر توابع شعاعی[1]، کرنل های گوسی، از مرسوم ترین کرنل ها هستند. در این شبکه ها هدف آموزش پارامترهای کرنل است؛ برای مثال در کرنل های گوسی مقادیر میانگین و واریانس آموزش داده می شوند. مقدار اولیه واریانس و میانگین مانند وزن های شبکه پرسپترون به طور تصادفی انتخاب می شوند. رابطه ۷-۳۳ یک کرنل گوسی است که معادل یک نورون در ساختار شبکه های مبتنی بر کرنل در نظر گرفته می شود. در رابطه ۷-۳۳، $\bar{x}$ میانگین و σ^2 واریانس کرنل است. مقدار اسکالر net در این ساختار مانند شبکه پرسپترون چند لایه محاسبه می شود.

$$\phi_{(net)} = e^{\frac{\|net-\bar{x}\|_2}{2\sigma^2}} \qquad (۷-۳۳)$$

۷.۶.۲ شبکه های عصبی دارای نورون های انعطاف پذیر[2]

ساختار این شبکه ها همانند ساختار شبکه های پرسپترون چندلایه است که در بخش های قبل معرفی کردیم، با این تفاوت که در رابطه تابع فعال ساز هر نورون یک پارامتر آموزشی نیز تعریف می شود. رابطه ۷-۳۴ رابطه تابع فعال ساز سیگموئید با پارامتر انعطاف پذیر a است.

[1] Radial based functions (RBFs)
[2] Flexible activation functions

$$f_{(net,a)} = \frac{a}{1+e^{-net}} \qquad (۷-۳۴)$$

آموزش پارامتر انعطاف پذیر نیز مانند پارامتر های وزن بر اساس مشتقات زنجیره ای تعریف می شود. مشتقات زنجیره ای تعریف شده برای آموزش این پارامتر انعطاف در یک نورون مطابق رابطه ۷-۳۵ تعریف می شود. در رابطه ۷-۳۵، $f^*_{(.)}$ مشتق رابطه تابع فعال ساز، رابطه ۷-۳۴، نسبت به پارامتر انعطاف پذیر است.

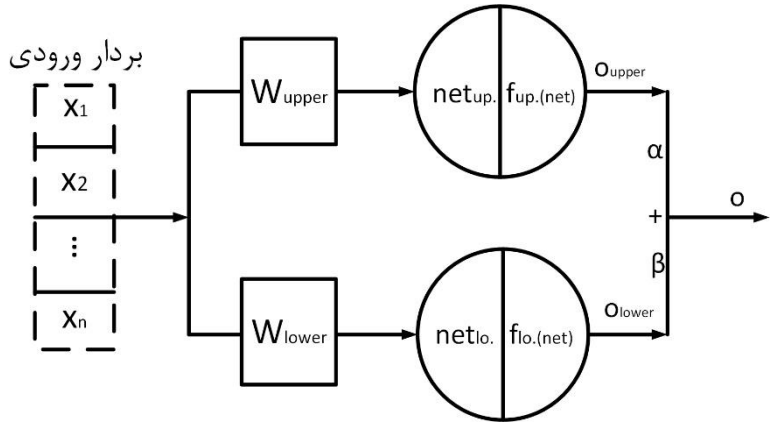
$$\frac{\partial E}{\partial a} = \frac{\partial E}{\partial \mathbf{e}} \underset{\mathbf{e}}{} \frac{\partial \mathbf{e}}{\partial \mathbf{o}^2} \underset{-1}{} \cdots \frac{\partial \mathbf{o}}{\partial a} \underset{f^*_{(\mathbf{a})}}{} \qquad (۷-۳۵)$$

۷.۶.۳ شبکه های عصبی دارای نورون های راف[1]

روند کلی ساختار این شبکه ها نیز مشابه شبکه های عصبی پرسپترون چند لایه بخش های قبل است. تفاوت این شبکه ها با شبکه های پرسپترون چندلایه، ساختار نورون های آن ها است. ساختار نورون های راف مطابق شکل ۷-۹، دارای یک باند بالا و یک باند پایین است. نورون های راف در مواردی که دادگان به نویز آغشته بوده و دارای عدم قطعیت باشند عملکرد مناسبی دارند ولی حجم محاسبات آن ها در مقایسه با ساختار مرسوم بیشتر است. در یک ساختار دو لایه اگر نورون های لایه اول راف باشند به آن راف نوع اول و اگر نورون های لایه دوم راف باشند به آن راف نوع دوم گفته می شود. در روند آموزش این ساختار مقادیر وزن های بلند بالا و پایین به طور مستقل به روزرسانی می شوند، همچنین برای آموزش لایه های پیش از این لایه مقادیر گرادیان هر دو مسیر بالا و پایین با هم جمع می شوند. در شکل ۷-۸، α, β مقادیر اسکالری هستند که وزن هر مسیر را در خروجی نهائی نورون تعیین می کنند، مجموع مقادیر این دو پارامتر همواره برابر با یک بوده و در حالت کلی برای هر دوی آن ها مقدار ۰.۵ در نظر گرفته می شود. همچنین برای این پارامترها نیز می توان روابط آموزشی تعریف کرده و مقادیر آن ها را در طی روند آموزش ساختار به روزرسانی کرد. مقدار خروجی یک نورون راف مطابق رابطه ۷-۳۶ محاسبه می شود.

$$o = \alpha.o_{upper} + \beta.o_{lower} \qquad (۷-۳۶)$$

[1] Rough neural networks



شکل ۹-۷: ساختار نورون راف

۷.۷ مسائل

۱. روابط مشتقات زنجیره ای مربوط به پارامترهای میانگین و واریانس در ساختار شبکه مبتنی بر کرنل های گوسی را بنویسید.

۲. روابط پیشرو و آموزش یک ساختار شبکه عصبی دو لایه که هر دو لایه آن دارای نورون های راف هستند را بنویسید.

۳. در مورد شبکه های عصبی مبتنی بر آموزش عاطفی مغز[1] تحقیق کنید.

[1] Brain inspired emotional learning based neural networks (BEL)

فصل ۸
توابع هزینه[1] و معیارهای ارزیابی[2]

۸.۱ مقدمه

همان طور که در مطالب فصل های ۶ و ۷ مطرح کردیم آموزش یک مدل بر اساس کمینه کردن تابع هزینه آن تعریف می شود، بنابراین تعریف تابع هزینه مناسب با توجه به مسئله و مدل ارائه شده برای آن اهمیت زیادی دارد. در این فصل انواع توابع هزینه که برای مسائل و مدل های گوناگون تعریف می شوند را معرفی کرده و روابط مربوط به هر کدام از آن ها را بررسی می کنیم. علاوه بر توابع هزینه معیارهای مختلفی را که برای ارزیابی عملکرد مدل های مختلف تعریف می شوند را مطرح می کنیم. بررسی روش های تقسیم نمونه های مجموعه دادگان آموزشی برای ارزیابی مدل نیز از مطالب طرح شده در این فصل است. در نهایت مسائل مربوط به این فصل برای آشنائی بیشتر با مطالب ارائه می شود.

۸.۲ توابع هزینه

هدف کلی از آموزش یک مدل در مسائل مختلف رگرسیون، طبقه بندی و... تنظیم پارامترهای مدل به گونه ای است که مقدار تابع درست نمائی[3] مدل به ازای همه نمونه های مجموعه دادگان است. تابع درست نمائی یک مدل مطابق رابطه ۱-۸ تعریف می شود. در رابطه ۱-۸، n تعداد نمونه های مجموعه دادگان، $\hat{y}_i$ خروجی تخمین مدل به ازای ورودی x_i، w مجموعه پارامترهای مدل و $P_{model(.)}$ توزیع احتمالاتی درست نمائی مدل برای نمونه i است. در حالت کلی مقدار درست نمائی به ازای یک نمونه هنگامی مقدار بیشینه، یک، خواهد داشت که خروجی تخمین مدل به خروجی مطلوب y نزدیک باشد

$$\arg\max\left(\prod_{i=1}^{n} P_{\text{model}(\hat{y}_i|w,\mathbf{x}_i)}\right) \qquad (۸-۱)$$

---

[1] Cost function
[2] Criterions
[3] Likelihood function

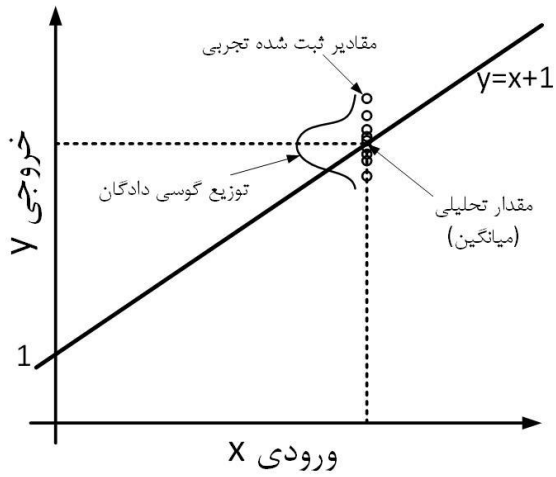
برای سهولت در محاسبات به ویژه محاسبات مربوط به مشتق و جلوگیری از ایجاد اعداد دارای تعداد ارقام اعشار بالا که منجر به اشغال حافظه می شود رابطه ۱-۸ را با اعمال لگاریتم منفی به صورت رابطه ۲-۸ تعریف می کنیم.

$$\arg\min\left(\frac{1}{n}\sum_{i=1}^{n}-\log\left(P_{\text{model}(\hat{y}_i|w,\mathbf{x}_i)}\right)\right) \qquad (۸-۲)$$

اعمال لگاریتم منفی در رابطه ۲-۸، علاوه بر موارد فوق باعث می شود که مسئله مقدار بیشینه رابطه ۱-۸ به مسئله مقدار کمینه تبدیل شده و به صورت یک تابع هزینه که هدف آموزش مدل کاهش مقدار آن است تعریف شود. مقدار رابطه ۲-۸ در حالت ایده آل که مقدار احتمال $P_{model(.)}$ به ازای همه نمونه ها یک باشد برابر با صفر خواهد بود. همچنین در صورتی که مقدار احتمال $P_{model(.)}$ به ازای نمونه ای کوچک، نزدیک به صفر، باشد استفاده از لگاریتم منفی باعث می شود که اندازه گرادیان، شیب، متناظر با این مقدار احتمالی در رابطه ۲-۸ مقدار بزرگی داشته باشد، این مورد که معمولا در گام های آموزشی اول مدل نمود پیدا می کند تاثیری مانند استفاده از نرخ آموزش تطبیقی داشته و باعث بهبود سرعت همگرائی مدل و جلوگیری از همگرائی به نقاط بهینه محلی می شود. استفاده از ضریب $\frac{1}{n}$ در رابطه ۲-۸ رفع حساسیت رابطه ۲-۸ به تعداد نمونه های مجموعه دادگان و نرمال سازی آن است.

همان طور که در مطالب فصل ۳ نیز اشاره کردیم در اکثر موارد در صورتی که تعداد نمونه های ثبت شده از یک پدیده به سمت بینهایت میل کند توزیع احتمالی مقادیر نمونه های مربوط به آن پدیده یک توزیع گوسی خواهد بود، برای درک این فرض کنید که مقدار خروجی یک سیستم، پدیده، خطی که دارای یک ورودی و خروجی است را به ازای مقدار ثابت ورودی به دفعات ثبت کنیم. اگر رابطه تحلیلی ۳-۸ را برای این سیستم فرض کنیم در حالت ایده آل همواره مقدار خروجی برابر با مقدار حاصل از رابطه ۳-۸ باشد اما در اکثر موارد به دلیل عدم قطعیت در پدیده ها مقدار ثبت شده کوچکتر یا بزرگتر از مقدار حاصل از رابطه ۳-۸ خواهد بود، اما تعداد نمونه هایی که مقادیر آن ها فاصله قابل توجهی از مقدار رابطه ۳-۸ دارد کمتر از نمونه هایی است که مقادیر آن ها به مقدار رابطه ۳-۸ نزدیک است، بنابراین با نزدیک شدن مقادیر به مقدار رابطه ۳-۸ احتمال متناظر با آن مقادیر نیز افزایش می یابد. از این رو مقادیر این نمونه به ازای یک ورودی ثابت ها از یک توزیع گوسی که میانگین آن همان مقدار رابطه ۳-۸ به ازای آن ورودی است. هر چه پراکندگی این نمونه ها از مقادیر میانگین بیشتر باشد مقدار واریانس این توزیع بزرگ تر خواهد بود. شکل ۱-۸ نمایش پراکندگی این ثبت شده و توزیع گوسی پراکندگی آن ها است. در رابطه ۸-۳، x ورودی و y خروجی سیستم است. بدین ترتیب برای مقادیر نمونه ها فرض آغشته بودن به نویز گوسی با میانگین صفر و واریانسی برابر با واریانس حاصل از مقادیر نمونه های ثبت شده و مقدار رابطه ۸-۳ در نظر می گیریم.

$$y = x + 1 \tag{۸-۳}$$



شکل ۱-۸: پراکندگی دادگان با توزیع گوسی

فرض توزیع احتمالاتی گوسی را برای توزیع احتمالاتی $P_{model(.)}$ در روابط ۱-۸ و ۲-۸ نیز در نظر می گیریم، بدین ترتیب احتمال مقادیر نزدیک به مقدار مطلوب بزرگتر از مقادیری است که فاصله بیشتری نسبت به مقدار مطلوب دارند. بدین ترتیب توزیع احتمالاتی $P_{model(.)}$ یک توزیع احتمالاتی مطابق رابطه ۸-۴ خواهد بود. در رابطه ۴-۸، y_i مقدار مطلوب برای خروجی تخمین $\hat{y}_i$ مدل است که معادل با مقدار میانگین توزیع در نظر گرفته شده است، همچنین σ^2 واریانس توزیع گوسی است که برای آن یک مقدار فرضی ثابت در نظر می گیریم.

$$P_{\text{model}(\hat{y}_i|w,\mathbf{x}_i)} = \frac{1}{\sqrt{2\pi\sigma^2}} e^{\frac{\|\hat{y}_i - y_i\|_2}{2\sigma^2}} \tag{۸-۴}$$

با جایگذاری رابطه ۴-۸ در رابطه ۲-۸ رابطه ۵-۸ استنباط می شود.

$$\arg\min\left(\frac{1}{n}\sum_{i=1}^{n} -\log\left(P_{\text{model}(\hat{y}_i|w,\mathbf{x}_i)}\right)\right) =$$

$$\arg\min\left(-\frac{1}{n}\log\sigma - \frac{1}{2n}\log(2\pi) + \frac{1}{n}\sum_{i=1}^{n}\frac{\|\hat{y}_i - y_i\|_2}{2\sigma^2}\right) \tag{۸-۵}$$

با توجه به مقدار ثابت σ و πدر رابطه ۵-۸، با حذف جملات ثابت در نهایت رابطه ۵-۸ به

صورت رابطه ۶-۸ خواهد بود که همان تابع هزینه میانگین مربعات خطا[1]، فاصله اقلیدسی خروجی از مقدار مطلوب، است که به عنوان تابع هزنیه در مدل هایی که خروجی کمی پیوسته دارند، مانند مدل رگرسیون و خود همبسته، استفاده می شود. در رابطه ۶-۸، مخرج دو این جمله در رابطه ۵-۸ را حذف نمی کنیم تا محاسبه مشتق این تابع هزینه نسبت به ورودی آن آسان تر باشد. در رابطه ۶-۸، $\boldsymbol{y}$ بردار مقادیر مطلوب شامل همه مقادیر y_i و $\hat{\boldsymbol{y}}$ بردار خروجی مدل شامل همه مقادیر $\hat{y}_i$ است.

$$E_{(\hat{\mathbf{y}},\mathbf{y})} = \frac{1}{2n}\sum_{i=1}^{n}\left\|\hat{y}_i - y_i\right\|_2 \qquad (۸-۶)$$

۸.۲.۱ تنظیم کننده[2] وزن

با اضافه کردن جملاتی به تابع هزینه مدل علاوه بر کمینه کردن خطای خروجی مدل، سایر مواردی که در طی روند آموزش یک مدل به وجود می آیند را کنترل کرد. یکی از مشکلاتی که در روند آموزش یک مدل با آموزش مبتنی بر گرادیان، شبکه عصبی، به وجود می آید رشد بیش از گرادیان[3] های ساختار است؛ در این صورت مقدار گرادیان های محاسبه شده متاثر از مقدار خطا و نرخ آموزش مقدار بزرگی دارد. رشد بیش از حد گرادیان ها در ابتدا باعث ناپایداری روند آموزش ساختار شده و در ادامه باعث رشد بیش از حد وزن[4] های ساختار می شود. در صورت رشد بیش از حد وزن ها، وزن های ساختار مقادیر بسیار بزرگ یا بسیار کوچکی اختیار می کنند، در صورتی که تعداد ارقام این مقادیر بیش از ظرفیت حافظه منبع محاسباتی باشد مقادیر NaN[5] در روابط ساختار ظاهر شده و باعث اختلال در روند آموزش ساختار می شود. علاوه بر این رشد بیش از حد وزن ها باعث بروز پدیده بیش برازش در طی روند آموزش می شود. برای رفع این معضل کاهش مقدار نرخ رشد باعث مشکلاتی می شود که پیش تر در مطالب فصل ۷ به آن ها اشاره کردیم. بدین ترتیب برای جلوگیری از رشد بیش از حد مقادیر وزن های ساختار می توانیم یک جمله تنظیم کننده وزن ها به تابع هزینه ساختار اضافه کنیم. جمله تنظیم کننده ساختار مجموع نرم اول یا دوم مقادیر وزن های یک یا چند لایه ساختار است که به تابع هزینه اضافه می شود. برای مثال تابع هزینه مجموع مربعات خطای رابطه ۶-۸ را در نظر بگیرید که برای یک شبکه عصبی پرسپترون دو لایه تعریف شده است. در صورت اضافه کردن جمله تنظیم کننده وزن ها برای هر دو لایه ساختار به صورت نرم اول به تابع هزینه، این تابع مطابق رابطه ۸-۷ تعریف می شود. در رابطه ۷-۸، عبارت $\sum_{u=1}^{l}(.)$ مربوط به لایه های ساختار است، در صورتی که جمله تنظیم وزن برای یک لایه تعریف شده باشد این عبارت در رابطه تعریف نمی شود، همچنین در صورتی که تنظیم جملات صرفا برای

[1] Mean squared errors (MSE)
[2] Weight regularizer
[3] Gradient explosion
[4] Weight explosion
[5] Not a number

چند لایه خاص اعمال شده باشد مقادیر u باید فقط به ازای مقادیر مربوط به شماره لایه های وزن های مورد نظر تعریف شوند، در این رابطه $w_{j,k}^{u}$ درایه j,k ماتریس وزن های لایه u و λ یک پارامتر اسکالر برای تنظیم تاثیر جمله تنظیم وزن در مقدار نهائی تابع هزینه است، معمولا در محاسبه مقدار پارامتر λ تعداد داریه های ماتریس وزن ها نیز اعمال می شود، مقدار پارامتر λ به صورت ضریبی از معکوس تعداد درایه های ماتریس وزن ها در نظر گرفته می شود؛ $\lambda = \frac{\alpha}{N}$ به طوری که N تعداد درایه های ماتریس وزن و α پارامتر اسکالر، تا مقدار این جمله نسبت به ابعاد ماتریس وزن ها نرمال شود.

$$E_{(\hat{\mathbf{y}},\mathbf{y})} = \frac{1}{2n}\sum_{i=1}^{n}\left\|\hat{y}_i - y_i\right\|_2 + \lambda\sum_{u=1}^{l}\sum_{j}\sum_{k}\left\|w_{j,k}^{u}\right\|_1 \qquad (۸-۷)$$

مشتق زنجیره ای تابع هزینه رابطه ۸-۷ برای وزن های لایه دوم و اول ساختار به ترتیب مطابق روابط ۸-۸ و ۸-۹ تعریف می شود.

$$\frac{\partial E}{\partial W_{n,m}{}^{2}} = \frac{\partial E}{\partial e}\frac{\partial e}{\partial o^2}\frac{\partial o^2}{\partial net^2}\frac{\partial net^2}{\partial o^1} + \frac{1}{\lambda}[1]_{n,m} \qquad (۸-۸)$$

$$\frac{\partial E}{\partial W_{n,m}{}^{1}} = \frac{\partial E}{\partial e}\frac{\partial e}{\partial o^2}\frac{\partial o^2}{\partial net^2}\frac{\partial net^2}{\partial W^2}\frac{\partial W^2}{\partial o^1}\frac{\partial o^1}{\partial net^1}\frac{\partial net^1}{\partial \mathbf{x}} + \frac{1}{\lambda}[1]_{n,m} \qquad (۸-۹)$$

در روابط ۸-۸ و ۸-۹، $e = \hat{y} - y$ و $y = o^2$ است، همچنین منظور از [1] در این روابط یک ماتریس با ابعادی برابر با ابعاد ماتریس وزن های لایه مورد نظر است که همه درایه های آن مقداری برابر با یک دارند. در استفاده از نرم دوم در جمله تنظیم وزن ها رابطه تابع هزینه ساختار مطابق رابطه ۸-۱۰ خواهد بود.

$$E_{(\hat{\mathbf{y}},\mathbf{y})} = \frac{1}{2n}\sum_{i=1}^{n}\left\|\hat{y}_i - y_i\right\|_2 + \lambda\sum_{u=1}^{l}\sum_{j}\sum_{k}\left\|w_{j,k}^{u}\right\|_2 \qquad (۸-۱۰)$$

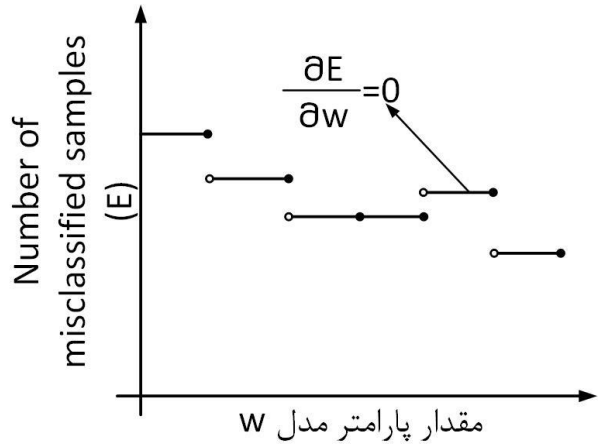
استفاده از نرم دوم در جمله تنظیم کننده وزن ها حجم محاسبات را افزایش می دهد ولی تغییرات مقدار تابع هزینه با تغییر مقادیر وزن ها نسبت به نرم اول ملایم تر خواهد بود، این موضوع در پایداری روند آموزش مدل موثر است. در توسعه روابط مربوط به جمله تنظیم کننده وزن ها که در این بخش معرفی کردیم، مقادیر وزن ها در بازه اعداد مثبت فرض شده است.

۸.۲.۲ تابع هزینه مسئله طبقه بندی

هدف از آموزش مدل طبقه بند است، کاهش تعداد نمونه هایی است که مدل کلاس آن ها را

اشتباه تشخیص می دهد[1]. اگر تابع هزینه مدل طبقه بند را برابر با تعداد این نمونه ها تعریف کنیم از آن جائی که تعداد این نمونه ها به ازای مقادیر مختلف وزن های مدل، به صورت مقادیر گسسته تعریف می شود مشتق تابع هزینه نسبت به پارامتر وزن مدل همواره برابر با صفر خواهد بود، این موضوع در نمودار شکل ۲-۸ نمایش داده شده است. بنابراین تعریف تابع هزینه ای برابر با تعداد نمونه هایی که کلاس آن ها به اشتباه تشخیص داده است برای آموزش یک مدل با روشی های مبتنی بر گرادیان کاربردی ندارد. همان طور که در ابتدای بخش نیز بررسی کردیم حالت کلی تابع هزینه یک مدل مطابق رابطه ۲-۸ تعریف می شود، برای یک مدل طبقه بندی که خروجی آن به صورت یک مقدار احتمالی بین صفر تا یک تعریف می شود، مانند رگرسیون لجستیک، مقدار خروجی مدل را می توانیم مستقیما در رابطه ۲-۸ استفاده کنیم، بدین ترتیب رابطه تابع هزینه یک مدل طبقه بند با خروجی احتمالی مطابق رابطه ۱۱-۸ تعریف می شود.

$$E_{(\hat{\mathbf{y}},\mathbf{y})} = -\frac{1}{n}\sum_{i=1}^{n} y_i \log(P_{\text{model}(\hat{y}_i|w,\mathbf{x}_i)}) \qquad (۸-۱۱)$$



شکل ۲-۸: تعریف تابع هزینه بر اساس تعداد نمونه ها با کلاس خروجی اشتباه

در رابطه ۱۱-۸، y_i مقدار مطلوب برای نمونه i-ام است که مقداری برابر با یک یا صفر دارد. با مقایسه رابطه ۱۱-۸ با رابطه ۲-۴ فصل ۴ می توانیم چنین برداشت کنیم که رابطه ۱۱-۸ همان آنتروپی متغیر خروجی، $\boldsymbol{y}$، است که با آموزش مدل کاهش می یابد. رابطه۱۱-۸ برای مسئله ای کاربرد دارد که در آن صرفا یک کلاس تعریف شده است، این مدل برای حالت دو کلاس ناسازگاری که برای هر دو کلاس خروجی مشترک تعریف می شود کاربردی ندارد، زیرا در این صورت به ازای نمونه های کلاسی که مقدار مطلوب صفر برای آن در نظر گرفته شده است مقدار رابطه ۱۱-۸ همواره صفر است، بنابراین نمونه های این کلاس در محاسبه مقدار تابع هزینه نقشی نداشته و مدل

[1] Misclassified Samples

حاصل در تشخیص این کلاس عملکرد مناسبی نخواهد داشت. برای رفع این معضل آنتروپی کلاس دوم که مقدار مطلوب صفر برای آن در نظر گرفته شده است را نیز وارد رابطه کرده و تابع هزینه مسئله طبقه بندی برای دو کلاس را که یک خروجی مشترک با مقادیر مطلوب صفر و یک برای آن تعریف می شود مطابق رابطه ۱۲-۸ تعریف می کنیم.

$$E_{(\hat{\mathbf{y}},\mathbf{y})} = -\frac{1}{n}\sum_{i=1}^{n}(y_i \log(P_{\text{model}(\hat{y}_i|w,\mathbf{x}_i)}) + (1-y_i)\log(1-P_{\text{model}(\hat{y}_i|w,\mathbf{x}_i)}))$$

(۸-۱۲)

به رابطه ۱۲-۸، آنتروپی متقابل باینری[1] گفته می شود. تابع رابطه ۱۲-۸ برای حالتی که از دو خروجی مجزا برای طبقه بندی دو کلاس ناسارگار استفاده شود و یا مسئله شامل بیش از دو کلاس ناسازگار باشد کاربرد ندارد، تابع هزینه چنین مسائلی را مطابق رابطه ۱۳-۸ تعریف می کنیم که به آن آنتروپی متقابل دسته ای[2] گفته می شود. در رابطه ۱۳-۸، n تعداد نمونه ها و m تعداد کلاس های تعریف شده است. همچنین در این رابطه $\boldsymbol{y}$ بردار خروجی و $\widehat{\boldsymbol{y}}$ بردار Onehot مقادیر مطلوب است که اندازه آن ها برابر با تعداد کلاس ها بوده و پارامتر j به ازای ابعاد مختلف آن ها تعریف می شود.

$$E_{(\hat{\mathbf{y}},\mathbf{y})} = -\sum_{i=1}^{n}\sum_{j=1}^{m}(\mathbf{y}_j \log(sm(\hat{\mathbf{y}})_j))_i$$

(۸-۱۳)

در رابطه ۱۳-۸، $sm(.)$ تابع بیشینه هموار[3] است که مطابق رابطه ۱۴-۸ تعریف می شود، این تابع دارای ورودی و خروجی بردار است، با اعمال تابع بیشینه هموار به یک بردار مجموع مقادیر ابعاد مختلف آن بردار برابر با یک شده و از این رو این تابع یک بیان آماری از مقادیر بردار ورودی خود ارائه می کند، علاوه بر این این تابع به بردار ورودی یک نگاشت غیرخطی اعمال می کند به طوری که در خروجی آن اختلاف بزرگ ترین مقدار بین ابعاد بردار ورودی بیشتر از می شود این عامل باعث تمایز بیشتر مقدار این بعدی از بردار و متعاقبا تشخیص کلاس نمونه ورودی می شود. از آن جائی که ورودی و خروجی تابع بیشینه هموار همانند یک لایه شبکه عصبی بردار است، در محاسبات مشتقات زنجیره ای برای آن ژاکوبین خروجی نسبت به ورودی تعریف می شود.

$$sm(\mathbf{y})_i = \frac{e^{y_i}}{\sum_{j=1}^{n} e^{y_j}}$$

(۸-۱۴)

[1] Binary cross entropy
[2] Categorical cross entropy
[3] Soft-max function

رابطه ۸-۱۴ محاسبه هر یک ابعاد بردار خروجی تابع بیشینه هموار را نشان می دهد، در این رابطه $\boldsymbol{x}$ بردار ورودی و x_i, x_j به ترتیب بعد i و j بردار هستند. همان طور که اشاره کردیم تابع بیشینه هموار رفتاری مشابه با لایه ساختار شبکه عصبی دارد، از این رو در مواردی این تابع به عنوان آخرین لایه ساختار شبکه در نظر گرفته شده و خروجی آن به عنوان خروجی نهائی شبکه تعریف می شود، در صورتی که بعد خروجی لایه پیش از لایه بیشینه هموار با تعداد کلاس های مسئله برابر نباشد برای لایه بیشینه هموار نیز همچون سایر لایه ها یک ماتریس وزن برای تبدیل ابعاد ورودی لایه به تعداد کلاس های خروجی ساختار تعریف کنیم اما در صورتی که بعد ورودی لایه با بعد خروجی آن برابر باشد استفاده از وزن در لایه بیشینه هموار ضروری نیست. در صورتی که تابع بیشینه هموار جزئی از لایه های ساختار شبکه باشد رابطه تابع هزینه آنتروپی دسته ای، رابطه ۸-۱۳، به صورت رابطه ۸-۱۵ تعریف می شود. به تابع هزینه رابطه ۸-۱۵ که تعریفی از رابطه کلی ۸-۲ است، تابع هزینه لگاریتم منفی درست نمائی[1] گفته می شود.

$$E_{(\hat{\mathbf{y}},\mathbf{y})} = -\sum_{i=1}^{n}\sum_{j=1}^{m}(\mathbf{y}_j \log(\hat{\mathbf{y}}_j))_i \qquad (۸-۱۵)$$

تابع هزینه آنتروپی متقابل صرفا برای حالتی که کلاس های مسئله با هم ناسازگار باشند قابل تعریف است. برای مسئله ای با کلاس های سازگار، تابع هزینه میانگین مربعات خطا، رابطه ۸-۶، تعریف می شود، البته با توجه به اینکه همواره پراکندگی دادگان در این حالت در یک سمت مقدار مطلوب قرار دارد، سمت منفی مقدار مطلوب یک و سمت مثبت مقدار مطلوب صفر، فرض توزیع گوسی این تابع نقض می شود اما با وجود اطلاع از این نقض تئوریک همچنان می توان از این رابطه به عنوان تابع هزینه استفاده کرد.

۸.۲.۳ توابع هزینه توزیع های احتمالاتی

در توابع هزینه ای که در بخش های قبل معرفی کردیم، هدف مقایسه مقدار خروجی یک نمونه با مقدار مطلوب آن نمونه بود در حالی که در بعضی موارد مقدار مطلوب به صورت یک توزیع احتمالاتی تعریف می شود و هدف از آموزش مدل نزدیک کردن توزیع احتمالاتی مقادیر خروجی به توزیع هدف، مطلوب، است. در این بخش به بررسی معیارهایی می پردازیم که برای مقایسه دو توزیع احتمالاتی تعریف شده و می توان آن ها را به عنوان تابع هزینه ساختارهای دارای توزیع احتمالاتی مطلوب استفاده کرد. از آن جائی که برای محاسبه یک توزیع احتمالاتی تعدادی نمونه مختلف استفاده می شود، در این ساختارها نمی توان به ازای یک نمونه مقدار تابع هزینه را محاسبه و پارامترهای مدل را به روزرسانی کرد و باید مقدار تابع هزینه به ازای کل نمونه ها، آموزش یا

[1] Negative log-likelihood (NLL)

ارزیابی، و یا حداقل تعدادی از آن ها محاسبه شود.

۸.۲.۳.۱ معیار واگرائی f[1]

معیار واگرائی f، برای محاسبه میزان تفاوت دو توزیع احتمالاتی تعریف می شود. این معیار را می توان به ازای مقادیر ورودی گسسته و پیوسته تعریف کرد. از آن جائی که در مسائل مربوط به یادگیری ماشین و یادگیری ژرف، مجموعه دادگان به صورت نمونه های مجزا تعریف می شوند حالت گسسته معیار واگرائی f را به عنوان تابع هزینه مطابق رابطه ۱۶-۸ تعریف می کنیم. در رابطه ۱۶-۸، $P_{(x)}, Q_{(x)}$ توزیع احتمالاتی هستند که با استفاده از مقادیر ورودی مشترک x_i محاسبه می شوند، این مقادیر ورودی همان بردار ورودی شبکه است، همچنین $f_{(.)}$ در این رابطه به صورت یک تابع محدب دلخواه تعریف می شود.

$$D_f(Q_{(x)} \,||\, P_{(x)}) = \sum_{i=1}^{n} P_{(x_i)} \, f(\frac{Q_{(x_i)}}{P_{(x_i)}}) \tag{۸-۱۶}$$

در صورتی که دو توزیع $P_{(x)}, Q_{(x)}$ با هم برابر باشند مقدار رابطه ۱۶-۸ برابر با صفر بوده و در حالت کلی $D_f(Q_{(x)}||P_{(x)}) \neq D_f(P_{(x)}||Q_{(x)})$ است، بنابراین معیارهای واگرائی فاصله محسوب نمی شوند. در تابع هزینه ای که مطابق رابطه ۱۶-۸ تعریف می شود، هر یک از دو توزیع $Q_{(x)}$ یا $P_{(x)}$ می تواند نسبت به تعریف مسئله به عنوان توزیع هدف و توزیع دیگر به عنوان توزیع خروجی مدل تعریف شود، این توزیع ها می تواند توزیع های پارامتریک یا غیرپارامتریک باشند.

۸.۲.۳.۲ معیار واگرائی KL[2]

در معرفی معیار واگرائی f اشاره کردیم که در رابطه ۱۶-۸ تابع $f_{(.)}$ یک تابع محدب دلخواه است. در صورتی که از تابع لگاریتم به عنوان تابع $f_{(.)}$ استفاده کنیم، رابطه حاصل معیار واگرائی KL خواهد بود. بنابراین رابطه معیار واگرائی KL را به صورت رابطه ۱۷-۸ تعریف می کنیم.

$$D_{kl}(Q_{(x)} \,||\, P_{(x)}) = KL(Q_{(x)} \,||\, P_{(x)}) = \sum_{i=1}^{n} P_{(x_i)} \log(\frac{Q_{(x_i)}}{P_{(x_i)}}) \tag{۸-۱۷}$$

معیار واگرائی KL با توجه به استفاده از لگاریتم در رابطه آن و سهولت محاسبات مربوط به مشتق آن به طور گسترده در مسائل مختلف به عنوان تابع هزینه به کار برده می شود. در صورتی که توزیع هدف دارای پراکندگی بالا، واریانس بزرگ، و توزیع ورودی متمرکز، دارای واریانس کوچک، باشد، در معیار KL، توزیع $P_{(x)}$ را به عنوان توزیع ورودی و توزیع $Q_{(x)}$ را به عنوان توزیع

[1] f-divergence
[2] Kullback–Leibler divergence

هدف در نظر می گیریم که به این تعریف، معیار KL پیشرو[1] اطلاق می شود. همچنین در صورتی که توزیع ورودی دارای پراکندگی بالا و توزیع هدف متمرکز باشد، توزیع $Q_{(x)}$ را به عنوان توزیع ورودی و توزیع $P_{(x)}$ را به عنوان توزیع هدف در نظر می گیریم، به این تعریف نیز، معیار KL معکوس[2] گفته می شود.

۸.۲.۳.۳ معیار MMD[3]

معیارهای واگرائی که پیش از این معرفی کردیم نسبت به شکل کلی دو توزیع احتمالاتی، رابطه دو توزیع در صورتی که توزیع ها پارامتریک باشند، حساس بودند؛ این این معنی که مقدار بهینه این معیارها صرفا در صورتی که دو توزیع دقیقا مشابه هم باشند حاصل می شود، در حالی که در برخی از مسائل هدف نه تغییر شکل توزیع بلکه تغییر بازه تعریف آن است. معیار MMD یکی از معیارهایی است که برای این نوع مسائل کاربرد دارد. این معیار براساس فاصله بین نقطه میانگین دو توزیع تعریف شده و مقدار آن در صورتی که نقطه میانگین دو توزیع بر هم منطبق باشند برابر صفر خواهد بود. این معیار را می توان به دو حالت گسسته و پیوسته تعریف کرد. رابطه معیار MMD برای حالت گسسته مطابق رابطه ۱۸-۸ تعریف می شود. این معیار را می توان برای هر دو نوع توزیع های پارامتریک و غیرپارامتریک به کار برد.

$$MMD(P,Q) = \left\| \mathrm{E}_{\mathbf{x}\sim P}[\varphi(\mathbf{x})] - \mathrm{E}_{\mathbf{y}\sim Q}[\varphi(\mathbf{y})] \right\|_H \tag{۸-۱۸}$$

در رابطه ۱۸-۸، $\boldsymbol{x},\boldsymbol{y}$ بردار مقادیر نمونه های دو توزیع P,Q و $\varphi(x)$ کرنل تبدیل فضای هیلبرت[4]، $E(.)$ نماد امید ریاضی و H نماد کرنل تکثیر فضای هیلبرت[5] است. کرنل تکثیر فضای هیلبرت یک فضای هیلبرت برای توابع است که با برقراری شرط رابطه ۱۹-۸ تعریف می شود.

$$\| f - g \| \to \forall x : \left\| f_{(x)} - g_{(x)} \right\| \tag{۸-۱۹}$$

طبق رابطه ۱۹-۹ اگر نرم دو تابع f,g بسته باشد، مقدار کوچکی داشته باشد، به ازای هر نقطه تعریف شده برای این دو توابع نیز نرم بسته است. با توجه به رابطه ۱۹-۸ برای کرنل $\varphi(x)$ می توانیم روابط متعددی تعریف کنیم که مرسوم ترین آن ها تابع همانی $\varphi(x) = x$ است.

۸.۲.۳.۴ فاصله واسااشتاین[6]

فاصله واسااشتاین معیار دیگری است که برای تعیین فاصله بین دو توزیع احتمالاتی تعریف می

[1] Forward KL-divergence
[2] Forward KL-divergence
[3] Maximum mean discrepancy
[4] Hilbert space
[5] Reproducing kernel Hilbert space (RKHS)
[6] Wasserstein distance

شود. رابطه فرم گسسته فاصله واسااشتاین به صورت رابطه ۸-۲۰ تعریف می شود. در رابطه ۸-۲۰، $\boldsymbol{x},\boldsymbol{y}$ بردار مقادیر نمونه های دو توزیع P,Q، $E(.)$ نماد امید ریاضی، $d(\boldsymbol{x},\boldsymbol{y})$ فاصله بین دو نقطه $\boldsymbol{x},\boldsymbol{y}$ و p یک پارامتر اسکالر است. در صورتی که $p=2$ باشد فاصله d به صورت فاصله اقلیدسی تعریف می شود. معیار فاصله واسااشتاین بر خلاف معیار MMD شکل توزیع ها را نیز در محاسبه فاصله آن ها در نظر گرفته و از این جهت رفتاری مشابه معیارهای واگرائی که پیش تر بررسی کردیم، دارد.

$$WS_p(P,Q)=(\inf \mathbb{E}(d(\mathbf{x},\mathbf{y})^p))^{\frac{1}{p}} \tag{۸-۲۰}$$

۸.۲.۴ تابع هزینه بر پایه آموزش عاطفی[1]

آموزش عاطفی در تابع هزینه به صورت تعریف مقادیر مشتق خطا در کنار مقدار خطا تعریف می شود، برای مثال اگر تابع هزینه رابطه ۸-۶ را برای یک مرحله به روزرسانی به صورت رابطه ۸-۲۱ تعریف کنیم، اگر مشتق خطا را نیز وارد رابطه کنیم، رابطه این تابع هزینه بر پایه آموزش عاطفی به صورت رابطه ۸-۲۲ تعریف می شود.

$$E_{(k)}=\frac{1}{2n}\sum_{i=1}^{n}\left\|e_{i(k)}\right\|_2 \tag{۸-۲۱}$$

$$E_{(k)}=\frac{1}{2n}\sum_{i=1}^{n}\left\|R_{i(k)}\right\|_2=\frac{1}{2n}\sum_{i=1}^{n}\left\|k_1e_{i(k)}+k_2\dot{e}_{i(k)}\right\|_2 \tag{۸-۲۲}$$

مشتق خطای تعریف شده در رابطه ۸-۲۲ را به صورت رابطه ۸-۲۳ تعریف می کنیم.

$$\dot{e}_{(k)}=e_{(k)}-e_{(k-1)} \tag{۸-۲۳}$$

در رابطه ۸-۲۲ صرفا از مشتق اول خطا استفاده شده است، در صورتی که نسبت به مسئله می توان از مشتقات بالاتر نیز در کنار مشتق اول یا به عنوان جایگزین آن استفاده کرد.

۸.۳ معیارها

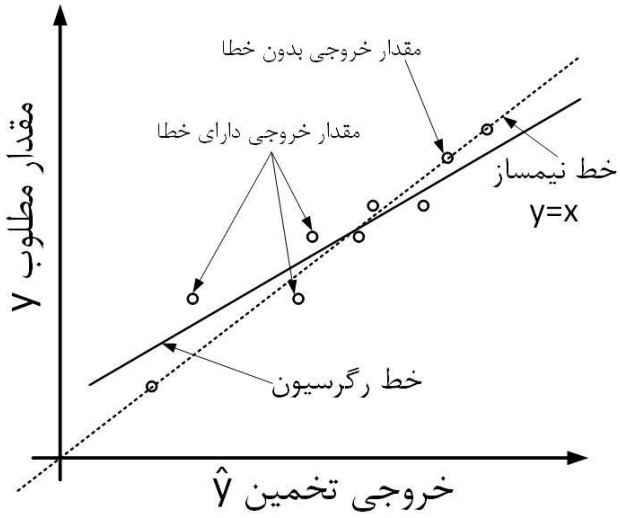
روابطی که بخش های گذشته به عنوان تابع هزینه معرفی کردیم اطلاعات شهودی خاصی از عملکرد مدل ارائه نمی کنند؛ اگرچه کاهش مقدار تابع هزینه نشانه آموزش و کاهش خطای مدل است ولی در نهایت مقدار این تابع گزارش مناسبی از میزان آموزش ارائه نمی کند. بدین منظور در کنار تابع هزینه ممکن است معیارهای دیگری نیز تعریف شود تا درک روشنی از وضعیت مدل در

[1] Emotional learning

هر گام ارائه کند. در این بخش به معرفی معیارهای مناسب برای مسائل مختلف حوزه یادگیری ژرف و یادگیری ماشین می پردازیم.

۸.۳.۱ خط رگرسیون

این معیار برای مسائلی که مدل آن ها دارای خروجی پیوسته هستند مانند مسئله رگرسیون و خود همبسته به کار برده شده براساس مقادیر مطلوب و خروجی مدل تعریف می شود. این معیار همواره برای یک بعد خروجی تعریف شده و اگر مدل دارای بیش از یک بعد خروجی داشته باشد خط رگرسیون برای هر بعد به طور جداگانه تعریف شده و عملکرد مدل را در آن بعد نشان می دهد. اگر از مقادیر خروجی و به مقادیر مطلوب متناظر با آن ها به ازای نمونه های مختلف یک تابع تعریف کنیم، $\hat{y}_i \rightarrow y_i$، در حالتی که خطای مدل صفر باشد، حالت ایده آل، مقدار خروجی و مطلوب با هم برابر بوده و رابطه خط افراز شده به مقادیر این تابع تعریف شده یک خط با شیب یک و عرض از مبدا صفر، $\hat{y}_i = y_i$، خواهد بود. برای محاسبه مقدار پارامترهای شیب و عرض از مبدا این خط از رابطه ۶-۹ فصل ۶ محاسبه می کنیم. فاصله مقادیر پارامترهای شیب و عرض از مبدا این خط از مقادیر ایده آل نشانه ضعف عملکرد مدل است. شکل ۳-۸ خط رگرسیون افراز شده به مقادیر خروجی و مطلوب یک مدل را نشان می دهد. خط رگرسیون در هر گام به ازای نمونه های آموزش و ارزیابی به طور مجزا تعریف می شود.



شکل ۳-۸: خط رگرسیون

میزان انحراف عرض از مبدا و شیب خط رگرسیون از خط نیمساز صرفا میزان بایاس شدن، کم برازش، مدل را نشان می دهد در صورتی که در یک مدل ممکن است خط رگرسیون به خط نیمساز

نزدیک باشد ولی میزان پراکندگی، واریانس، نقاطی که خط رگرسیون بر اساس آن ها محاسبه می شود بالا باشد، بیش برازش، بنابراین رسم مجموعه نقاط مطلوب نسبت به ورودی در نموداری مشابه شکل ۳-۸ علاوه بر محاسبه پارامترهای خط رگرسیون ضروری است.

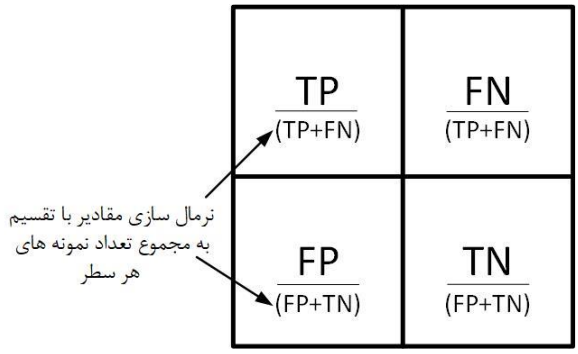
۸.۳.۲ ماتریس آشفتگی[1]

ماتریس آشفتگی معیاری است که برای مسائل طبقه بندی بر اساس کلاس نسبت داده شده توسط مدل به نمونه های دادگان تعریف می شود. برای تعریف ماتریس آشفتگی مسئله ای که دارای دو کلاس ناسازگار است مفاهیم زیر را تعریف می کنیم:

- **TP**[2]: تعداد نمونه های مربوط به کلاس اول، کلاس مثبت، که مدل کلاس آن ها را به درستی تشخیص داده است.
- **FN**[3]: تعداد نمونه های مربوط به کلاس اول، کلاس مثبت، که مدل کلاس آن ها را به اشتباه کلاس دوم، کلاس منفی، تشخیص داده است.
- **FP**[4]: تعداد نمونه های مربوط به کلاس دوم، کلاس منفی، که مدل کلاس آن ها را به اشتباه کلاس اول، کلاس مثبت، تشخیص داده است.
- **TN**[5]: تعداد نمونه های مربوط به کلاس دوم، کلاس منفی، که مدل کلاس آن ها را به درستی تشخیص داده است.

با استفاده از مفاهیم فوق ماتریس آشفتگی مربوط به مقادیر فوق را به صورت شکل ۴-۸ تعریف می کنیم. هر یک از درایه های این ماتریس یک مقدار اسکالر صحیح مثبت دارند، در صورتی که بخواهیم این مقادیر نسبت به تعداد نمونه ها نرمال سازی شده و به صورت درصد بیان شوند درایه های سطر اول را به تعداد کل نمونه های کلاس اول موجود و درایه های سطر دوم را به تعداد کل نمونه های کلاس دوم موجود در مجموعه دادگان آموزش یا ارزیابی که ماتریس برای آن تعریف شده است تقسیم می کنیم.

[1] Confusion matrix
[2] Ture positives
[3] False negatives
[4] False positives
[5] True negatives



شکل ۴-۸: ماتریس آشفتگی

ماتریس آشفتگی را می توان برای مسائل طبقه بندی شامل بیش از دو کلاس ناسازگار نیز تعمیم دهیم. برای مثال ماتریس آشفتگی یک مسئله طبقه بندی دارای n کلاس به صورت یک ماتریس مربعی $n\times n$ تعریف می شود؛ در این ماتریس درایه $(i\ i)$ تعداد نمونه های کلاس i-ام است که مدل آن ها را به درستی تشخیص داده است، درایه $(i\ j)$ نیز تعداد نمونه های کلاس i-ام است که مدل به اشتباه آن ها را کلاس j تشخیص داده است. برای نرمال سازی مقادیر این ماتریس و بیان آن ها به صورت درصد همه درایه های سطر i-ام به تعداد کل نمونه های کلاس i در مجموعه دادگان تقسیم می شوند. مفاهیم فوق الذکر که برای حالت دو کلاس ناسازگار معرفی کردیم را برای ماتریس آشفتگیی که بیش از دو کلاس دارد می توانیم در هر سطر به روش یک کلاس در برابر بقیه[1] تعریف کنیم. از روی ماتریس آشفتگی یک مدل می توان با توجه به تعریف مسئله معیارهای دیگری نیز توسعه داد که در اینجا به چند مورد از آن ها اشاره می کنیم:

۱. **معیار Accuracy:** این معیار به صورت میانگین قطر اصلی ماتریس آشفتگیی که مقادیر آن نرمال سازی شده است، به صورت درصد بیان شده است، تعریف می شود. رابطه این معیار برای حالت دو کلاس ناسازگار مطابق رابطه ۲۴-۸ است.

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \qquad (۸-۲۴)$$

۲. **معیار Recall (Sensitivity)**: این معیار به صورت نسبت درایه قطر اصلی در هر سطر به مجموع همه درایه های آن سطر تعریف می شود. رابطه این معیار برای هر کلاس مطابق رابطه ۲۵-۸ است. مقدار حاصل از این معیار مربوط به کلاسی است که درایه T-P آن در سطر مورد نظر از ماتریس بر روی قطر اصلی قرار گرفته است.

[1] One versus the rest

$$Recall = \frac{TP}{TP + FN} \tag{۸-۲۵}$$

۳. **معیار Precision:** این معیار به صورت نسبت درایه قطر اصلی در هر ستون به مجموع همه درایه های آن ستون تعریف می شود. رابطه این معیار برای هر کلاس مطابق رابطه ۸-۲۶ است. مقدار حاصل از این معیار مربوط به کلاسی است که درایه T-P آن در ستون مورد نظر از ماتریس بر روی قطر اصلی قرار گرفته است.

$$Precision = \frac{TP}{TP + FP} \tag{۸-۲۶}$$

۴. **معیار Specificity**: این معیار برای هر کلاسی که تعداد T-P مربوط به آن در درایه (i,i) قرار دارد به صورت نسبتی تعریف می شود که صورت آن مجموع مقادیر درایه های قطر اصلی به جز درایه (i,i) بوده و مخرج آن مقدار صورت کسر به علاوه مجموع درایه های ستونی که درایه (i,i) بر روی آن قرار دارد به جز درایه (i,i) تعریف می شود. رابطه این معیار برای حالت دو کلاس ناسازگار مطابق رابطه ۸-۲۷ است.

$$Specificity = \frac{TN}{TN + FP} \tag{۸-۲۷}$$

۵. **معیار F1-Score:** این معیار برای هر کلاس مطابق رابطه ۸-۲۸ تعریف می شود.

$$F1-score = \frac{2 \times Precision \times Recall}{Precision + Recall} \tag{۸-۲۸}$$

هر یک از معیارهای فوق می تولند دید مناسبی از عملکرد مدل طبقه بند با توجه به تعاریف مسئله ارائه کنند، بازه خروجی معیارهای فوق صفر تا یک بوده و مقدار بزرگ تر حاصل از این معیارها نشانگر آموزش مناسب و عملکرد مطلوب مدل است. علاوه بر این معیارها می توان معیارهای دیگری بر اساس ماتریس آشفتگی تعریف کرد. معیارهایی که بر اساس ماتریس آشفتگی به ازای هر کلاس تعریف می شوند را می توان با محاسبه میانگین آن به ازای همه کلاس ها برای به صورت کلی برای مسئله تعریف کرد.

۸.۳.۳ نمودار ROC[1]

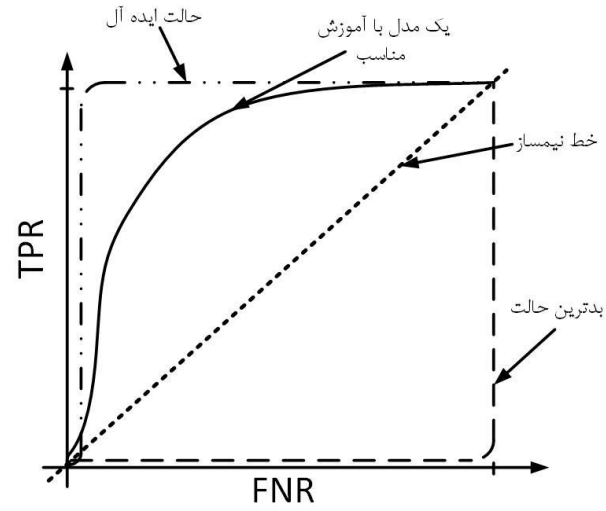
این نمودار معیار دیگری است که برای مدل های طبقه بند تعریف می شود. فرض کنید به ازای هر نمونه ورودی به مدل در گام آموزش یا ارزیابی بخواهیم ماتریس آشفتگی مدل را به ازای نمونه های وارد شده از ابتدا تا نمونه فعلی محاسبه و ذخیره کنیم. در این صورت به ازای هر نمونه ورودی یک ماتریس آشفتگی داریم که به ازای آن می توانیم معیار های مختلفی تعریف کنیم. برای هر یک از این ماتریس های آشفتگی دو معیار TPR[2]و FPR[3]را مطابق روابط ۲۹-۸ و ۳۰-۸ به ازای هر کلاس تعریف می کنیم. برای یک مدل با عملکرد مطلوب مقدار TPR باید بزرگ و مقدار FPR کوچک باشد. بازه عددی این دو معیار نیز بین صفر تا یک تعریف می شود.

$$TPR = Recall \tag{۸-۲۹}$$

$$FNR = 1 - Specificity \tag{۸-۳۰}$$

نمودار ROC یک کلاس با رسم مقادیر TPR و FPR آن کلاس در یک دستگاه مختصات دو بعدی به طوری که محور افقی مربوط به معیار و محور عمودی مربوط به محور عمودی است به ازای ماتریس های آشفتگیی حاصل از نمونه های مختلف حاصل می شود. منحنی ROC مربوط به یک کلاس اگر نزدیک به خط نیمساز باشد نشانه رشد یکسان پارامترهای TPR و FPR است. در یک مدل با آموزش مطلوب این منحنی در بالای خط نیمساز قرار می گیرد. شکل ۵-۸ نمایش یک نمودار ROC که در آن منحنی مربوط به یک کلاس فرضی و منحنی حالت ایده آل و منحنی بدترین حالت ممکن نشان داده شده است.

[1] Receiver operating characteristic
[2] True positive rate
[3] False negative rate



شکل ۵-۸: نمودار ROC

به دلیل حجم محاسبات بالا معمولا به ازای هر چند نمونه مقدار معیارهای روابط ۲۹-۸ و ۳۰-۸ محاسبه و مقدار آن ها در نمودار ROC نمایش داده می شود.

۸.۴ دسته بندی نمونه های مجموعه دادگان

در مطالب فصل ۶ بررسی کردیم که برای ارزیابی نمونه های مجموعه دادگان را به دو بخش نمونه های آموزش و ارزیابی تقسیم می کنند، بدین ترتیب هر گام در روند آموزش مدل شامل یک گام آموزش و یک گام ارزیابی می خواهد بود. در برخی موارد یک مجموعه نمونه دیگر از دادگان که به آن دادگان تست[1] گفته می شود، برای ساختار تعریف می شود تا بعد از اتمام روند آموزش مدل عملکرد نهائی مدل با استفاده از آن ها مورد تست قرار گیرد. روند اجرا شده برای دادگان تست، مشابه دادگان ارزیابی است که مطابق آن روابط پیشرو ساختار به ازای نمونه ها اجرا شده و بر اساس آن ها مقدار تابع هزینه و معیارهای مورد نظر محاسبه می شود. تفاوت دادگان ارزیابی و تست در این است که دادگان ارزیابی در هر گام، پس از اتمام گام آموزش به کار برده شده و کیفیت آموزش مدل را در طول روند آموزش با استفاده از آن بررسی می کنند، در حالی که دادگان تست صرفا یک بار پس از اتمام همه ی گام های آموزش و ارزیابی به کار برده می شود. از دادگان تست معمولا برای مقایسه چند مدل با هم استفاده می شود و در مواردی که صرفا یک مدل مورد بررسی است می توانیم از دادگان تست استفاده نکرده و عملکرد مدل را با استفاده از دادگان ارزیابی بررسی کنیم. در هر یک از مجموعه دادگان آموزش، ارزیابی و تست باید همواره توجه کنیم که نمونه های

[1] Test dataset

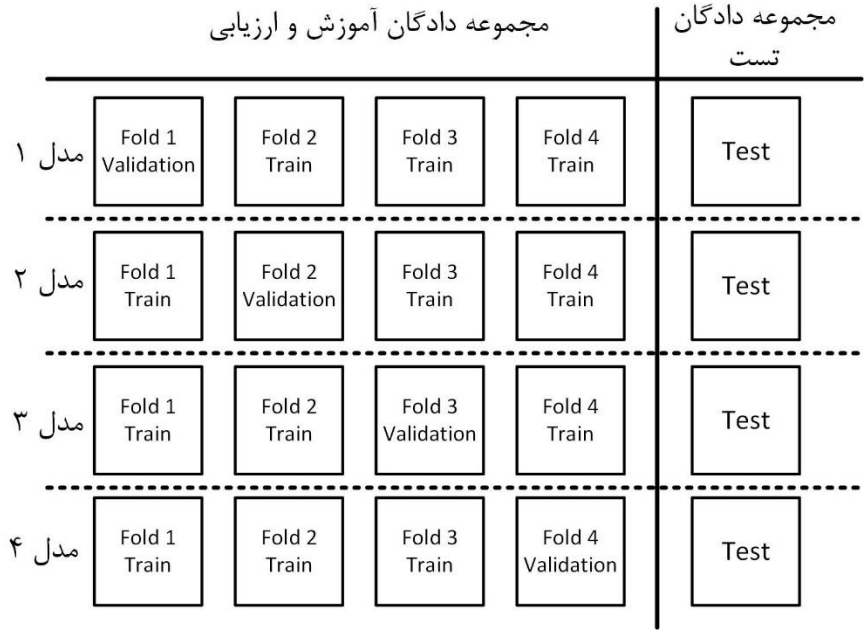
هر مجموعه همواره از هم تفکیک شده و هیچگاه نمونه های یک مجموعه وارد مجموعه دیگر نشود.

۸.۴.۱ ارزیابی متقابل K-fold[1]

اشـاره کردیم که مجموعه دادگان تعریف شـده برای یک مدل را به سـه زیر مجموعه، آموزش ارزیابی و تسـت تقسـیم می کنیم. در طی روند آموزش نمونه های هر یک از این سـه مجموعه ثابت است در حالی که ممکن است نمونه های ارزیابی اگر به عنوان نمونه آموزش به کار برده شوند مدل عملکرد متفاوتی داشـــته باشـــد، همچنین اگر از نمونه هایی که در مجموعه آموزش قرار دارند به عنوان نمونه ارزیابی اســـتفاده کنیم خروجی تابع هزینه و معیارهای ارزیابی متفاوت باشـــد. برای بررسی این مسئله در روند آموزش ساختار روش ارزیابی متقابل K-Fold را تعریف می کنیم. در این روش نمونه های آموزش و ارزیابی را به K مجموعه، دسـته[2]، با تعداد نمونه برابر تقسـیم می کنیم، تعداد نمونه های هر دســـته $\frac{N}{K}$ خواهد بود به طوری که N تعداد کل نمونه هایی اســـت که برای آموزش و ارزیابی مورد استفاده قرار می گیرد. سپس K مدل مجزا با ساختار یکسان تعریف کرده و هر یک از این مدل ها را با انتخاب K-1 دسـته از K دسـته ای که پیش تر تعریف کردیم به عنوان مجموعه آموزش و یک دســـته به عنوان مجموعه ارزیابی آموزش می دهیم، به طوری که دســـته ارزیابی هر یک از این K مدل با دسـته ارزیابی مدل های دیگر متفاوت باشـد. بدین ترتیب همه ی نمونه های دادگان یک بار به عنوان نمونه آموزش و یک بار به عنوان نمونه ارزیابی مورد اســـتفاده قرار گرفته اند. در این روش می توانیم وزن های اولیه هر K مدل را با یکسـان در نظر بگیریم، بدین ترتیب با ثابت بودن شـرایط اولیه صـرفا تاثیر دادگان در روند آموزش بررسـی می شـود، نکته حائز اهمیت در اینجا این است که هیچ گاه نباید وزن های حاصل از یک مدل را به عنوان وزن های اولیه مدل دیگر در نظر بگیریم زیرا این حالت بدین معنی است که از نمونه های آموزش به عنوان ارزیابی استفاده کرده ایم و ارزیابی مدل به درستی انجام نشده است. همان طور که اشاره کردیم نمونه های تســـت معمولا برای مقایســـه بین مدل های مختلف اســـتفاده می شـــود از این رو نمونه های این مجموعه در دسته ها دخالت داده نشده و از ابتدا از مجموعه دادگان آموزش و ارزیابی تفکیک شده و تغییری نمی کند. نسبت به عملکرد مدل با دادگان تست می توانیم هر یک از K مدل را به عنوان مدل نهائی تعریف کنیم. همچنین مقدار تابع هزینه و معیار کلی به ازای هر گام را می توانیم با محاســـبه میانگین این مقادیر به ازای همه مدل ها در آن گام بیان کنیم. شـــکل ۶-۸ نحوه دســـته بندی دادگان در روش ارزیابی متقابل K-Fold را نشـــان می دهد. برای بهبود عملکرد این روش ارزیابی از روش K-Fold طبقه بندی شـده[3] اسـتفاده می شـود؛ در این روش در مسـائل طبقه بندی تعداد نمونه های هر کلاس در هر دسته یکسان انتخاب می شود، در مسائل رگرسیون نیز محدوده عددی نمونه ها به بازه های یکسان تقسیم شده و در هر دسته تعداد از هر بازه تعداد نمونه یکسانی

[1] K-fold cross validation
[2] Fold
[3] Stratified

انتخاب شده و در دسته قرار می گیرد.



شکل ۶-۸: ارزیابی متقابل K-Fold

۸.۴.۲ ارزیابی متقابل Leave one out

این روش نیز برای ارزیابی متقابل دادگان با اســتفاده از همه نمونه ها به عنوان نمونه آموزش و ارزیابی تعریف می شود. روند کلی این روش مشابه روش ارزیابی متقابل K-Fold است با این تفاوت که در این روش تعداد دســته ها برابر با تعداد مجموع نمونه هایی که برای آموزش و ارزیابی در نظر گرفته شده است تعریف می شود. با این توضیحات روش ارزیابی متقابل Leave one out همان روش ارزیابی متقابل K-Fold اسـت که در آن دسـته ارزیابی همواره دارای یک نمونه اسـت. در این روش N مدل مجزا تعریف می شـود به طوری که N تعداد کل نمونه هایی اسـت که برای آموزش و ارزیابی مورد استفاده قرار می گیرد. با توجه به حجم محاسبات بالای این روش در مواردی که تعداد نمونه ها، N، بالا باشد استفاده از آن چندان مناسب نیست.

۸.۵ مسائل

۱. رابطه مشتق خروجی نسبت به ورودی توابع هزینه آنتروپی متقابل، معیار KL و فاصله واسااشتاین را محاسبه کنید.

۲. رابطه تابع هزینه بر پایه آموزش عاطفی، ۲۲-۸، را با اضافه کردن مشتق دوم خطا به آن بازنویسی کنید.

۳. به ازای مقادیر مختلف هر یک از معیارهایی که در بخش ۲-۳-۸ بر اساس ماتریس آشفتگی تعریف شده است، عملکرد مدل فرضی متناظر با آن مقادیر را تفسیر کنید.

۴. در مورد روش های یادگیری سنجه[1] تحقیق کنید.

[1] Metric learning

فصل ۹
روش های بهینه سازی[1] در یادگیری ژرف

۹.۱ مقدمه

در مطالب فصل ۶ روش پس انتشار خطا بر اساس گرادیان نزولی را برای آموزش شبکه های عصبی پرسپترون ارائه کردیم. در این فصل به بررسی ضعف های روش گرادیان نزولی پرداخته و روش های بهینه سازی که به عنوان جایگزینی روش گرادیان نزولی تعریف شده اند را معرفی و ویژگی های هر کدام از آن ها را بررسی می کنیم. نحوه به کارگیری این روش ها در روند آموزش ساختار نیز از دیگر موضوعات مورد بحث در این فصل است.

۹.۲ حالت های مختلف پیاده سازی روش های بهینه سازی

هر یک از روش های بهینه سازی که در بخش های بعدی معرفی خواهیم کرد را می توان در حالت کلی به سه حالت در روند آموزش یک ساختار پیاده سازی کرد که در ادامه آن ها را بررسی می کنیم. در این بخش این سه حالت را برای روش گرادیان نزولی بررسی می کنیم.

۹.۲.۱ روش تصادفی[2]

روش تصادفی همان روشی است که در فصل ۶ برای روش گرادیان نزولی که برای آموزش شبکه پرسپترون ارائه شده بود، معرفی کردیم. در روش های تصادفی مقادیر گرادیان در هر گام به آموزشی به ازای هر نمونه محاسبه شده و متعاقبا پارامترهای آموزشی ساختار برای به ازای هر نمونه ورودی به روزرسانی می شوند. رابطه ۵-۷ یا ۸-۷ فصل ۷ که برای روش گرادیان نزولی تعریف شده است را در نظر بگیرید، در صورتی که این روش به صورت تصادفی پیاده سازی شده باشد مقادیر k به ازای هر نمونه تعریف شده و به آن روش گرادیان نزولی تصادفی[3] اطلاق می شود.

---

[1] Optimization algorithms
[2] Stochastic
[3] Stochastic gradient descent (SGD)

۹.۲.۲ روش دسته ای[1]

در این روش در هر گام آموزشی مقادیر گرادیان پارامترهای آموزشی شبکه به ازای هر نمونه ورودی محاسبه شده و این مقادیر آن ها ذخیره می شود، در انتهای گام، پس از اتمام محاسبه گرادیان پارامترها به ازای همه نمونه های آموزشی، برای هر پارامتر مقدار میانگین گرادیان های آن به ازای همه نمونه های آموزشی بر اساس مقادیر گرادیان نمونه های آموزشی محاسبه می شود. سپس پارامترهای ساختار با استفاده از این مقدار میانگین گرادیان ها به روزرسانی می شوند. بدین ترتیب اگر روش گرادیان نزولی را به صورت دسته ای پیاده سازی کنیم، در روابط ۵-۷ یا ۸-۷، مقادیر k به ازای هر گام آموزشی تعریف می شوند. با توجه به استفاده از مقدار میانگین گرادیان ها در روش دسته ای، در صورت وجود نویز در تعدادی از نمونه های آموزشی، تاثیر آن ها در فرایند آموزش نسبت به روش تصادفی کمتر خواهد بود، از این رو روش دسته ای نسبت به روش تصادفی نسبت به نویز مقاوم تر است. اما از طرف دیگر استفاده از روش دسته ای با توجه به تعریف میانگین گرادیان ها برای همه نمونه های آموزش نسبت به روش تصادفی دقت کمتری داشته و همگرائی در آن به کندی صورت می گیرد، از این رو کاربرد این روش در ساختارهای امروزی محدود تر است.

۹.۲.۳ روش دسته ای کوچک[2]

روند اجرای این روش نیز مشابه روش دسته ای است با این تفاوت که در روش دسته ای کوچک، مقدار گرادیان میانگین به ازای تعدادی از نمونه های آموزشی محاسبه می شود در حالی که در روش دسته ای این مقدار به ازای همه نمونه ها محاسبه می شد. به تعداد نمونه هایی که در روش دسته ای کوچک مقدار گرادیان میانگین به ازای آن محاسبه می شود اندازه دسته[3] گفته می شود. در روش دسته ای کوچک اگر اندازه دسته یک باشد این روش همان روش تصادفی و اگر اندازه دسته برابر با تعداد کل نمونه های آموزشی باشد همان روش دسته ای محسوب می شود. تاثیر نویز احتمالی در تعدادی از نمونه ها و همگرائی این روش بین روش دسته ای و تصادفی است؛ میزان تاثیر نویز در آن کمتر از روش تصادفی و بیشتر از روش دسته و سرعت همگرائی آن نیز بیشتر از روش دسته ای و کمتر از روش تصادفی است. در این روش با تعریف اندازه دسته ای برابر با اندازه دسته آموزشی می توانیم مقدار تابع هزینه را به ازای هر دسته محاسبه کرده و با هم مقایسه کنیم. این روش در اکثر ساختار های امروزی به طور گسترده به کار برده می شود. با توجه به کاربرد محدود روش آموزش دسته ای در برخی منابع به روش دسته ای کوچک، روش دسته ای اطلاق می شود.

---

[1] Batch
[2] Mini batch
[3] Batch size

۹.۳ الگوریتم های بهینه سازی

در این بخش به بررسی سایر الگوریتم های بهینه سازی می پردازیم که برای به روزرسانی مقادیر پارامترهای یک ساختار تعریف می شوند. هر یک از این الگوریتم ها را می توان به سه روش فوق الذکر در ساختار پیاده سازی کنیم.

۹.۳.۱ گرادیان نزولی[1]

این روش مرسوم ترین روشی است که برای بهینه سازی بر اساس گرادیان خطا استفاده می شود. رابطه کلی الگوریتم مطابق رابطه ۱-۹ است. در رابطه ۱-۹، y,x به ترتیب ورودی و خروجی مطلوب ساختار، η نرخ آموزش، θ پارامتری آموزشی است که مقادیر آن با محاسبه گرادیان به روزرسانی می شود و $\nabla_\theta E(.)$ نماد گرادیان، مشتق، تابع هزینه نسبت به پارامتر است، که به صورت میانگینی به ازای نمونه i تا j نمونه محاسبه می شود. در رابطه ۱-۹ در صورتی که j برابر صفر باشد این روش، روش گرادیان نزولی تصادفی، در صورتی که i برابر یک و j برابر N به طوری که N تعداد کل نمونه های آموزشی باشد روش گرادیان نزولی دسته ای و در صورتی که $j = n < N$ باشد روش گرادیان نزولی دسته ای کوچک خواهد بود.

$$\theta_{(k+1)} = \theta_{(k)} - \eta \nabla_\theta E(\theta, x_{(i:j)}, y_{(i:j)})_{(k)} \qquad (۹-۱)$$

۹.۳.۱.۱ تکانه[2]

یکی از عمده مشکلاتی که در روش گرادیان نزولی بازه تغییرات گسترده اندازه و جهت بردار گرادیان در هر مرحله k است، که احتمال بروز پدیده زیگ زاگ و ناپایداری روند آموزش مدل را بالا می برد به خصوص در مواردی که نرخ آموزش مقدار بزرگی داشته باشد. این موضوع به ویژه در مواردی که برخی از نمونه های مجموعه دادگان به نویز آغشته باشند بیشتر نمود پیدا می کند. برای رفع این معضل در روش گرادیان نزولی، به جای استفاده از گرادیان یک پارامتر در آن مرحله برای به روزرسانی مقادیر آن پارامتر، از میانگین وزنی گرادیان آن مرحله و گرادیان مراحل قبل آن استفاده می کنند. رابطه کلی روش گرادیان نزولی با استفاده از تکانه مطابق روابط ۹-۲ و ۹-۳ است. در این روش با توجه به اینکه تاثیر گرادیان مراحل قبل در به روزرسانی بیشتر است، بردار مقادیر به روزرسانی نوسانات کمتری داشته و سرعت همگرائی ساختار در روند آموزش بالاتر خواهد بود.

$$\upsilon_{(k)} = \gamma \upsilon_{(k-1)} + \eta \nabla_\theta E(\theta, x_{(i:j)}, y_{(i:j)})_{(k)} \qquad (۹-۲)$$

[1] Gradient descent
[2] Momentum

$$\theta_{(k+1)} = \theta_{(k)} - \upsilon_{(k)} \quad (۹-۳)$$

در رابطه ۹-۲، η و γ پارامترهای اسکالر کنترلی هستند که مقادیر آن ها به ترتیب تاثیر جمله تکانه و گرادیان را مشخص می کند. مقدار پارامتر γ که به آن اندازه تکانه گفته می شود، معمولا نزدیک به ۰.۹ و مقدار پارامتر η که همان نرخ آموزش است نیز کوچکتر از ۰.۱ انتخاب می شود. در صورتی که مقدار پارامتر γ برابر با صفر باشد رابطه ۹-۱ حاصل می شود. در ابتدای شروع روند آموزش در مرحله $k = 1$، مقدار υ_{k-1} در رابطه ۹-۲ برابر صفر خواهد بود.

۹.۳.۲ NAG[1]

این روش با انتخاب دقیق تر بردار به روزرسانی نسبت به روش گرادیان نزولی سرعت همگرائی بالاتری دارد. در این روش هر مرحله به روزرسانی تقریبا معادل دو مرحله از روش گرادیان نزولی است. روند کلی در این الگورتیم به صورت روابط ۹-۴ تا ۹-۶ تعریف می شود، طبق این روابط ابتدا یک مقدار فرضی، $\theta'_{(k)}$، برای پارامترهای مدل با استفاده از مقدار فعلی پارامتر، $\theta_{(k)}$، و مقدار به روزرسانی مرحله قبل، $\theta_{update(k-1)}$، تعریف می کنیم. سپس با محاسبه گرادیان های این نقطه فرضی، $\nabla\theta'_{(k)}$، و مقدار به روزرسانی مرحله قبل، که معادل تکانه است، مقدار به روزرسانی مرحله فعلی را محاسبه می کنیم، $\theta_{update(k)}$، پس از محاسبه این مقدار با استفاده از آن مقدار پارامتر را به روزرسانی کرده و مقدار مرحله بعد، $\theta_{(k)}$، آن را محاسبه می کنیم.

$$\theta'_{(k)} = \theta_{(k)} - \gamma\theta_{update(k-1)} \quad (۹-۴)$$

$$\theta_{update(k)} = \gamma\theta_{update(k-1)} + \eta\nabla\theta'_{(k)} \quad (۹-۵)$$

$$\theta_{(k+1)} = \theta_{(k)} - \theta_{update(k)} \quad (۹-۶)$$

در برخی منابع روش بهینه سازی NAG را به صورت روابط ۹-۷ و ۹-۸ نیز نمایش می دهند. این روش را نیز می توان مانند سایر روش ها به صورت تصادفی، دسته ای و یا دسته ای کوچک پیاده سازی کنیم.

$$\upsilon_{(k)} = \gamma\upsilon_{(k-1)} + \eta\nabla_{\theta}E(\theta - \gamma\upsilon_{(k-1)}, x_{(i:j)}, y_{(i:j)})_{(k)} \quad (۹-۷)$$

$$\theta_{(k+1)} = \theta_{(k)} - \upsilon_{(k)} \quad (۹-۸)$$

۹.۳.۳ Adagrad[2]

در روش های بهینه سازی که پیش از این معرفی کردیم، مقدار نرخ آموزش برای همه

[1] Nesterov accelerated gradient
[2] Adaptive Gradient

پارامترهای ساختار به صورت یکسان تعریف می شود، در حالی که در روش Adagrad برای پارامترهایی که اندازه گرادیان آن ها در مراحل قبل بزرگ است، مقادیر نرخ آموزشی کوچکتری تعیین می شود تا بدین ترتیب از بروز پدیده زیگ زاگ و ناپایداری روند آموزش جلوگیری شده و سرعت همگرائی ساختار بهبود پیدا کند. روند کلی روش Adagrad به صورت رابطه ۹-۹ و تعریف می شود. در رابطه ۹-۹، G_θ مجموع مربعات گرادیان های پارامتر θ از مرحله اول تا مرحله کنونی است که طبق رابطه ۹-۱۰ تعریف می شود. در این روش مقدار نرخ رشد، η، معمولا برابر با ۰.۰۱ در نظر گرفته می شود. با توجه به جمع شدن مقدار گرادیان ها از مراحل قبل در مخرج و متعاقبا کوچک شدن متوالی مقدار کسر، ممکن است در مراحل پایانی این روش باعث ایجاد کسری با مقدار بسیار کوچک و تعداد ارقام اعشار بالا شده و در نهایت با ظاهر شدن مقادیر NaN در روابط روند آموزش شبکه مختل شود، این موضوع به خصوص در مواردی که اندازه گرادیان های مراحل اولیه بزرگ باشد اهمیت بیشتری پیدا می کند. در رابطه ۹-۹، ε یک مقدار کوچک است که برای جلوگیری از ایجاد کسر با مخرج صفر تعریف شده است.

$$\theta_{(k+1)} = \theta_{(k)} - \frac{\eta}{\sqrt{G_\theta + \varepsilon}} \nabla_\theta E(\theta, x_{(i:j)}, y_{(i:j)})_{(k)} \qquad (۹-۹)$$

$$G_\theta = \sum_{m=1}^{k} (\nabla_\theta E(\theta, x_{(i:j)}, y_{(i:j)})_{(m)})^2 \qquad (۹-۱۰)$$

۹.۳.۴ RMSprop[1]

این روش برای رفع معضل مربوط به رشد فزاینده مخرج کسر در روش Adagrad با توجه به جمع شدن متوالی مقادیر گرادیان مربوط به هر پارامتر تعریف شده است. در این روش به جای استفاده از مقدار مجموع مربعات مرحله قبل از میانگین وزنی گرادیان هر پارامتر که مطابق رابطه ۹-۱۱ تعریف می شود استفاده می کنیم.

$$\begin{aligned} &\mathrm{E}((\nabla_\theta E(\theta, x_{(i:j)}, y_{(i:j)})_{(k)})^2) = \\ &\gamma\, \mathrm{E}((\nabla_\theta E(\theta, x_{(i:j)}, y_{(i:j)})_{(k-1)})^2) + (1-\gamma)(\nabla_\theta E(\theta, x_{(i:j)}, y_{(i:j)})_{(k-1)})^2 \end{aligned} \qquad (۹-۱۱)$$

$$\begin{aligned} &\theta_{(k+1)} = \theta_{(k)} - \frac{\eta}{\sqrt{\mathrm{E}((\nabla_\theta E(\theta, x_{(i:j)}, y_{(i:j)})_{(k)})^2) + \varepsilon}} \nabla_\theta E(\theta, x_{(i:j)}, y_{(i:j)})_{(k)} = \\ &\theta_{(k)} - \frac{\eta}{RMS(\nabla_\theta E(\theta, x_{(i:j)}, y_{(i:j)})_{(k)})} \nabla_\theta E(\theta, x_{(i:j)}, y_{(i:j)})_{(k)} \end{aligned} \qquad (۹-۱۲)$$

[1] Root mean square propagation

با توجه به رابطه ۹-۱۱ در روش RMSprop بر خلاف روش Adagrad از پارامتر مستقل G_θ در محاسبات استفاده نشده و از این نظر حجم محاسبات و حافظه مورد نیاز برای این الگورتیم نسبت به الگوریتم Adagrad کمتر است. در روش RMSprop معمولا مقدار پارامتر γ برابر با ۰.۹ و پارامتر η برابر با ۰.۰۰۱ انتخاب می شود.

۹.۳.۵ Adadelta[1]

در روابطی الگوریتم های بهینه سازی که تا کنون معرفی کرده ایم مانند رابطه ۹-۱ الگوریتم گرادیان نزولی اگر برای مقدار فعلی پارامتر، $\theta_{(k)}$، یک واحد فرضی در نظر بگیریم، $Unit\ \theta$، واحد مربوط به تغییرات مقدار پارامتر که همان ضرب نرخ آموزش در گرادیان تابع هزینه نسبت به پارامتر است، $\Delta\theta_{(k)}$، برابر با $\frac{1}{Unit\ \theta}$ خواهد بود، بنابراین در این رابطه دو جمله با دو واحد متفاوت با هم جمع می شوند، این موضوع که در مورد سایر الگوریتم های بهینه سازی که پیش از این معرفی کردیم نیز صدق می کند که یکی از دلایل همگرائی کند و ناپایداری روند آموزش ساختار است. روش Adadelta برای رفع این معضل بر اساس الگوریتم RMSprop تعریف شده است، این روش برای رفع معضل مربوط به عدم تطابق واحد از روش نیوتن که یک روش مرتبه دوم بر اساس ماتریس هسیان[2] محسوب می شود استفاده می کند. در روش نیوتن مقدار تغییرات پارامتر رابطه ۹-۱۳ تعریف می شود. در رابطه ۹-۱۳، H ماتریس هسیان است. واحد فرضی رابطه ۹-۱۳ با واحد پارامتر θ برابر است.

$$\Delta\theta_{(k)} = H^{-1}\nabla_\theta E(\theta, x_{(i:j)}, y_{(i:j)})_{(k)} \qquad (۹-۱۳)$$

رابطه ۹-۱۳ را می توانیم به صورت رابطه ۹-۱۴ نیز نشان دهیم.

$$\Delta\theta_{(k)} = \frac{\overline{\frac{\partial E}{\partial\theta}}}{\overline{\frac{\partial^2 E}{\partial\theta^2}}}(k) \qquad (۹-۱۴)$$

بر اساس رابطه ۹-۱۴ می توانیم رابطه ۹-۱۵ را تعریف کنیم.

$$H^{-1} = \frac{\Delta\theta}{\nabla_\theta E} \qquad (۹-۱۵)$$

با جایگذاری رابطه ۹-۱۵ در رابطه ۹-۱۳ رابطه ۹-۱۶ را نتیجه می گیریم.

[1] An Adaptive Learning Rate Method
[2] Hessian matrix

$$\Delta\theta_{(k)} = \frac{\Delta\theta}{\nabla_\theta E}\nabla_\theta E(\theta, x_{(i:j)}, y_{(i:j)})_{(k)} \quad (۹-۱۶)$$

برای تعریف این الگوریتم بر اســـاس الگوریتم RMSprop باید رابطه ۹-۱۱ را بر اســـاس رابطه ۹-۱۶ برای الگوریتم Adadelta توسعه دهیم، این رابطه مطابق رابطه ۹-۱۷ است.

$$\mathrm{E}((\Delta\theta_{(k)})^2) = \gamma\,\mathrm{E}((\Delta\theta_{(k-1)})^2) + (1-\gamma)(\Delta\theta_{(k)})^2 \quad (۹-۱۷)$$

بدین ترتیب رابطه به روزرســـانی مقادیر پارامتر θ بر اســـاس الگوریتم Adadelta مطابق رابطه ۹-۱۸ تعریف می شـود. همان طور که از رابطه ۹-۱۸ نیز اسـتنباط می شـود این الگوریتم برای به روزرسانی مقادیر نیازی به پارامتر نرخ رشد ندارد.

$$\theta_{(k+1)} = \theta_{(k)} - \frac{RMS(\Delta\theta_{(k)})}{RMS(\nabla_\theta E(\theta, x_{(i:j)}, y_{(i:j)})_{(k)})}\nabla_\theta E(\theta, x_{(i:j)}, y_{(i:j)})_{(k)} \quad (۹-۱۸)$$

در رابطه ۹-۱۸ برای محاســـبه مقدار تغییرات پارامتر، $\Delta\theta_{(k)}$، مقدار $RMS(\Delta\theta_{(k)})$ مورد نیاز است و این موضوع باعث تناقض در این الگوریتم می شود. برای رفع این تناقض در رابطه ۹-۱۸ به جای $RMS(\Delta\theta_{(k)})$ از $RMS(\Delta\theta_{(k-1)})$ اســـتفاده شـــده و رابطه نهائی الگوریتم Adadelta مطابق رابطه ۹-۱۹ تعریف می شود.

$$\theta_{(k+1)} = \theta_{(k)} - \frac{RMS(\Delta\theta_{(k-1)})}{RMS(\nabla_\theta E(\theta, x_{(i:j)}, y_{(i:j)})_{(k)})}\nabla_\theta E(\theta, x_{(i:j)}, y_{(i:j)})_{(k)} \quad (۹-۱۹)$$

۹.۳.۶ Adam[1]

الگوریتم Adam یکی دیگر از روش های بهینه سازی است که علاوه بر استفاده از مقادیر میانگین مربعات گرادیان های مراحل گذشـــته مانند الگوریتم های RMSprop و Adadelta از میانگین گرادیان های مراحل گذشته نیز اسـتفاده می کند این بخش از این الگورتیم مشابه تعریف تکانه در الگوریتم گرادیان نزولی اســـت. رابطه به روزرســـانی مقادیر یک پارامتر، θ، طبق الگوریتم Adam مطابق رابطه ۹-۲۰ تعریف می شود.

$$\theta_{(k+1)} = \theta_{(k)} - \frac{\eta}{\sqrt{\hat{\upsilon}_{(k)}} + \varepsilon}\hat{m}_{(k)} \quad (۹-۲۰)$$

در رابطه ۹-۲۰، η نرخ رشد است که به صورت یک پارامتر اسکالر تعریف شده و ε مقدار اسکالر

[1] Adaptive moment estimation

کوچک برای جلوگیری از ایجاد مخرج صـــفر اســـت، همچنین در این رابطه $\hat{\upsilon}_{(k)}, \hat{m}_{(k)}$ به ترتیب مطابق روابط ۹-۲۱ و ۹-۲۲ تعریف می شوند.

$$\hat{m}_{(k)} = \frac{m_{(k)}}{1-\beta_1^k} \qquad (۹-۲۱)$$

$$\hat{\upsilon}_{(k)} = \frac{\upsilon_{(k)}}{1-\beta_2^k} \qquad (۹-۲۲)$$

در روابط ۹-۲۱ و ۹-۲۲ پارامتر های β_2, β_1 اســـکالرهایی با مقادیر مثبت برای تنظیم مقدار مخرج هستند. در این الگوریتم معمولا مقدار پارامتر β_1 برابر با ۰.۹، پارامتر β_2 برابر با ۰.۹۹۹ و نرخ رشد برابر ۰.۰۰۱ تعیین می شود. پارامترهای $m_{(k)}$ و $\upsilon_{(k)}$ نیز مطابق روابط ۹-۲۳ و ۹-۲۴ تعریف می شوند که به ترتیب میانگین گرادیان ها و توان دوم، واریانس، گرادیان های مراحل قبل هستند.

$$m_{(k)} = \beta_1 m_{(k-1)} + (1-\beta_1)\nabla_\theta E(\theta, x_{(i:j)}, y_{(i:j)})_{(k)} \qquad (۹-۲۳)$$

$$\upsilon_{(k)} = \beta_2 \upsilon_{(k-1)} + (1-\beta_2)(\nabla_\theta E(\theta, x_{(i:j)}, y_{(i:j)})_{(k)})^2 \qquad (۹-۲۴)$$

دلیل اسـتفاده از روابط ۹-۲۱ و ۹-۲۲ در رابطه به روزرسـانی این الگوریتم، رابطه ۹-۲۰، و عدم جاگذاری مسـتقیم مقادیر حاصـل از روابط ۹-۲۳ و ۹-۲۴ در رابطه ۹-۲۰ این اسـت که طبق نتایج تجربی در اکثر موارد در مراحل اولیه اجرای الگورتیم مقادیر روابط ۹-۲۳ و ۹-۲۴، مقادیری کوچک و نزدیک به صـفر هسـتند، بنابراین اگر از مقادیر این روابط مسـتقیما در رابطه ۹-۲۰ اسـتفاده شـود، تغییرات پارامتر θ ناچیز بوده و روند همگرائی کند خواهد بود از این رو با تعریف روابط ۹-۲۱ و ۹-۲۲ مقادیر حاصل از روابط ۹-۲۳ و ۹-۲۴ را به ویژه در مراحل اولیه اجرای الگوریتم افزایش داده و ســـرعت همگرائی ســـاختار به نقطه کمینـه تابع هزینـه به ازای تغییرات پارامتر θ را بهبود می بخشند.

۹.۳.۷ گرادیان نزولی طبیعی[1]

در الگوریتم هایی مانند گرادیان نزولی و NAG برای به روزرســـانی مقادیر پارامترها از مقدار گرادیان، مشـتقات مرتبه اول، تابع هزینه نسـبت به پارامتر اسـتفاده می شـود؛ این روش صـرفا براسـاس بردار شـیب تابع هزینه نسـبت به مقادیر پارامتر تعریف شـده و دربردارنده نواقصـی اسـت، مقدار نزدیک به صـفر گرادیان لزوما نشـانه همگرائی به نقطه کمینه سـراسـری و یا حتی یک نقطه کمینـه محلی نیســـتند، همچنین این مقادیر به ازای تغییرات کوچک پارامتر ممکن اســـت به طور گسـترده تغییر کنند از این رو علاوه بر افزایش احتمال ناپایداری روند آموزش دید محدودی نسـبت

[1] Natural gradient descent

به تغییر مقادیر پارامتر دارند. برای رفع مشـــکلات مربوط به گرادیان ها، الگوریتم هایی بر پایه تابع لاپلاســـین[1]، مشـــتقات مرتبه تابع هزینه نســـبت به مقادیر پارامتر مانند الگوریتم RMSprop و Adadelta توســـعه یافته اند که براســـاس روش نیوتن، رابطه ۱۳-۹، تعریف می شـــوند، با توجه به اسـتفاده از مشـتقات مرتبه دوم در این روش ها علاوه بر شـیب تابع هزینه نسـبت به مقادیر پارامتر، انحنای این تابع نیز در تعریف رابطه به روزرســـانی مقادیر ســـاختار دخیل بوده و از این نظر دید مناسـبی نسـبت به همگرائی سـاختار در مقایسـه با روش های مبتنی بر مشـتقات مرتبه اول دارند، حجم محاسبات بالا از معایب روش های مبتنی بر تابع لاپلاسین است.

راه حلی که برای رفع مشـکل هر دو روش مبتنی بر گرادیان و تابع لاپلاسـین ارائه شـده اسـت، استفاده از ماتریس اطلاعات فیشر[2]، مشتقات مرتبه دوم، تابع لگاریتم درست نمائی در تعریف رابطه به روزرسانی مقادیر پارامترها است. بدین ترتیب رابطه به روزسانی مقادیر پارامتر θ در روند آموزش ساختار مطابق رابطه ۲۵-۹ تعریف می شود.

$$\theta_{(k+1)} = \theta_{(k)} - \eta F^{-1}_{(\theta_{(k)})}.\nabla_{\theta} E(\theta, x_{(i:j)}, y_{(i:j)})_{(k)} \qquad (۹-۲۵)$$

در رابطه ۲۵-۹، $F_{(.)}$ ماتریس اطلاعات فیشـر اسـت، ماتریس اطلاعات فیشـر یک ماتریس مربعی $n \times n$، n برابر با تعداد پارامترهای مدل، اسـت که درایه های آن به صـورت امید ریاضـی مشـتقات مرتبه دوم تابع درســـت نمائی به ازای گرادیان های مراحل مختلف از ابتدای روند آموزش تا مرحله فعلی، k، مطابق رابطه ۲۶-۹ تعریف می شود، طبق رابطه ۲۵-۹ همه پارامترهای ساختار در تعریف درایه های ماتریس اطلاعات فیشر دخیل هستند.

$$F_{(\theta)i,j} = \mathrm{E}((\frac{\partial \log(P_{\mathrm{model}(\hat{y}|\boldsymbol{\theta},\mathbf{x})}}{\partial \theta_i})(\frac{\partial \log(P_{\mathrm{model}(\hat{y}|\boldsymbol{\theta},\mathbf{x})}}{\partial \theta_j})) \qquad (۹-۲۶)$$

در رابطه ۲۶-۹، $P_{model(.)}$ تابع درســـت نمائی مدل و $\hat{y}, \boldsymbol{\theta}, \boldsymbol{x}$ به ترتیب بردار ورودی، مجموعه کل پارامترهای مدل و خروجی مدل به ازای بردار ورودی است. ماتریس اطلاعات فیشر را می توانیم به صـورت ماتریس کواریانس گرادیان های تابع درسـت نمائی مدل به ازای مقادیر این گردایان ها از ابتدای روند آموزش تا مرحله فعلی، k، به صـورت رابطه ۲۷-۹ نیز تعریف کنیم. طبق مطالب فصـل ۸ تابع درسـت نمائی یک مدل می تواند با تابع هزینه تعریف شـده برای مسـئله برابر باشـد؛ مواردی مانند اســـتفاده از جمله تنظیم وزن ها در تابع هزینه و... از جمله مواردی اســـت که در آن ها تابع

[1] Laplacian function
[2] Fisher information matrix

هزینه و درست نمائی تعریف یکسانی ندارند.

$$F_{(\theta)} = Cov(\nabla_{\mathbf{\theta}} P_{\text{model}(\hat{y}|\mathbf{\theta},\mathbf{x})}) \quad (۹-۲۷)$$

به توجه به رابطه ۲۵-۹ در این الگوریتم برای به روزرسانی مقادیر پارامتر ها از نسبت گرادیان بر ماتریس اطلاعات فیشر استفاده می شود که به آن گرادیان طبیعی[1] گفته می شود، از این رو به این روش گرادیان نزولی طبیعی اطلاق می شود. استفاده از مشتقات مرتبه دوم تابع لگاریتم درست نمائی به کمک ماتریس اطلاعات فیشر در الگوریتم گرادیان نزولی طبیعی باعث می شود تا این الگوریتم دید کلی تری از روند همگرائی مدل به نقطه بهینه سراسری نسبت به سایر الگوریتم های مبتنی بر تابع لاپلاسین که پیش از این معرفی کردیم داشته باشد، از این رو سرعت همگرائی این الگوریتم بالاتر، احتمال ناپایداری روند آموزش کمتر و از طرفی حجم محاسبات آن نسبت به الگوریتم های مبتنی بر تابع لاپلاسین کمتر است.

در این بخش الگوریتم ها را به ترتیب تاریخ معرفی مطرح کردیم، بدین ترتیب هر یک از آن ها برای رفع نواقص و بهبود الگوریتم پیش از خود تعریف شده است. اما باید در نظر داشته باشیم که الگوریتم های جدیدتر لزوما عملکرد بهتری نسبت به روش های پیش از خود ندارند و هر یک از روش های معرفی شده در این بخش می تواند با توجه به ساختار و تعریف مسئله عملکرد بهتری نسبت به بقیه داشته باشند.

[1] Natural gradient

۹.۴ مسائل

۱. در مورد روش های آموزش دسته ای گوس-نیوتن[1] و لونبرگ مارکوآت[2] تحقیق کرده و مزایا و محدودیت های هر کدام از آن ها را توضیح دهید.

۲. روند الگوریتم های ذیل را شرح داده و بیان کنید که هریک از آن ها براساس کدام الگوریتم تعریف شده در این فصل توسعه پیدا کرده اند.

أ. Adamax

ب. Nadam

ج. AMSGrad

[1] Gauss-Newton
[2] Levenberg–Marquardt

فصل ۱۰
یادگیری بازنمائی[1]

۱۰.۱ مقدمه

در فصل ۷ ساختار مناسب برای دادگان یک شبکه عصبی پرسپترون چند لایه را معرفی کرده و همچنین در مورد چالش ها و مشکلات یادگیری شبکه های عصبی که از دادگان ناشی می شود بحث کردیم. از جمله چالش های معرفی شده، بعد بالای دادگان است که گاها باعث مشکلاتی در روند آموزش شبکه های عصبی می شود. برای رفع این مشکل معمولا از روش تحلیل مولفه های اصلی[2] و انواع منشعب از آن مانند تحلیل مولفه های اصلی با کرنل[3] که در فصل ۶ معرفی شده است، استفاده می شود؛ این روش معیاری برای تعیین میزان وابستگی خطی[4] هر بعد از دادگان نسبت به ابعاد دیگر بوده و می توان براساس آن بعدهایی را که وابستگی خطی آن ها نسبت به ابعاد دیگر بیشتر است، از همه نمونه های موجود در مجموعه دادگان، دادگان آموزش و سنجش، حذف کرد. استفاده از این روش که به عنوان یک روش کاهش ابعاد[5] بدون نظارت[6] محسوب می شود، به ویژه در مواردی که مجموعه دادگان دارای ابعادی باشد که وابستگی بالایی به یک یا چند بعد دیگر دارند بسیار موثر بوده و می تواند ابعاد زائد را از مجموعه دادگان حذف کند. اما استفاده از روش تحلیل مولفه های اصلی دارای مشکلاتی است که به چند مورد از مهم ترین آنها اشاره می کنیم:

۱. در مواردی که وابستگی خطی بالایی بین ابعاد مختلف مجموعه دادگان نباشد و همه ابعاد داده مستقل از هم باشند، استفاده از این روش چندان موثر نیست و ممکن است باعث حذف ویژگی های مهمی از داده ها شود که در روند آموزش شبکه بسیار موثر هستند.

۲. در مواردی ممکن است میزان وابستگی خطی دو بعد، مولفه، تقریبا برابر باشد، ولی یکی از آن ها نسبت به دیگری اهمیت بیشتری داشته و استفاده از آن بعد در روند

---

[1] Representation learning
[2] PCA
[3] Kernel-PCA
[4] Linear correlation
[5] Dimension reduction
[6] Unsupervised

آموزش شبکه عصبی مناسب تر باشد. در چنین مواردی روش تحلیل مولفه های اصلی قادر به تعیین بعد مناسب از بین دو بعد فوق الذکر نیست.

۳. روند کاهش ابعاد و انتخاب بعدهایی که استقلال خطی بیشتری نسبت به سایر ابعاد دارند به صورت گسسته است؛ بدین معنی که کاربر پس از اعمال روش تحلیل مولفه های اصلی تعداد مشخصی از ابعاد که دارای مقادیر ویژه بزرگتری نسبت به بقیه ابعاد در ماتریس کواریانس[1] هستند را انتخاب و سایر ابعاد به کلی حذف شده و تاثیری در روند آموزش شبکه عصبی ندارند. در اینجا علاوه بر هدر رفت منابع استفاده مختلف شده برای جمع آوری داده های ابعادی که پس از اعمال روش تحلیل مولفه های اصلی حذف می شوند، اگر دو بعد در شرایط ذکر شده در مورد ۲ را داشته و در حد آستانه[2] انتخاب قرار گیرند؛ برای مثال پس از اعمال روش تحلیل مولفه های اصلی و مرتب سازی ابعاد بر اساس اندازه مقادیر ویژه ماتریس کواریانس آن ها، کاربر بخواهد تعداد n بعد از مجموع کل N بعد موجود در مجموعه دادگان را انتخاب کند، در این صورت ممکن است طبق مورد ۲ مقادیر ویژه بعد $n+1$ و n تقریبا با هم برابر باشند و بعدی که مقدار ویژه آن اندکی بزرگتر، $n+1$، است اهمیت بیشتری در روند آموزش شبکه داشته باشد ولی با اعمال این روش این بعد حذف خواهد شد.

۴. در مواردی که ابعاد مختلف در یک مجموعه داده دارای وابستگی محلی[3] هستند مانند دادگان تصویری و یا سری های زمانی، استفاده از روش تحلیل مولفه های اصلی ممکن باعث حذف ابعادی که به هم وابستگی محلی دارند می شود. برای مثال در صورتی که را در دادگان تصویری هر کدام از کانال های رنگی هر پیکسل[4] از تصویر معادل یک بعد از داده است، حال اگر فرض کنیم مسئله مدنظر آموزش یک شبکه عصبی طبقه بند برای تشخیص قرار داشتن یا فقدان یک جسم خاص در تصویر بر اساس دادگان تصویری باشد. در این صورت مجموعه پیکسل های مربوط به جسم مدنظر شباهت بالایی نسبت به هم داشته و وابستگی خطی نسبت به هم دارند از این رو با اعمال روش تحلیل مولفه های اصلی ممکن است تعداد قابل ملاحظه ای از پیکسل های جسم حذف شود در صورتی که دادهای مربوط به این پیکسل ها معرف ساختار جسم مدنظر در تصویر بوده و حذف آن ها ممکن است باعث بروز اختلال در روند آموزش شبکه عصبی شود.

[1] Covariance matrix
[2] Threshold
[3] Local correlation
[4] Pixel

۵. استفاده از روش تحلیل مولفه های اصلی بار محاسباتی[1] نسبتا بالایی داشته و پیچیدگی محاسباتی آن در بدترین حالت اگر m تعداد نمونه های مجموعه داده و n بعد نمونه ها باشد از مرتبه $O(mn^2 + n^3)$ است. این روش از دو قسمت اصلی محاسبه ماتریس کواریانس و محاسبه بردار و مقادیر ویژه تشکیل شده است که پیچیدگی محاسباتی هر قسمت به شرح زیر است:

- محاسبه ماتریس کواریانس: پیچیدگی محاسباتی اصلی در محاسبه ماتریس کواریانس مربوط به ضرب دو ماتریس با ابعاد $m \times n$ و $n \times m$ که شامل مقادیر نمونه های دادگان است. در این صورت این مرتبه پیچیدگی محاسباتی این عملیات در بدترین حالت برابر با $O(mn^2)$ است. در محاسبه پیچیدگی محاسباتی برای ماتریس کواریانس از سایر عملیات ها مانند تفاضل مقادیر نمونه ها از مقدار میانگین و تقسیم ضریب تصحیح بسل[2] به دلیل تاثیر ناچیز و مرتبه کوچکتر پیچیدگی آن ها نسبت به مرتبه پیچیدگی ضرب دو ماتریس صرف نظر می کنیم.

- محاسبه بردار و مقادیر ویژه: در محاسبه مقادیر ویژه ماتریس کواریانس محاسبه شده در مرحله قبل نیز بیشترین پیچیدگی محاسباتی مربوط به ضرب ماتریس کواریانس در خودش است که در بدترین حالت از مرتبه $O(n^3)$ است. در این قسمت نیز مشابه ماتریس کواریانس از سایر عملیات ها صرف نظر شده است.

بنابراین مرتبه پیچیدگی محاسباتی کلی برای این روش برابر با مجموع مرتبه پیچیدگی محاسباتی دو قسمت فوق، $O(mn^2 + n^3)$، است.

توضیحات فوق الذکر متمرکز بر چالش ابعاد بالای دادگان مورد استفاده در شبکه های عصبی است. در کنار این مورد چالش عمده دیگری که باعث اختلال در روند آموزش شبکه های عصبی شده و از دادگان ناشی می شود، فضایی است که دادگان در آن تعریف شده اند. گاهی ممکن است ابعاد مختلف یک مجموعه داده مورد استفاده مستقل خطی بوده و وابستگی چندانی نسبت به هم نداشته باشند ولی فضایی که توسط این ابعاد برازش[3] شده، فضای مناسبی نبوده و روند آموزش شبکه عصبی با استفاده از این دادگان خروجی مطلوبی نداشته باشد. در این صورت مجموعه دادگان باید به فضای دیگری با ابعاد برابر با ابعاد دادگان اصلی یا ابعادی متفاوت با آن، بالاتر یا پایین تر، نگاشت شوند.

این نگاشت مشابه با کرنل مورد استفاده در ماشین های بردار پشتیبان و همچنین روش تحلیل

---

[1] Computational complexity (CC): این روش در قسمت پیوست شرح داده شده است

[2] Bessel's correction

[3] Span

مولفه های اصلی با کرنل که هر دو در فصل ۶ معرفی شده اند عمل می کند. لازم به ذکر است که لایه های یک شبکه عصبی خود دارای نگاشت خطی و غیرخطی هستند؛ نگاشت خطی با اعمال وزن به ورودی هر لایه و استفاده از تابع فعال ساز خطی و نگاشت غیرخطی با اعمال توابع فعال ساز غیرخطی حاصل می شود. اما در مواردی نگاشت های موجود در لایه های شبکه عصبی کافی و مناسب نبوده و لازم است نگاشت هایی افزون بر لایه های شبکه عصبی پیش از استفاده از دادگان برای آموزش شبکه برآن ها اعمال شود تا عملکرد شبکه عصبی بهبود یابد. بدیهی است که در این چالش صرفا کاهش ابعاد نبوده و روش تحلیل مولفه های اصلی به عنوان رهیافت این چالش مطرح نمی شود.

چالش مطرح دیگر که از دادگان ناشی می شود احتمال وجود نویز[1] در دادگان است که می تواند عملکرد شبکه عصبی که با آن دادگان آموزش داده می شود را به طور چشمگیری کاهش دهد. الگوی رفتاری دادگان آغشته به نویز نسبت به حالت بدون نویز پیچیده تر است، از این رو آموزش شبکه عصبی پرسپترونی که پس از آموزش بتواند منحنی دادگان آغشته به نویز را شبیه سازی کند با چالش بیشتری نسبت به حالت بدون نویز همراه است، علاوه بر این موضوع احتمال بروز پدیده بیش برازش در دادگان آلوده به نویز بیشتر از حالت بدون نویز است، مطابق شکل ۸-۷ فصل ۷. بنابراین تمرکز بر توسعه روش هایی که تاثیر نویز دادگان را در روند آموزش شبکه کاهش دهد در کنار روش های مورد استفاده برای مرتفع کردن سایر چالش های ناشی از دادگان که پیش تر به آن ها اشاره شد دارای اهمیت است.

منظور از یادگیری بازنمائی آموزش و توسعه مدل هایی برای استخراج ویژگی است تا بر دادگان اصلی اعمال شده و پس از اعمال نگاشت های مناسب مشکلات فوق الذکر را که ناشی از دادگان هستند را مرتفع کنند. در این روش به جای حذف ابعاد، ویژگی ها، داده ها که حاصل روش تحلیل مولفه های اصلی است، ویژگی های جدیدی از ویژگی های دادگان اصلی استخراج می شوند. نمونه دادگان استخراج شده از این مدل های بازنمائی برای سایر مراحل از جمله آموزش شبکه های عصبی به جای نمونه های دادگان اصلی مورد استفاده قرار می گیرند.

در این فصل ابتدا به معرفی ماشین بولتزمن[2] و انواع آن پرداخته و سپس مدل های خودرمزگذار[3] و انواع آن را مطالعه می کنیم این دو مدل از مهم ترین مدل های مطرح در یادگیری بازنمائی هستند، همچنین رویکرد های مطرح شده در ماشین های بولتزمن و خودرمزگذارها برای مرتفع کردن هریک از چالش های مربوط به دادگان که پیش تر ذکر شد را بررسی می کنیم.

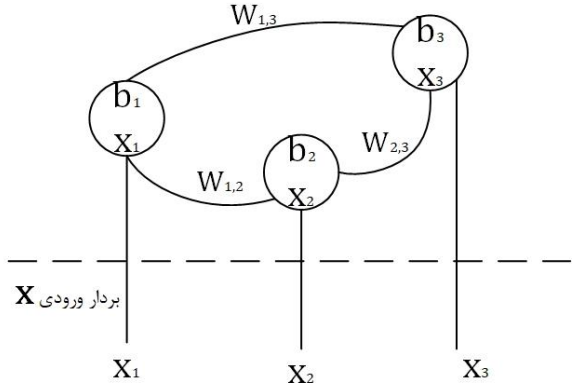
[1] Noise
[2] Boltzmann machine
[3] Autoencoder

۱۰.۲ ماشین بولتزمن

ماشین بولتزمن که به عنوان یک روش مولد[1] شناخته می شود، شبکه ای از n واحد است که به نورون های یک شبکه عصبی شباهت داشته و دو به دو به هم متصل هستند. هر یک از این واحد ها، $i \in \{1, \dots, n\}$، دارای یک متغیر تصادفی x_i هستند که این متغیر می تولند مقادیر باینری $x_i \in \{0,1\}$ را داشته باشد. همچنین هر یک از این واحد ها دارای یک بایاس b_i بوده و بین هر دو واحد، $i \in \{1, \dots, n-1\}, j \in \{1, \dots, n\}$، متصل به هم یک وزن $w_{i,j}$ قرار دارد. بنابراین می توان مجموعه متغیر های تصادفی واحد ها را با بردار متغیر های تصادفی، مجموعه بایاس های یک ماشین بولتزمن را به صورت بردار بایاس $\boldsymbol{b}$ و مجموعه وزن های بین واحد های متصل آن را با ماتریس وزن های W نشان داد. شکل ۱-۱۰ یک ماشین بولتزمن متشکل از سه واحد را نشان می دهد که متغیر های تصادفی، بردار بایاس و ماتریس وزن های آن به ترتیب در رابطه های ۱-۱۰ و ۲-۱۰ نشان داده شده است.

$$\boldsymbol{b} = \begin{bmatrix} b_1 \\ b_2 \\ \vdots \\ b_n \end{bmatrix} \tag{۱۰-۱}$$

$$W = \begin{bmatrix} w_{1,1} & \cdots & w_{1,n} \\ \vdots & \ddots & \vdots \\ w_{n,1} & \cdots & w_{n,n} \end{bmatrix}_{n \times n} \tag{۱۰-۲}$$



شکل ۱-۱۰: گراف ساختار ماشین بولتزمن

[1] Generative model

تابع انرژی یک ماشین بولتزمن مطابق رابطه ۳-۱۰ تعریف می شود که در آن x_i و x_j به ترتیب متغیر تصادفی واحد i و j، b_i بایاس واحد i و $w_{i\,j}$ وزن بین دو واحد i و j است. نماد θ زیروند تابع انرژی بیانگر مجموعه پارامتر های ماشین بوده و متشکل از کل بایاس ها و وزن های ماشین است که در رابطه ۴-۱۰ نشان داده شده است. برای محاسبه مقدار انرژی یک ماشین بولتزمن با مجموعه پارامترهای به ازای متغیر های تصادفی x_i ابتدا متغیر مربوط به هر واحد در بایاس متناظر خود ضرب شده و مجموع این حاصل ضرب ها محاسبه می شود، سپس مجموع حاصل ضرب های هر وزن در متغیر دو واحد متصل نیز حساب شده و در نهایت قرینه دو عبارت مجموع حاصل ضرب های بایاس ها در متغیر ها و وزن ها در متغیر ها با هم جمع و حاصل به عنوان انرژی ماشین به ازای متغیر های تصادفی x در نظر گرفته می شود.

$$E_{\theta(\boldsymbol{x})} = -\sum_{i=1}^{n} b_i x_i - \sum_{i=1}^{n-1}\sum_{j=i+1}^{n} w_{i,j} x_i x_j = -\boldsymbol{b}^T \boldsymbol{x} - \boldsymbol{x}^T W \boldsymbol{x} \qquad (۱۰-۳)$$

$$\boldsymbol{\theta} = \{b_1, \ldots, b_n, w_{1,2}, \ldots, w_{n-1,n}\} \qquad (۱۰-۴)$$

حال با داشتن تابع انرژی یک ماشین بولتزمن، توزیع احتمالاتی[1] مجموعه بردارهای متغیرهای تصادفی باینری را برای مجموعه دادگان آموزشی به صورت رابطه ۵-۱۰ تعریف می شود. پیش تر اشاره کردیم که ماشین بولتزمن یک روش مولد است، این مدل با استفاده از مجموعه پارامتر های $\boldsymbol{\theta}$ توزیع رابطه ۵-۱۰ را برای الگوهای باینری که به صورت بردارهای متغیر های تصادفی $\boldsymbol{x}$ ماشین بولتزمن است را شبیه سازی می کند. بدین ترتیب پس از آموزش و شبیه سازی توزیع ۵-۱۰ برای دادگان آموزشی از ماشین بولتزمن می توان برای تولید بردار هایی که از این توزیع پیروی می کنند استفاده کرد. این بردارهای تولید شده توسط ماشین ممکن است جز مجموعه دادگان آموزشی بوده و یا بردار جدیدی باشند که پیش تر در مجموعه دادگان آموزشی وجود نداشته است ولی در هر صورت از توزیع رابطه ۵-۱۰ تبعیت می کنند. بنابراین توزیع رابطه ۵-۱۰ توزیع بازنمائی دادگان آموزشی و مدل ماشین بولتزمن یک مدل بازنمائی برای دادگان خواهد بود. بدین ترتیب به جای استفاده از دادگان آموزش اصلی می توان از نمونه های تولید شده توسط ماشین بولتزمن به عنوان دادگان بازنمائی برای اهداف و مقاصد دیگر، آموزش سایر مدل‌ها و...، استفاده کرد.

$$P_{\theta(x)} = \frac{e^{-E_{\theta(x)}}}{\sum_{\tilde{x}} e^{-E_{\theta(\tilde{x})}}} \qquad (۱۰-۵)$$

[1] Probability distribution

از آن جایی که توزیع احتمالاتی حاصل از ماشین بولتزمن به پارامتر های ماشین وابسته بوده و با تغییر آن ها توزیع تغییر می کند، بنابراین در رابطه ۵-۱۰ مجموعه پارامتر های ماشین بولتزمن در صورت و مخرج کسر با هم برابر و ثابت است. در رابطه ۵-۱۰ مخرج کسر تابع پارتیشن نامیده می شود که در رابطه ۶-۱۰ نشان داده شده است، تابع پارتیشن برابر با مجموع مقادیر تابع نمائی به ازای قرینه انرژی ماشین بولتزمن برای همه ی بردار های متغیر تصادفی قابل تعریف، فضای نمونه‌ای، برای ماشین است، در این رابطه $\tilde{x}$ مجموعه تمامی مقادیر ممکن برای بردار متغیر تصادفی $\boldsymbol{x}$ است که در رابطه ۷-۱۰ نشان داده شده است. اگر تعداد واحد های ماشین بولتزمن n باشد در این صورت از آن جایی که هر واحد یک متغیر تصادفی x دارد اندازه بردار متغیرها برابر با تعداد واحد های ماشین است. از طرف دیگر این متغیر ها باینری بوده و فقط دو مقدار صفر یا یک دارند، بنابراین تعداد کل بردارهای متغیرهای تصادفی قابل تعریف که معادل تعداد جملات تابع پارتیشن است، برابر ۲^n است.

$$Z = \sum_{\tilde{x}} e^{-E_{\theta(\tilde{x})}} \tag{۱۰-۶}$$

$$\tilde{x} = \{x_1, ..., x_{2^n}\} \tag{۱۰-۷}$$

۱۰.۲.۱ تابع هزینه در ماشین بولتزمن

پیش تر اشاره کردیم که ماشین بولتزمن یک مدل مولد بوده و هدف آن شبیه سازی یک توزیع احتمالاتی است، بنابراین بدیهی است که خروجی مطلوب[1]، مشابه مقدار مطلوب در شبکه های عصبی پرسپترون چند لایه، برای یک ماشین بولتزمن نیز یک توزیع احتمالاتی باشد. مقادیر مجموعه پارامتر های ماشین بولتزمن باید به گونه ای باشد که توزیع احتمالاتی ماشین بولتزمن به توزیع مطلوب تا حد ممکن نزدیک بوده و دو توزیع خروجی ماشین و توزیع مطلوب بیشترین هم پوشانی را داشته باشند. برای سنجش عملکرد ماشین و وضعیت مجموعه پارامتر های آن نیازمند تابع هزینه ای هستیم که بتواند هم پوشانی و شباهت توزیع احتمالاتی حاصل از ماشین و توزیع احتمالاتی مطلوب را بسنجد، با توجه انواع توابع هزینه که در فصل بررسی کردیم مناسب ترین تابع هزینه برای این مسئله استفاده از تابع واگرایی KL است. رابطه ۸-۱۰ تابع هزینه واگرایی KL برای یک ماشین بولتزمن را نشان می دهد که در آن P_{target} توزیع احتمالاتی مطلوب برای مجموعه بردارهای متغیر های تصادفی و P_θ توزیع احتمالاتی حاصل از ماشین بولتزمن است.

$$KL(P_{\text{target}} \,||\, P_\theta) = \sum_{\tilde{x}} P_{\text{target}(\tilde{x})} \log \frac{P_{\text{target}(\tilde{x})}}{P_{\theta(\tilde{x})}} \tag{۱۰-۸}$$

[1] Target

رابطه ۸-۱۰ را می توانیم به شکل رابطه ۹-۱۰ نیز نشان دهیم. در رابطه ۹-۱۰ جمله اول رابطه مستقل از پارامتر های ماشین بوده و فقط جمله دوم با تغییر مقادیر مجموعه پارامترها تغییر می کند، بنابراین تابع هدف ماشین که با تنظیم مقادیر مجموعه پارامترهای ماشین باید آن را به حداکثر مقدار ممکن رساند همان جمله دوم رابطه ۹-۱۰ است که در رابطه ۱۰-۱۰ نشان داده شده است.

$$KL(P_{\text{target}} \parallel P_\theta) = \sum_{\tilde{x}} P_{\text{target}(\tilde{x})} \log P_{\text{target}(\tilde{x})} - \sum_{\tilde{x}} P_{\text{target}(\tilde{x})} \log P_{\theta(\tilde{x})} \quad (۱۰-۹)$$

$$f_{(\theta)} = \sum_{\tilde{x}} P_{\text{target}(\tilde{x})} \log P_{\theta(\tilde{x})} \quad (۱۰-۱۰)$$

با توجه به رابطه ۷ -۱۰ اگر توزیع احتمالاتی هدف $P_{target(x)}$ در رابطه ۱۰-۱۰ توزیع نمونه ای[1] باشد می توانیم رابطه ۱۰-۱۰ را به صورت رابطه ۱۱-۱۰ و رابطه ۱۲-۱۰ نیز نشان دهیم که در آن ها 2^n تعداد بردار های مجموعه $\tilde{x}$ که در رابطه ۷-۱۰ تعریف شده است.

$$f_{(\theta)} = \frac{1}{2^n} \sum_{\boldsymbol{x} \in \tilde{x}} \log P_{\theta(\boldsymbol{x})} \quad (۱۰-۱۱)$$

$$f_{(\theta)} = \frac{1}{2^n} \log \prod_{\boldsymbol{x} \in \tilde{x}} P_{\theta(\boldsymbol{x})} \quad (۱۰-۱۲)$$

رابطه ۱۲-۱۰ در واقع لگاریتم تابع درست نمائی[2] توزیع احتمالاتی مجموعه رابطه ۷-۱۰ به ازای مجموعه پارامترهای رابطه ۴-۱۰ ماشین است که می توانیم آن را به صورت رابطه ۱۳-۱۰ نشان دهیم.

$$P_{\theta(\tilde{x})} = \prod_{\boldsymbol{x} \in \tilde{x}} P_{\theta(\boldsymbol{x})} \quad (۱۰-۱۳)$$

۱۰.۲.۲ آموزش ماشین بولتزمن به روش قاعده هبیان[3] بر اساس گرادیان نزولی

آموزش ماشین های بولتزمن به روش قاعده هبیان تعریف می شود. این قاعده آموزش از نحوه رشد و تکامل سلول های زیستی الهام گرفته شده است. بر اساس این قاعده اتصالات بین نورونی، هر دو نورونی که تعداد دفعات بیشتری به طور همزمان فعال باشند قوی تر از سایر اتصالات خواهد بود. در ماشین های بولتزمن نیز وزن بین هر دو واحدی که به ازای ورودی های مختلف، تعداد دفعات بیشتری فعال باشد، متغیر هر دو واحد مقدار یک داشته باشد، مقدار بزرگ تری خواهد

[1] Empirical distribution
[2] Log likelihood
[3] Hebbian learning rule

داشت.

با تعریف تابع هزینه ماشین بولتزمن در رابطه برای آموزش مجموعه پارامترهای ماشین به روش قاعده هبیان بر اساس گرادیان نزولی باید ابتدا گرادیان تابع هزینه رابطه ۱۰-۱۰ را محاسبه کنیم. آموزش ماشین بولتزمن به روش گرادیان نزولی مشابه حالت کلی آموزش مدل های مولد به این روش بوده و گرادیان تابع هزینه ۱۰-۱۰ که نسبت به مجموعه پارامترهای ماشین محاسبه می شود مطابق رابطه ۱۰-۱۴ است.

$$\nabla f_{(\theta)} = \sum_{\tilde{x}} P_{\text{target}(\tilde{x})} \nabla \log P_{\theta(\tilde{x})} \tag{۱۰-۱۴}$$

گرادیان لگاریتم توزیع احتمالاتی مجموعه بردار متغیرهای $\tilde{x}$ رابطه ۱۰-۱۴ را به صورت روابط ۱۰-۱۵ تا ۱۰-۱۸ نمایش می دهیم:

$$\nabla \log P_{\theta(x)} = \nabla \log \frac{e^{-E_{\theta(x)}}}{\sum_{\tilde{x}} e^{-E_{\theta(\tilde{x})}}} \tag{۱۰-۱۵}$$

$$= -\nabla E_{\theta(x)} - \nabla \log \sum_{\tilde{x}} e^{-E_{\theta(\tilde{x})}} \tag{۱۰-۱۶}$$

$$= -\nabla E_{\theta(x)} + \frac{\sum_{\tilde{x}} e^{-E_{\theta(\tilde{x})}} \nabla E_{\theta(\tilde{x})}}{\sum_{\tilde{x}} e^{-E_{\theta(\tilde{x})}}} \tag{۱۰-۱۷}$$

$$= -\nabla E_{\theta(\tilde{x})} + \sum_{\tilde{x}} P_{\theta(\tilde{x})} \nabla E_{\theta(\tilde{x})} \tag{۱۰-۱۸}$$

طبق روابط ۱۰-۱۵ تا ۱۰-۱۸ می توانیم رابطه ۱۰-۱۴ را به صورت روابط ۱۰-۱۹ بازنویسی کنیم:

$$\nabla f_{(\theta)} = -\sum_{\tilde{x}} P_{\text{target}(\tilde{x})} \nabla E_{\theta(\tilde{x})} + \sum_{\tilde{x}} P_{\text{target}(\tilde{x})} \sum_{\tilde{x}} P_{\theta(\tilde{x})} \nabla E_{\theta(\tilde{x})} \tag{۱۰-۱۹}$$

با توجه به رابطه ۱۰-۲۰ رابطه ۱۰-۱۹ را به صورت رابطه ۱۰-۲۱ و ۱۰-۲۲ نشان می دهیم:

$$\sum_{\tilde{x}} P_{\text{target}(\tilde{x})} = 1 \tag{۱۰-۲۰}$$

$$\nabla f_{(\theta)} = -\sum_{\tilde{x}} P_{\text{target}(\tilde{\text{x}})} \nabla E_{\theta(\tilde{x})} + \sum_{\tilde{x}} P_{\theta(\tilde{\text{x}})} \nabla E_{\theta(\tilde{x})} \tag{۱۰-۲۱}$$

$$= -\sum_{\tilde{x}} (P_{\text{target}(\tilde{x})} - P_{\theta(\tilde{x})}) \nabla E_{\theta(\tilde{x})} \tag{۱۰-۲۲}$$

همچنین می توانیم رابطه ۱۰-۱۹ را به صورت رابطه ۱۰-۲۳ نیز نشان دهیم که در آن E_{target} امید ریاضی شرطی[1] نسبت به P_{target} و E_{θ} امید ریاضی شرطی نسبت P_{θ} است، طبق این رابطه گرادیان تابع هزینه ماشین بولتزمن تفاضل دو امید ریاضی مجموعه گرادیان است.

$$\nabla f_{(\theta)} = -\mathbb{E}_{\text{target}}(\nabla E_{(\boldsymbol{x})}) + \mathbb{E}_{\theta}(\nabla E_{(\boldsymbol{x})}) \tag{۱۰-۲۳}$$

با توجه به رابطه گرادیان ۱۰-۱۱ برای محاسبه پارامترهای گام $k+1$، گام بعدی، بر اساس گرادیان و پارامترهای گام k، گام فعلی، به روش گرادیان نزولی رابطه ۱۰-۱۲ را داریم که در آن η نرخ یادگیری است:

$$\theta_{(k+1)} = \theta_{(k)} + \eta \nabla f_{(\theta)(k)} \tag{۱۰-۲۴}$$

در رابطه ۱۰-۲۴، θ مجموعه پارامترهای ماشین بولتزمن متشکل از تمامی وزن ها و بایاس های ماشین است، بنابراین براساس رابطه ۱۰-۲۴ محاسبه مقادیر بردار بایاس ها و ماتریس وزن ها در گام آینده، $k+1$، به روش گردایان نزولی به ترتیب طبق روابط ۱۰-۲۵ و ۱۰-۲۶ است.

$$\boldsymbol{b}_{(k+1)} = \boldsymbol{b}_{(k)} + \eta \underbrace{\frac{\partial f_{(\theta)}}{\partial \boldsymbol{b}}(k)}_{-\boldsymbol{x}_{(k)}} \tag{۱۰-۲۵}$$

$$W_{(k+1)} = W_{(k)} + \eta \underbrace{\frac{\partial f_{(\theta)}}{\partial W}(k)}_{-\boldsymbol{x}_{(k)}{}^{T} \boldsymbol{x}_{(k)}} \tag{۱۰-۲۶}$$

۱۰.۲.۳ ماشین های بولتزمن با واحدهای پنهان[2] و نمایان[3]

تابع هزینه و روابط آموزشی طرح شده در بخش ۲-۲-۱۰ و ۳-۲-۱۰ با فرض ماشین بولتزمن با

[1] Conditional expectation
[2] Visible
[3] Hidden

واحدهایی که همه آن ها مشابه شکل ۱-۱۰ نمایان هستند است. اما ممکن است یک ماشین بولتزمن متشکل از واحد های پنهان نیز باشد که در این صورت روابط تابع انرژی، توزیع و آموزش آن متفاوت از حالتی است که ماشین فقط از واحد های نمایان تشکیل شده است. قبل از طرح روابط ماشین های بولتزمن با واحد های پنهان، ابتدا لزوم استفاده از واحد های پنهان و تفاوت آن ها را با واحد های نمایان را بررسی می کنیم.

۱۰.۲.۳.۱ لزوم استفاده از واحد های پنهان در ماشین های بولتزمن

یک ماشین بولتزمن با واحدهای نمایان مشابه شکل ۱-۱۰ را در نظر بگیرید. اندازه مجموعه پارامتر های این ماشین برابر با تعداد بایاس ها و وزن های ماشین است. از آن جایی که هر واحد ماشین دارای یک پارامتر بایاس است، بنابراین تعداد پارامترهای بایاس در یک ماشین با n واحد برابر با n است. از طرفی بین هر دو واحد از ماشین یک پارامتر وزن قرار دارد که از طریق آن این دو واحد به هم متصل می شوند، بنابراین تعداد وزن های ماشین متشکل از n واحد برابر با $\binom{n}{2}$ است. با این تفاسیر تعداد کل پارامتر های یک ماشین بولتزمن، اندازه مجموعه پارامترهای ماشین، از رابطه ۱۰-۲۷ محاسبه می شود.

$$n+\binom{n}{2}=n+\frac{1}{2}n(n-1)=\frac{1}{2}n(n+1) \qquad (۱۰-۲۷)$$

از طرفی هدف از آموزش یک ماشین بولتزمن مدل کردن یک توزیع احتمالاتی بر اساس مجموعه بردارهای متغیرهای تصادفی تی است، به عبارت دیگر هدف این است که ماشین برای هر یک از بردارهای متغیرهای تصادفی یک مقدار احتمال نسبت دهد. از آنجایی که مقادیر متغیرهای تصادفی این بردار ها باینری است، فقط دو مقدار صفر و یک برای آن های قابل تعریف است، بنابراین تعداد کل بردارهای قابل تعریف به اندازه n، اندازه بردار برابر با تعداد واحد های ماشین است، ۲^n است که همان تعداد جملات تابع پارتیشن ۶-۱۰ اشاره شده در قسمت ۲-۱۰ است. در حالت ایده آل برای مدل کردن یک توزیع احتمالاتی برای این مجموعه بردارها، تعداد پارامتر مورد نیاز مطابق رابطه ۱۰-۲۸ است.

$$2^n-1 \qquad (۱۰-۲۸)$$

رابطه ۱۰-۲۸ همان تعداد کل بردارهای متغیرهای تصادفی، تعداد نمونه ها، است که به آن ها پارامتر تصحیح بسل، در فصل ۳ مطرح شده است، که برای محاسبه واریانس یک جامعه بر اساس مجموعه نمونه هایی از آن جامعه استفاده می شود، اعمال شده است. در این حالت ایده آل به ازای هر بردار باینری قابل تعریف، پارامتر جداگانه ای برای ماشین در نظر گرفته شده است و مجموعه این پارامترها مقدار احتمال هر بردار را به آن اختصاص داده و توزیع احتمالاتی مجموعه بردار ها را مدل می کنند. با مقایسه رابطه ۱۰-۲۷ و ۱۰-۲۸ متوجه می شویم که به ازای هر n بزرگتر از دو،

$n > 2$، تعداد پارامترهای ماشین از تعداد پارامترهای ایده آل مطلوب برای مدل کردن توزیع احتمالاتی کمتر است.

برای رفع این معضل آسان ترین راه حل اضافه کردن پارامتر به ماشین بولتزمن است، اما از طرفی پارامترهای ماشین، وزن ها و بایاس ها، از واحدهای تشکیل دهنده ماشین ناشی می شوند و اضافه کردن پارامتر به ماشین مستلزم افزودن واحد به آن است، واحدهایی که هر کدام دارای یک متغیر تصادفی x هستند. در این صورت با افزودن واحد به ماشین اندازه بردارهای متغیرهای تصادفی نیز افزایش یافته و مجموعه متغیرهای تصادفی تغییر می کند. در این صورت شاید بتوان با ثابت در نظر گرفتن مقدار متغیر درایه، بعد، اضافه شده به بردار متغیرها برای همه ی بردارهای مجموعه متغیرها که از افزودن واحد به ماشین به وجود آمده است، چالش ناشی از این تغییر را تا حدودی مرتفع کنیم ولی باید در نظر داشته باشیم حتی در این حالت میزان افزایش پارامترهای مورد نیاز مطابق رابطه ۲۸-۱۰ بیشتر از افزایش تعداد پارامترهای ماشین، رابطه ۲۷-۱۰، است. بر این اساس افزودن واحد به ماشین باید به گونه ای باشد که باعث اضافه شدن پارامتر به ماشین بشود ولی اندازه بردارهای متغیرهای تصادفی را تغییر ندهد. این واحدها که واحدهای پنهان نامیده می شوند رفتاری مشابه با واحدهای نمایان دارند ولی متغیرهای تصادفی آن ها جز بردار متغیرهای تصادفی که هدف ماشین مدل سازی توزیع احتمالاتی آن‌ها است محسوب نمی‌شوند. بنابراین مجموعه بردارهای متغیرهای تصادفی مربوط به واحدهای پنهان را با نماد $\boldsymbol{h}$ نشان می دهیم. رابطه ۲۹-۱۰ این مجموعه بردارهای متغیرهای تصادفی واحد های پنهان را نمایش می دهد که در آن m تعداد واحدهای پنهان ماشین است. مقادیر متغیرهای واحدهای پنهان براساس مقادیر بردار ورودی و پارامترهای گام قبلی مطابق رابطه ۳۰-۱۰ محاسبه می شوند. رابطه ۳۰-۱۰ مقدار متغیر یک واحد پنهان را نمایش می دهد که با تعمیم آن به همه متغیرهای واحد های پنهان مجموعه ۳۰-۱۰ حاصل می شود. با توجه به تعریف مقادیر باینری برای متغیرهای واحدها در صورتی که مقدار حاصل از رابطه ۳۰-۱۰ بزرگتر ۰.۵ باشد مقدار متغیر واحد پنهان یک و در صورتی که در بازه صفر تا ۰.۵ باشد صفر خواهد بود. در رابطه ۳۰-۱۰، m تعداد واحدهای پنهان ماشین، $w_{i,j}$ وزن بین دو واحد i و j و b_i بایاس واحد i و تابع $f_{(.)}$ نیز یک تابع علامت مطابق رابطه ۳۱-۱۰ است. در محاسبه رابطه ۳۰-۱۰ دقت کنیم که مقادیر پارامترها و متغیرهای واحدهای پنهان از گام قبل و متغیرهای واحدهای نمایان از گام فعلی حاصل می شوند.

$$\tilde{h} = \{\boldsymbol{h}_1, ..., \boldsymbol{h}_{2^m}\} \qquad (۱۰-۲۹)$$

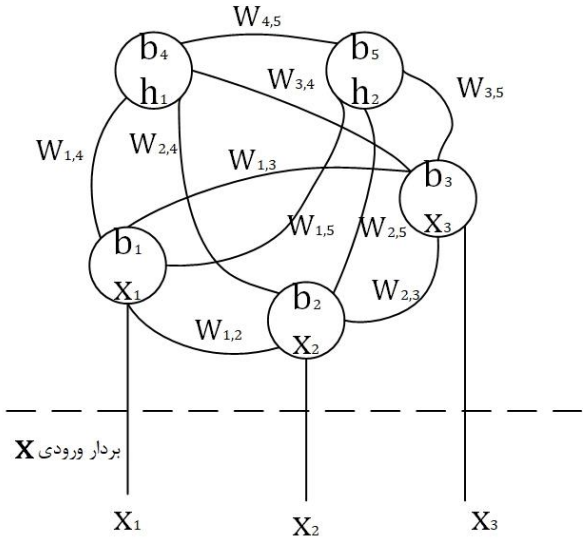
$$P_{h_j(k)} = f\left(\sum_{i=1}^{n} w_{i,j(k-1)} x_{i(k)} + b_{h_{j(k-1)}} + \sum_{i=1}^{m-1} w_{i,j(k-1)} h_{i(k-1)}\right) \qquad (۱۰-۳۰)$$

$$f(x) = \text{sign}(x) = \begin{cases} 1, & if \quad x > 0.5 \\ 0, & if \quad x \le 0.5 \end{cases} \qquad (۱۰-۳۱)$$

به روشی که برای به روزرسانی مقادیر متغیرهای واحدهای ماشین در رابطه ۳۰-۱۰ استفاده شده است به روزرسانی ناهمگام[1] اطلاق می شود؛ از این رابطه پس از آموزش یک ماشین بولتزمن برای تعیین مقادیر متغیرهای نمایان نیز می توان استفاده کرد؛ بدین صورت که پس از آموزش مقادیر متغیرهای واحدهای مختلف، اعم از پنهان و نمایان، ماشین را به صورت تصادفی تعیین کرده و سپس یک به یک مقادیر آن ها را با استفاده از رابطه ۳۰-۱۰ به روزرسانی می کنیم، این روند تا زمانی ادامه پیدا می کند که با اعمال رابطه ۳۰-۱۰ مقادیر متغیرهای واحدها تغییر نکند بدین ترتیب بردار حاصل از واحدهای نمایان ماشین یک نمونه از توزیع احتمالاتی ماشین خواهد بود، این مورد یکی از مهم ترین کاربردهای ماشین های بولتزمن بوده و از آن برای ایجاد نمونه هایی از توزیع های احتمالاتی غیرپارامتریک، توزیع آموزش داده شده به ماشین، استفاده می شود. علاوه بر روش اشاره شده برای به روزرسانی مقادیر متغیرهای ماشین با استفاده از رابطه ۳۰-۱۰ می توانیم مقادیر متغیرهای واحدها، پنهان و نمایان، در یک مرحله با استفاده از رابطه ۳۲-۱۰ به روزرسانی کنیم، مقادیر ابعاد مختلف بردار حاصل از رابطه ۱۰-۳۲ که هر کدام با یکی از واحدهای ماشین هستند در صورتی که بزرگتر ۰.۵ باشند مقدار متغیر آن واحد برابر با یک و در صورتی که کوچکتر از ۰.۵ باشند مقدار متغیر برابر با صفر خواهد بود، به این روش به روزرسانی مقادیر متغیرهای ماشین بولتزمن، به روزرسانی همگام[2] اطلاق می شود. در رابطه ۳۲-۱۰، $\boldsymbol{h}, \boldsymbol{x}$ به ترتیب بردار مقادیر متغیرهای واحدهای نمایان و پنهان، تابع $f_{(.)}$ یک تابع علامت مطابق رابطه ۳۱-۱۰، W ماتریس وزن های ماشین و b بردار بایاس های ماشین است. شکل ۲-۱۰ نیز نمایش گراف ساختار یک ماشین بولتزمن با واحدهای پنهان و نمایان است.

$$[\mathbf{x},\mathbf{h}]_{(k)} = f\left(W[\mathbf{x},\mathbf{h}]_{(k-1)} + \mathbf{b}^T[\mathbf{x},\mathbf{h}]_{(k-1)}\right) \qquad (۱۰-۳۲)$$

[1] Asynchronous
[2] Synchronous



شکل ۲-۱۰: ساختارماشین بولتزمن با واحدهای پنهان و نمایان

۱۰.۲.۳.۲ تابع هزینه در ماشین های بولتزمن با واحدهای نمایان و پنهان

تابع انرژی ماشین بولتزمنی با ساختار مشابه شکل ۲-۱۰ مطابق رابطه ۳۳-۱۰ است، که در محاسبه آن هر دو واحدهای نمایان و پنهان دخیل هستند.

$$E_{\theta(\boldsymbol{x},\boldsymbol{h})} = -\boldsymbol{b}^T \begin{pmatrix} \boldsymbol{x} \\ \boldsymbol{h} \end{pmatrix} - \begin{pmatrix} \boldsymbol{x}^T & \boldsymbol{h}^T \end{pmatrix} W \begin{pmatrix} \boldsymbol{x} \\ \boldsymbol{h} \end{pmatrix} \tag{۱۰-۳۳}$$

$$= -(\boldsymbol{b}^V)^T \boldsymbol{x} - (\boldsymbol{b}^H)^T \boldsymbol{h} - \boldsymbol{x}^T W^{VV} \boldsymbol{x} - \boldsymbol{x}^T W^{VH} \boldsymbol{h} - \boldsymbol{h}^T W^{HH} \boldsymbol{h} \tag{۱۰-۳۴}$$

طبق تابع انرژی رابطه ۳۳-۱۰ برای ماشین های بولتزمن با واحدهای پنهان توزیع احتمالاتی خروجی ماشین طبق رابطه ۳۵-۱۰ تعریف می شود.

$$P_{\theta(\boldsymbol{x},\boldsymbol{h})} = \frac{e^{-E_{\theta(\boldsymbol{x},\boldsymbol{h})}}}{\sum_{\tilde{x},\tilde{h}} e^{-E_{\theta(\tilde{x},\tilde{h})}}} \tag{۱۰-۳۵}$$

از آن جایی که هدف اصلی ماشین مدل کردن توزیع احتمالاتی برای متغیرهای تصادفی متناظر با متغیرهای واحدهای نمایان است، بنابراین رابطه توزیع حاشیه ای[1] ۳۶-۱۰ را براساس رابطه ۳۶-۱۰ تعریف می کنیم تا توزیع مدل شده برای متغیرهای تصادفی متناظر با واحدهای نمایان حاصل شود.

[1] Marginal probability distribution

$$P_{\theta(x)} = \sum_{\tilde{h}} P_{\theta(\boldsymbol{x},\tilde{h})} = \frac{e^{-E_{\theta(\boldsymbol{x},\tilde{h})}}}{\sum_{\tilde{x},\tilde{h}} e^{-E_{\theta(\tilde{x},\tilde{h})}}} \quad (۱۰-۳۶)$$

بر اساس صورت کسر رابطه ۱۰-۳۶، انرژی آزاد ماشین بولتزمن را به صورت رابطه ۱۰-۳۷ تعریف می کنیم:

$$F_{\theta(\boldsymbol{x})} = -\log \sum_{\tilde{\boldsymbol{h}}} e^{-E_{\theta(\boldsymbol{x},\tilde{h})}} \quad (۱۰-۳۷)$$

بر اساس رابطه انرژی آزاد ۱۰-۳۷، رابطه توزیع احتمالاتی ماشین بولتزمن با واحدهای پنهان و نمایان به صورت رابطه ۱۰-۳۸ تعریف می کنیم.

$$P_{\theta(x)} = \frac{e^{-F_{\theta(\boldsymbol{x})}}}{\sum_{\tilde{x}} e^{-F_{\theta(\tilde{x})}}} \quad (۱۰-۳۸)$$

۱۰.۲.۳.۳ آموزش ماشین بولتزمن با واحدهای پنهان و نمایان به روش گرادیان نزولی

با تعریف رابطه ۱۰-۳۸ تابع هزینه ماشین را مشابه حالت پیشین براساس تابع واگرایی KL تعریف می کنیم، بدین ترتیب گرادیان تابع هزینه یک ماشین بولتزمن با واحدهای پنهان و نمایان که مطابق رابطه ۱۰-۳۹ است که معادل رابطه ۱۰-۲۱ برای این ساختار است.

$$\nabla f_{(\theta)} = -\sum_{\tilde{x}} P_{\text{target}(\tilde{\text{x}})} \nabla F_{\theta(\tilde{x})} + \sum_{\tilde{x}} P_{\theta(\tilde{\text{x}})} \nabla F_{\theta(\tilde{x})} \quad (۱۰-۳۹)$$

گرادیان تابع انرژی آزاد در رابطه ۱۰-۳۹ برابر است با:

$$\nabla F_{\theta(\boldsymbol{x})} = -\nabla \log \sum_{\tilde{h}} e^{-E_{\theta(\boldsymbol{x},\tilde{h})}} \quad (۱۰-۴۰)$$

$$= \frac{\sum_{\tilde{h}} e^{-E_{\theta(\boldsymbol{x},\tilde{h})}} \nabla E_{\theta(\boldsymbol{x},\tilde{h})}}{\sum_{\tilde{h}} e^{-E_{\theta(\boldsymbol{x},\tilde{h})}}} \quad (۱۰-۴۱)$$

$$= \sum_{\tilde{h}} P_{\theta(\tilde{h}|\boldsymbol{x})} \nabla E_{\theta(\boldsymbol{x},\tilde{h})} \quad (۱۰-۴۲)$$

$P_{\theta(\boldsymbol{h}|\boldsymbol{x})}$ توزیع احتمالاتی شرطی[1] مقادیر بردار متغیرهای تصادفی واحدهای پنهان به شرط بردارهای متغیرهای تصادفی بردارهای واحدهای نمایان نسبت به مجموعه پارامترهای ماشین است که به صورت رابطه ۱۰-۴۳ تا ۱۰-۴۵ تعریف می شود.

$$P_{\theta(\boldsymbol{h}|\boldsymbol{x})} = \frac{e^{-E_{\theta(\boldsymbol{x},\boldsymbol{h})}}}{\sum_{\tilde{h}} e^{-E_{\theta(\boldsymbol{x},\tilde{h})}}} \tag{۱۰-۴۳}$$

$$= \frac{e^{-E_{\theta(\boldsymbol{x},\boldsymbol{h})}}}{\sum_{\tilde{x},\tilde{h}} e^{-E_{\theta(\boldsymbol{x},\tilde{h})}}} \frac{\sum_{\tilde{x},\tilde{h}} e^{-E_{\theta(\boldsymbol{x},\tilde{h})}}}{\sum_{\tilde{h}} e^{-E_{\theta(\boldsymbol{x},\tilde{h})}}} \tag{۱۰-۴۴}$$

$$= \frac{P_{\theta(\boldsymbol{x},\boldsymbol{h})}}{\sum_{\tilde{h}} P_{\theta(\boldsymbol{x},\tilde{h})}} = \frac{P_{\theta(\boldsymbol{x},\boldsymbol{h})}}{P_{\theta(\boldsymbol{x})}} \tag{۱۰-۴۵}$$

با جاگذاری روابط به دست آمده در رابطه ۱۰-۳۹ تابع هزینه ماشین به صورت رابطه ۱۰-۴۶ محاسبه می شود.

$$\nabla f_{(\theta)} = -\mathbb{E}_{\text{target}}[\mathbb{E}_{\theta}[\nabla E_{\theta(\boldsymbol{x},\boldsymbol{h})} \mid \boldsymbol{x}]] + \mathbb{E}_{\theta}[\nabla E_{\theta(\boldsymbol{x},\boldsymbol{h})}] \tag{۱۰-۴۶}$$

بدین ترتیب مجموعه پارامترهای ماشین در گام آینده از رابطه ۱۰-۴۷ محاسبه می شود که در آن η نرخ یادگیری است.

$$\theta_{(k+1)} = \theta_{(k)} + \eta \nabla f_{(\theta)}(k) \tag{۱۰-۴۷}$$

براساس رابطه ۱۰-۴۷ برای محاسبه مقادیر بردار بایاس و ماتریس وزن ها به ترتیب از روابط ۱۰-۲۱ و ۱۰-۲۲ استفاده می کنیم.

$$\boldsymbol{b}_{(k+1)} = \boldsymbol{b}_{(k)} + \eta \frac{\partial f_{(\theta)}}{\partial \boldsymbol{b}}(k) \tag{۱۰-۴۸}$$

$$W_{(k+1)} = W_{(k)} + \eta \frac{\partial f_{(\theta)}}{\partial W}(k) \tag{۱۰-۴۹}$$

[1] Conditional probability distribution

روابط ۴۸-۱۰ و ۴۹-۱۰ برای آموزش پارامترهای مختلف شامل بایاس واحدهای نمایان، بایاس واحدهای پنهان، وزن بین دو واحد پنهان، نمایان و پنهان-نمایان به کار برده می‌شوند.

۱۰.۲.۴ ماشین بولتزمن به عنوان یک روش متمایزکننده[1]

ماشین های بولتزمن به عنوان روش های مولد شناخته شده و هدف آن ها مدل سازی یک توزیع احتمالاتی براساس مجموعه الگوهای، بردارهای، متغیرهای تصادفی باینری است. در واقع این مجموعه الگوها به عنوان ورودی ماشین و توزیع مدل شده به عنوان خروجی که بر اساس ورودی ها ایجاد شده است، تعریف می شود. از آنجایی که مقادیر هر کدام از بردارهای ورودی در واحدهای نمایان متناظر خود وارد می شوند بنابراین می توان واحدهای نمایان را به عنوان واحدهای ورودی در نظر گرفت.

در فصل ۶ اشاره کردیم که هدف یک مدل متمایزکننده طبق رابطه بیز[2] مدل کردن یک توزیع احتمالاتی شرطی است؛ به طوری که پس از مدل سازی توزیع مدنظر، مدل بتواند بر اساس توزیع مدل شده احتمال شرطی هر ورودی را به ازای مقادیر مختلف خروجی محاسبه و در نهایت به هر ورودی، خروجی را که به ازای آن بیشترین مقدار احتمال شرطی حاصل شده است را به عنوان خروجی متناظر آن ورودی نسبت دهد. در رابطه ۵۰-۱۰ توزیع احتمالاتی شرطی که هدف مدل متمایزکننده مدل کردن آن است نشان داده شده است که در آن x و y به ترتیب بردارهای ورودی و خروجی مطابق روابط ۵۱-۱۰ و ۵۲-۱۰ هستند که در آن ها n اندازه بردار ورودی و m اندازه بردار خروجی است.

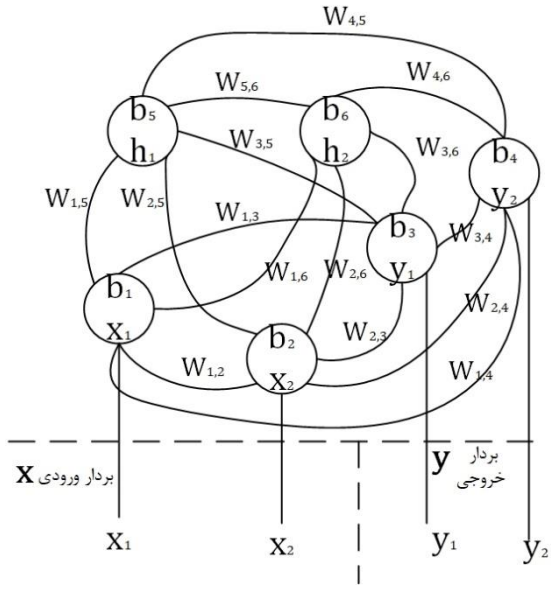
$$P_{\theta(x|y)} = \frac{P_{\theta(y,x)}}{P_{(y)}} = \frac{P_{\theta(y|x)}P_{(x)}}{P_{(y)}} \qquad (۱۰-۵۰)$$

$$\boldsymbol{x} = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix} \qquad (۱۰-۵۱)$$

[1] Discriminative model
[2] Bayes' theorem

$$\boldsymbol{y} = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{bmatrix} \tag{۱۰-۵۲}$$

در به رابطه ۱۰-۵۰ از آنجایی که هدف اصلی ماشین مدل کردن توزیع شرطی است به منظور ساده سازی مسئله برای توزیع های احتمالاتی $P_{(y)}$ و $P_{(x)}$ می توانیم یک توزیع فرضی در نظر بگیریم که با توجه به ماهیت باینری دادگان فرض توزیع یکنواخت برای آن های مناسب است. بنابراین هدف از آموزش مدل تعیین توزیع $P_{\theta(y|x)}$ ، تابع درست نمائی، خواهد بود.



شکل ۳-۱۰: ساختارماشین بولتزمن متمایزکننده

با توجه به این توضیحات یک ماشین بولتزمن برای این که به عنوان یک مدل متمایزکننده در نظر گرفته شود باید علاوه بر واحدهای ورودی، و در مواردی واحدهای پنهان، دارای واحدهای خروجی نیز باشد تا مقدار خروجی متناظر برای هر ورودی را محاسبه کند. بنابراین مطابق شکل ۳-۱۰ به اندازه بردار خروجی، واحدهای خروجی به ساختار ماشین بولتزمن اضافه می کنیم، از آن جایی که مقادیر متغیرهای تصادفی این واحدها با مقادیر بردار خروجی برابر است، این واحدهای خروجی به عنوان واحد نمایان در نظر گرفته می شوند به بیانی دیگر، در این ساختار واحدهای نمایان ماشین به دو بخش واحدهای ورودی و خروجی تقسیم می شود. بدیهی است که تعداد

واحدهای نمایان در این ساختار برابر مجموع لندازه بردار ورودی و بردار خروجی است. توزیع احتمال شرطی، تابع درست نمائی، رابطه ۵۰-۱۰ برای ساختار متمایزکننده با واحدهای پنهان مطابق رابطه ۵۳-۱۰ است.

$$P_{\theta(\boldsymbol{y}|\boldsymbol{x})} = \sum_{\tilde{h}} P_{\theta(\boldsymbol{y},\tilde{h}|\boldsymbol{x})} = \frac{\sum_{\tilde{h}} P_{\theta(\boldsymbol{x},\boldsymbol{y},\tilde{h})}}{\sum_{\tilde{y},\tilde{h}} P_{\theta(\boldsymbol{x},\boldsymbol{y},\tilde{h})}} \qquad (۱۰-۵۳)$$

۱۰.۲.۵ تابع هزینه در های ماشین بولتزمن متمایزکننده

با توجه به این که هدف آموزش ماشین بولتزمن متمایزکننده مدل سازی توزیع احتمالاتی شرطی رابطه ۵۳-۱۰، به طوری که مقدار احتمال نسبت داده شده به هر جفت نمونه ورودی و خروجی متناظر آن در داده های آموزشی حداکثر مقدار ممکن، به سمت مقدار یک میل کند، باشد، است. باید تابع هزینه تعریف شده برای این ساختار از ماشین های بولتزمن به طوری تعریف شود که شرایط زیر را داشته باشد:

- با میل کردن مقدار احتمال نسبت داده شده به هر جفت نمونه ورودی و خروجی متناظر با آن ورودی در داده های آموزشی، به سمت یک مقدار تابع هزینه کاهش یابد و بالعکس.
- با میل کردن مقدار احتمال نسبت داده شده به هر جفت نمونه ورودی و خروجی غیر از خروجی متناظر آن ورودی در داده های آموزشی، به سمت صفر مقدار تابع هزینه کاهش یابد و بالعکس.
- هنگامی که به همه ی جفت نمونه های ورودی و خروجی متناظر با آن ورودی در داده های آموزشی مقدار احتمال یک و به همه ی جفت نمونه های ورودی و خروجی غیر از خروجی متناظر با آن ورودی در داده های آموزشی مقدار احتمال صفر نسبت داده شود، تابع هزینه در نقطه کمینه خود قرار بگیرد.

با در نظر گرفتن موارد فوق و با توجه به مطالب فصل ۸، تابع هزینه ی مناسب برای ماشین های بولتزمن متمایزکننده لگاریتم تابع درست نمائی است که برای توزیع شرطی رابطه ۵۳-۱۰ ماشین تعریف می شود. این تابع هزینه مطابق رابطه ۵۴-۱۰ است که معادل روابط ۱۱-۱۰ و ۱۲-۱۰ برای ساختار متمایزکننده است و در آن N_D تعداد اعضای مجموعه D است که شامل همه ی جفت ورودی و خروجی هایی است که مدل متمایزکننده باید آن ها را به هم ربط دهد.

$$f_{(\theta)} = \frac{1}{N_D} \sum_{(\boldsymbol{x},\boldsymbol{y})\in D} \log P_{\theta(\boldsymbol{y}|\boldsymbol{x})} \qquad (۱۰-۵۴)$$

$$= \frac{1}{N_D} \log \prod_{(x,y)\in D} P_{\theta(y|x)} \quad (۱۰-۵۵)$$

$$D = \{(x_1, y_1), ..., (x_n, y_n)\} \quad (۱۰-۵۶)$$

مطلب حائز اهمیت در اینجا این است که ماشین بولتزمن در اصل یک مدل مولد بوده و تقسیم واحدهای نمایان آن به واحدهای ورودی و خروجی برای فرض آن به عنوان یک مدل متمایزکننده تغییری در اصل مولد ماشین ایجاد نمی کند. بر این اساس همچنان می توان تابع واگرایی KL را برای ماشین تعریف کرد. برای تعریف این تابع برای این ساختار توزیع احتمالاتی هدف و توزیع احتمالاتی ماشین را به صورت نشان داده شده در رابطه ۱۰-۵۳، توزیع احتمالاتی شرطی در نظر می گیریم. بنابراین تابع واگرایی ساختار به ترتیب زیر تعریف می شود.

$$KL(P_{\text{target}} \,||\, P_\theta) = \sum_{\tilde{x},\tilde{y}} P_{\text{target}(\tilde{x},\tilde{y})} \log P_{\text{target}(\tilde{y}|\tilde{x})} \ldots$$
$$\ldots - \sum_{\tilde{x},\tilde{y}} P_{\text{target}(\tilde{x},\tilde{y})} \log P_{\theta(\tilde{y}|\tilde{x})} \quad (۱۰-۵۷)$$

در رابطه ۱۰-۵۷ فقط جمله دوم وابسته به پارامترهای ماشین است بنابراین تابع هزینه ساختار براساس آن مطابق رابطه ۱۰-۵۸ تعریف می شود.

$$f_{(\theta)} = \sum_{\tilde{x},\tilde{y}} P_{\text{target}(\tilde{x},\tilde{y})} \log P_{\theta(\tilde{y}|\tilde{x})} \quad (۱۰-۵۸)$$

با فرض توزیع نمونه ای برای توزیع احتمالاتی هدف در رابطه ۵۶-۷ این رابطه را به صورت رابطه ۱۰-۵۹ و ۱۰-۶۰ بازنویسی می کنیم.

$$f_{(\theta)} = \frac{1}{N_D} \sum_{(x,y)\in D} \log P_{\theta(\tilde{y}|\tilde{x})} \quad (۱۰-۵۹)$$

$$f_{(\theta)} = \frac{1}{N_D} \log \prod_{(x,y)\in D} P_{\theta(\tilde{y}|\tilde{x})} \quad (۱۰-۶۰)$$

با مقایسه روابط ۱۰-۵۴ و ۱۰-۵۵ با روابط ۱۰-۵۹ و ۱۰-۶۰ می توان نتیجه گرفت که این روابط با هم برابر هستند و این موضوع اثبات توضیحات فوق مبنی بر عدم تغییر اصل مولد ماشین بولتزمن با وجود فرض مدل متمایزکننده برای آن است.

رابطه ۱۰-۵۸ را می توانیم به صورت روابط ۱۰-۶۱ تا ۱۰-۶۳ نیز نشان دهیم.

$$f_{(\theta)} = \sum_{\tilde{x},\tilde{y}} P_{\text{target}(\tilde{x},\tilde{y})} \log \frac{P_{\theta(\tilde{x},\tilde{y})}}{P_{\theta(\tilde{x})}} \quad (۱۰-۶۱)$$

$$= \sum_{\tilde{x},\tilde{y}} P_{\text{target}(\tilde{x},\tilde{y})} \log P_{\theta(\tilde{x},\tilde{y})} - \sum_{\tilde{x}} P_{\text{target}(\tilde{x})} \log P_{\theta(\tilde{x})} \quad (۱۰-۶۲)$$

$$= \mathbb{E}_{\text{target}}[\log P_{\theta(\boldsymbol{x},\boldsymbol{y})}] - \mathbb{E}_{\text{target}}[\log P_{\theta(\boldsymbol{x})}] \quad (۱۰-۶۳)$$

۱۰.۲.۶ آموزش ماشین بولتزمن متمایزکننده به روش قاعده هبیان بر اساس گرادیان نزولی

مشابه ساختارهای قبلی بر اساس رابطه، روابط مشتق برای آموزش ماشین بولتزمن متمایزکننده به روش گرادیان نزولی را محاسبه می کنیم. در جمله اول رابطه ۶۳-۱۰ واحدهای ورودی و خروجی به عنوان واحدهای نمایان و در جمله دوم واحدهای ورودی به عنوان واحدهای نمایان و واحدهای خروجی به عنوان واحدهای پنهان در روابط ظاهر می شوند، به عبارت دیگر برای محاسبه توزیع احتمالاتی P_y، در جمله اول توزیع اشتراک آن با ورودی ها $P_{(x,y)}$ محاسبه و در جمله دوم توزیع ورودی P_x از توزیع اشتراک جمله اول حذف می شود تا در نهایت توزیع واحدهای خروجی حاصل شود P_y. بنابراین هدف تابع هزینه رابطه ۶۳-۱۰ مدل کردن توزیع احتمالاتی برای واحدهای خروجی بوده و گرادیان آن به صورت رابطه ۶۴-۱۰ و ۶۵-۱۰ است.

$$\nabla f_{(\theta)} = \mathbb{E}_{\text{target}}[\nabla \log P_{\theta(\boldsymbol{x},\boldsymbol{y})}] - \mathbb{E}_{\text{target}}[\nabla \log P_{\theta(\boldsymbol{x})}] \quad (۱۰-۶۴)$$

$$\nabla f_{(\theta)} = \mathbb{E}_{\text{target}}[\mathbb{E}_{\theta}[\nabla P_{\theta(\boldsymbol{x},\boldsymbol{y})} \mid \boldsymbol{x},\boldsymbol{y}]] - \mathbb{E}_{\theta}[\nabla P_{\theta(\boldsymbol{x},\boldsymbol{y})} \mid \boldsymbol{x}] \quad (۱۰-۶۵)$$

بدین ترتیب برای محاسبه مقدار گام آینده بر اساس گام فعلی و گرادیان برای پارامترها به شرح ذیل است:

$$\theta_{(k+1)} = \theta_{(k)} + \eta \nabla f_{(\theta)}(k) \quad (۱۰-۶۶)$$

$$\boldsymbol{b}_{(k+1)} = \boldsymbol{b}_{(k)} + \eta \frac{\partial f_{(\theta)}}{\partial \boldsymbol{b}}(k) \quad (۱۰-۶۷)$$

$$W_{(k+1)} = W_{(k)} + \eta \frac{\partial f_{(\theta)}}{\partial W}(k) \quad (۱۰-۶۸)$$

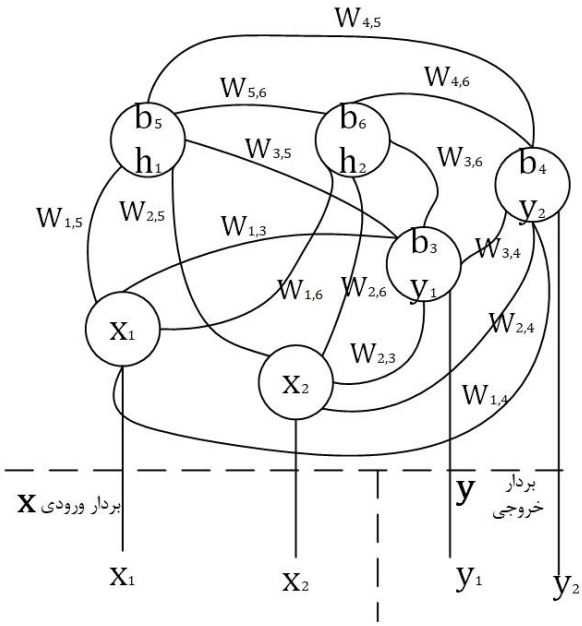
در روابط ۶۶-۱۰ تا ۶۸-۱۰ η نرخ یادگیری است.

با محاسبه روابط ۶۷-۱۰ و ۶۸-۱۰ برای وزن ها، بین دو واحد ورودی، ورودی و خروجی، پنهان و ورودی و...، و بایاس ها، واحدهای ورودی، پنهان و خروجی، مختلف مشاهده می کنیم که مقدار

بلیاس واحدهای ورودی و وزن های بین هر دو واحد ورودی در گام های مختلف ثلبت بوده و مقدار گرادیان آن ها نسبت به تابع هزینه صفر است، این موضوع را می توان با توجه به رابطه ۵۵-۱۰ که مجموعه دادگان آموزشی این ساختار را نشان می دهد و همچنین توزیع $P_{(y|x)}$ تابع هزینه نیز استباط کرد. ساختارهای پیشین توزیع $P_{(x)}$ را برای یک مجموعه بردارهای متغیرهای تصادفی در فضای نمونه ای متشکل از همه الگوهای قابل تعریف برای متغیرهای تصادفی مدل سازی می کردند و از این رو یادگیری آن ها از نوع بدون نظارت محسوب می شد در حالی که ساختار ماشین بولتزمن متمایزکننده توزیع احتمالاتی را مدل می کند که در آن به ازای هر بردار ورودی موجود در مجموعه رابطه ۵۵-۱۰ احتمال بردار خروجی متناظر با آن در فضای نمونه ای شامل همه خروجی های قابل تعریف به ازای آن ورودی حداکثر باشد، نزدیک به مقدار مطلوب یک باشد، و از این رو آموزش در این ساختار آموزش با نظارت[1] است. همچنین می توان این موضوع را با فرض توزیع یکنواخت برای توزیع های $P_{(y)}$ و $P_{(x)}$ و محدود کردن توزیع هدف آموزشی مدل که در ابتدای بخش بحث کردیم استنباط کرد.

بنابراین وزن های بین واحدهای ورودی و بلیاس های واحدهای ورودی در ساختار ماشین بولتزمن متمایزکننده زائد هستند زیرا با توجه به توضیحات فوق الگوی ورودی در مجموعه ۵۵-۱۰ تعریف شده و هدف مدل تعریف آن نیست، این هدف در ساختار های ماشین های بولتزمنی که پیش از این بررسی شدند دنبال می شد، بلکه مرتبط کردن آن با الگوی متناظر خود طبق مجموعه ۵۵-۱۰ است. ساختار ماشین بولتزمن متمایزکننده پس از حذف وزن های بین واحدهای ورودی و بایاس های ورودی مطابق شکل ۴-۱۰ است.

[1] Supervised



شکل ۴-۱۰: ساختار ماشین بولتزمن متمایزکننده بدون اتصال بین واحدهای ورودی

۱۰.۲.۷ مشکلات و چالش های آموزش ماشین های بولتزمن

آموزش ماشین های بولتزمن عموما مشکلات زیادی داشته و همگرایی مقدار تابع هزینه آن به مقدار کمینه بسیار کند است، عمده دلایل این مشکلات در روند آموزش ماشین های بولتزمن عبارت است از:

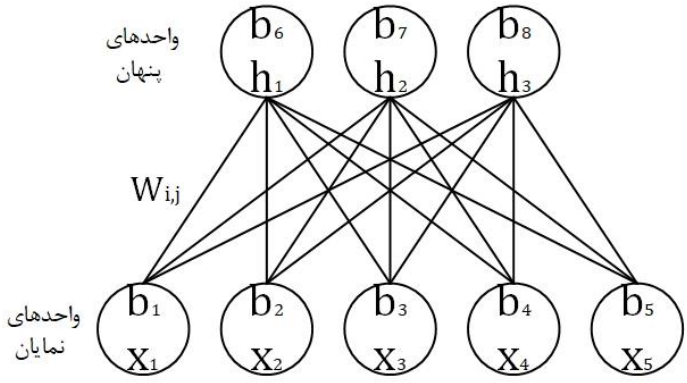
- با افزایش تعداد واحدهای پنهان برای بهبود عملکرد ماشین، تعداد پارامترهای آن افزایش می یابد. این پارامترها برای محاسبه توزیع احتمالاتی ماشین (روابطی مانند رابطه ۵-۱۰) به کار رفته و جملات رابطه این توزیع را می سازند. از طرفی تغییر مقادیر هر کدام از این پارامترها تاثیر مستقیم در توزیع احتمالاتی ماشین دارد. بنابراین آموزش و تعیین مقدار مناسب برای هر کدام از این پارامترها با افزایش تعداد آن ها مشکل تر شده و آموزش ماشین را با چالش روبرو می کند.
- طبق روابطی که پیشتر بررسی کردیم، تابع هزینه ماشین بولتزمن اختلاف دو مقدار میانگین شرطی است، با نظر به این که محاسبه میانگین از نمونه های دادگان مستعد عدم قطعیت است، بنابراین گرادیان حاصل از این تابع هزینه که آموزش و تغییر مقادیر پارامترهای ماشین را ممکن می سازد ممکن است عدم قطعیت بالائی داشته و مقادیر پارامتر های ماشین را در راستای کاهش تابع هزینه و همگرایی به مقدار کمینه سوق ندهد.

با توجه به مشکلات اشاره شده در فرایند آموزش ماشین بولتزمن، راهکارهای کلی برای بهبود

روند آموزش این ساختار های شامل مواردی چون، کاهش اتصال بین واحدها که منجر به کاهش پارامترهای وزن می شود، تغییر روش آموزش مانند ثلبت قرار دادن برخی پارامترها در هر گام و تغییر دادن بخش دیگری از آن ها در طول فرایند آموزش است. اما باید در نظر داشت که این راهکارها نباید باعث ایجاد ضعف در ساختار ماشین شده و قابلیت ماشین برای مدل کردن توزیع احتمالاتی هدف خود را کاهش دهد.

۱۰.۲.۸ ماشین های بولتزمن محدود شده[1]

با توجه به مشکلات اشاره شده در قسمت ۷-۲-۱۰ ماشین های بولتزمن محدود شده برای مرتفع کردن آموزش ماشین های بولتزمن ارائه شدند. ماشین های بولتزمن محدود شده به ساختاری از این ماشین ها اطلاق می شود که در آن ها اتصال بین جفت واحدهای نمایان و جفت واحدهای پنهان حذف شده و تنها جفت واحدهای نمایان-پنهان به هم متصل هستند. با این تغییرات در ساختار ماشین، ساختار آن مشابه شکل ۵-۱۰ بوده و شبیه به شبکه عصبی می شود.



شکل ۵-۱۰: ساختار ماشین بولتزمن محدود شده

در این ساختار از تعدادی واحد نمایان به طور مشترک به عنوان ورودی و خروجی استفاده نی شود. آموزش این ساختارهای نیز آموزش بدون نظارت بوده و هدف آن مدل کردن توزیع الگوهای ورودی است. در این ساختار نیز از تابع واگرایی KL برای مقایسه هم پوشانی توزیع دادگان آموزش و توزیع مدل مطابق رابطه ۸-۱۰ استفاده می شود. روند آموزش ماشین بولتزمن محدود شده به ترتیب زیر است:

۱. ابتدا برای هر یک از واحدهای پنهان یک مقدار احتمال بین صفر تا یک به صورت تصادفی در نظر گرفته می شود.

[1] Restricted Boltzmann machine (RBM)

۲. مقادیر بردار ورودی، بردار دارای مقادیر حقیقی نرمال سازی شده بین صفر و یک است، در متغیرهای واحدهای ورودی، نمایان، قرار گرفته و به با استفاده از رابطه ۶۹-۱۰ مقدار احتمال جدید برای هر یک از واحدهای پنهان محاسبه می شود. در رابطه ۶۹-۱۰ n تعداد واحدهای نمایان است و منظور از f تابع فعال ساز مشابه آنچه که در شبکه های پرسپترون چند لایه استفاده می شود، است که در این ساختار با توجه به این که هدف تولید مقدار احتمالی بین صفر تا یک است معمولا از توابع شعاعی یا سیگموئید استفاده می شود.

$$P_{h_j} = f(\sum_{i=1}^{n} w_{i,j} x_i + b_{h_j}) \qquad (۶۹-۱۰)$$

رابطه ۶۹-۱۰ مشابه رابطه محاسبه خروجی در نورون های شبکه های عصبی پرسپترون چندلایه است.

۳. در صورتی که مقدار احتمال محاسبه شده از رابطه ۶۹-۱۰ برای هر واحد پنهان از مقدار احتمال اولیه بزرگتر باشد، مقدار جدید به عنوان احتمال آن واحد در نظر گرفته شده و مقدار متغیر باینری آن واحد برابر یک، واحد فعال می شود، و در غیر این صورت مقدار احتمال واحد تغییری نکرده و متغیر تصادفی باینری آن برابر صفر، واحد غیرفعال باقی می ماند، می شود.

۴. در این مرحله بردار خروجی مدل محاسبه می شود، بدین ترتیب که براساس رابطه ۷۰-۱۰ و متغیرهای تصادفی واحدهای پنهان که در مرحله قبل محاسبه شد، متغیر هر واحد نمایان محاسبه می شود.

$$O_{x_j} = f(\sum_{i=1}^{m} w_{i,j} h_i + b_{x_j}) \qquad (۷۰-۱۰)$$

۵. با مقایسه مقدار خروجی واحدهای نمایان با مقادیر بردار ورودی، خطای بازسازی مدل محاسبه می شود، از آن جایی که مقادیر بین صفر تا یک نرمال سازی شده اند می توان این مقدار خروجی را یک توزیع احتمالاتی در نظر گرفته و با استفاده از تابع واگرایی KL رابطه ۹-۱۰ مقدار تابع هزینه مدل را محاسبه و با استفاده از گرادیان آن مطابق رابطه ۳۹-۱۰ مقادیر پارامترهای مدل در گام آینده را حساب کنیم. تابع هزنیه و روابط آموزشی در این ساختار مشابه ساختار ماشین بولتزمن با واحدهای نمایان و پنهان است که در بخش ۳-۲-۱۰ بررسی شده است با این تفاوت که در ساختار با واحدهای نمایان و پنهان فقط متغیر واحدهای پنهان براساس متغیرهای واحدهای

نمایان به روزرسانی شده و توزیع احتمالاتی رابطه ۳۶-۱۰ محاسبه می شد در حالی که در این ساختار براساس به روزرسانی متغیرهای پنهان، بردار ورودی نیز به روزرسانی شده و مقدار حاصل در رابطه ۳۶-۱۰ اعمال می شود، تفاوت دیگر دو ساختار استفاده از نگاشت غیرخطی در ساختار محدود شده است.

مقادیر متغیرهای تصادفی واحدهای پنهان را می توان همانند واحدهای نمایان مقادیر حقیقی در نظر گرفت در این صورت، مقادیر احتمال های اولیه که به هر واحد پنهان به صورت تصادفی نسبت داده شده‌اند به عنوان مقدار اولیه متغیر تصادفی واحد پنهان در نظر گرفته شده و در صورتی که مقدار حاصل از رابطه ۶۹-۱۰ از مقدار اولیه بزرگتر بود به عنوان مقدار متغیر تصادفی جدید واحد پنهان در نظر گرفته می شود. مقادیر حقیقی متغیرهای واحدهای پنهان در رابطه ۷۰-۱۰ به کار برده می شوند.

استفاده از مقادیر حقیقی به جای باینری در واحدهای پنهان باعث بهبود روند آموزش و عملکرد کلی ماشین می شود. در این صورت بردار پنهان نسبت به حالت متغیرهای باینری حاوی اطلاعات بیشتری است که از آن می توان به صورت توضیح داده شده در بخش ۹-۲-۱۰ استفاده کرد. از طرف دیگر متغیرهای باینری با اصل اولیه تعریف ریاضیاتی ماشین های بولتزمن که در آن مقادیر متغیرهای تصادفی باینری معرفی شده است سازگاری بیشتری دارد. استفاده از متغیرهای باینری نسبت به متغیرهای حقیقی فضای کمتری در حافظه سیستم های محاسباتی اشغال کرده و از نظر استفاده از منابع بهینه تر است.

همان طور که اشاره شد استفاده از متغیرهای حقیقی آموزش ماشین را بهبود می بخشد، اما در مواردی که دادگان را در بازه ای شامل اعداد مثبت و منفی، معمولا منفی یک تا یک، نرمال سازی کرده و از توابع فعال ساز با محدوده خروجی شامل بازه اعداد منفی؛ به طور مشخص منفی یک تا یک، مانند تانژانت هیپربولیک، استفاده کنیم، متغیرهای باینری واحدهای پنهان عملکرد بهتری نسبت به متغیرهای حقیقی دارند.

پس از آموزش ماشین بولتزمن محدود شده می توان از بردار متغیرهای پنهان آن که به ازای بردار ورودی به روز می شوند به عنوان بردار بازنمائی متناظر با ورودی استفاده کرد، در این صورت مدل ماشین بولتزمن محدود یک مدل بازنمائی خواهد بود.

۱۰.۲.۹ ماشین های بولتزمن محدود شده ژرف[1] و شبکه باور عمیق[2]

همان طور کردیم می توانیم ساختار ماشین بولتزمن را را به عنوان یک مدل بازنمائی در نظر گرفته و از بردار خروجی ساختار به عنوان بردار بازنمائی متناظر با بردارهای ورودی برای اهداف و کاربردهای دیگر استفاده کنیم. این بردارهای بازنمائی در ساختار ماشین بولتزمن متمایزکننده

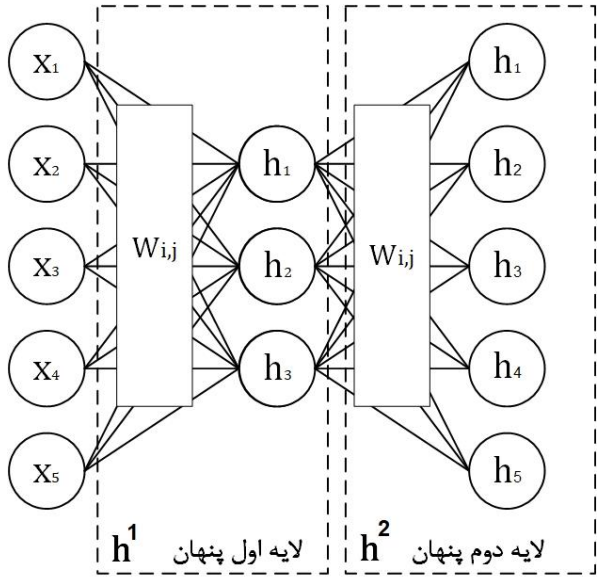
[1] Deep-RBM
[2] Deep belief networks (DBN)

نسبت به هر بردار ورودی تعریف می شدند در حالی که در ساختار کلی ماشین بولتزمن نمونه های توزیع احتمالاتی ماشین بودند.

در ماشین های بولتزمن محدود شده نیز می توان از بردار پنهان، متغیرهای واحدهای پنهان، منتاظر با هر بردار ورودی به عنوان بردار بازنمائی آن بردار ورودی به ویژه در حالتی که متغیرهای تصادفی واحدهای پنهان دارای مقادیر حقیقی باشند، استفاده کرد. بنابراین می توان این بردار پنهان را به عنوان نگاشتی از بردار ورودی به فضایی با بعدی برابر با تعداد واحد های پنهان در نظر گرفت، با این توصیف همچنین می توان با تنظیم تعداد واحد های پنهان از این روش به عنوان یک روش کاهش بعد نیز استفاده کرد که یکی از اهداف اصلی توسعه مدل های بازنمائی است.

از طرفی در فصل ۷ بررسی کردیم که افزودن لایه پنهان به یک شبکه عصبی پرسپترون می تواند باعث افزایش ظرفیت شبکه شده و عملکرد شبکه را به ویژه در مدل کردن الگوهایی با درجه غیرخطی بالا ارتقا دهد اما با افزودن لایه به شبکه مشکل محو شدگی گرادیان ها[1] شده و باعث می شود لایه های اولیه آموزش مناسبی نداشته باشند. این موضوع در مورد ماشین های بولتزمن محدود شده نیز صدق می کند، همان طور که در بخش ۱-۳-۲-۱۰ بررسی کردیم افزایش واحدهای پنهان عملکرد ماشین را بهبود می دهد. در ماشین های بولتزمن محدود شده که ساختار لایه ای شبیه به شبکه های عصبی داشته و اتصال بین واحدهای یک لایه در آن تعریف نمی شود این افزایش واحدهای پنهان به شکل اضافه شدن لایه در ساختار ظاهر می شود که به ساختار حاصل ماشین بولتزمن محدود شده عمیق اطلاق می شود. بنابراین توضیحات می‌توانیم ساختار یک ماشین بولتزمن محدود شده عمیق را به صورت شکل ۶-۱۰ تعریف کنیم. این ساختار یک ماشین بولتزمن محدود شده عمیق با دو لایه پنهان را نشان می دهد.

[1] Gradient vanishing



شکل ۶-۱۰: ساختار ماشین بولتزمن محدود شده عمیق

در ساختار ماشین بولتزمن محدود شده پس از به روز کردن بردار پنهان طبق رابطه ۹۷-۱۰ از مقادیر حاصل طبق رابطه ۷۰-۱۰ برای محاسبه بردار خروجی استفاده می شد در حالی که در ساختار ماشین بولتزمن محدود شده عمیق پس از به روزرسانی بردار پنهان لایه پنهان اول از این بردار حاصل از رابطه ۶۹-۱۰ طبق رابطه ۷۱-۱۰ برای به روزرسانی بردار پنهان لایه دوم استفاده می شود. پس از به روز رسانی لایه پنهان دوم از متغیرهای آن برای به روزرسانی مجدد لایه پنهان اول طبق رابطه ۷۲-۱۰ استفاده می شود. سپس از بردار لایه پنهان اول که مجددا به روزرسانی شده است برای به روزرسانی لایه ورودی، نمایان، طبق رابطه ۷۰-۱۰ استفاده می شود.

$$P_{h_j^2} = f(\sum_{i=1}^{n} w_{i,j} h_j^1 + b_{h_j}) \quad (۱۰-۷۱)$$

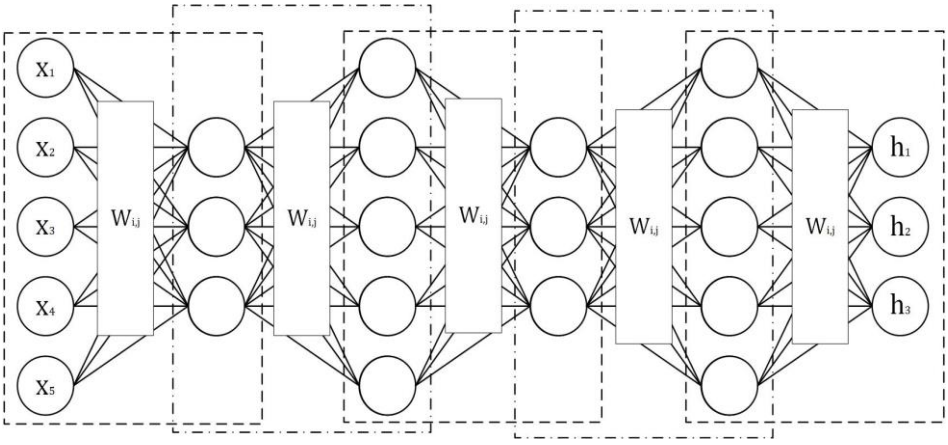
$$P_{h_j^1} = f(\sum_{i=1}^{n} w_{i,j} h_j^2 + b_{h_j}) \quad (۱۰-۷۲)$$

پس از محاسبه بردار به روزرسانی شده در لایه نمایان از روابط تابع هزنیه و آموزش برای ساختار ماشین بولتزمن با واحدهای پنهان و نمایان که در بخش ۳-۲-۱۰ مطرح شده است استفاده می کنیم. بدین ترتیب ساختار ماشین بولتزمن محدود شده عمیق مشابه یک ساختار ماشین بولتزمن با واحدهای پنهان و نمایان است که در آن مقادیر متغیرهای واحدها به صورت سلسله مراتبی محاسبه می شود. تفاوت ساختار ماشین بولتزمن محدود شده عمیق با ساختار ماشین

بولتزمن با واحدهای پنهان و نمایان در این است که در ساختار عمیق مقدار احتمالاتی که ماشین آن را طبق رابطه ۳۶-۱۰ شبیه‌سازی می کند به ازای همان بردار ورودی که در ابتدا اعمال می شود محاسبه می شود در حالی که در ساختار ماشین بولتزمن محدود شده ژرف مقادیر به روزرسانی شده در محاسبات رابطه ۳۶-۱۰ به کار برده می شوند. همچنین در ساختار ژرف نیز مشابه ساختار محدود شده از نگاشت های غیرخطی استفاده می‌کنیم.

برای استفاده از ساختار ماشین بولتزمن محدود شده ژرف به عنوان یک مدل بازنمائی می توان از بردار متغیرهای آخرین لایه پنهان که به ازای هربردار ورودی به روزرسانی می شود به عنوان بردار بازنمائی استفاده کنیم.

با توجه به افزایش تعداد واحدها و اتصالات در ساختار ماشین بولتزمن محدود شده عمیق مشکل آموزشی مربوط به پیچیدگی ساختار که پیش تر به آن اشاره کردیم مجددا در روند آموزش ساختار ظاهر می شود برای رفع این مشکل شبکه های باور عمیق معرفی شده‌اند که ساختاری شبیه به ماشین بولتزمن محدود شده عمیق دارند ولی روند آموزش آن ها متفاوت است. شبکه باور عمیق که یکی از اولین شبکه های مطرح شده در یادگیری ژرف محسوب می شود. این شبکه مشکل محو شدگی گرادیان ها در شبکه های پرسپترون با تعداد لایه پنهان بالا را به کمک استفاده از ماشین های بولتزمن محدود شده‌ای که به صورت پشته‌ای[1]، سری، مطابق شکل ۷-۱۰ در کنار هم قرار گرفته‌اند حل می کند.



شکل ۷-۱۰: ساختار شبکه باور عمیق

روند آموزش شبکه باور عمیق بدین صورت است که ابتدا یک ماشین بولتزمن مطابق شکل ۵-۱۰ آموزش داده می شود پس از آموزش به ازای بردارهای ورودی، بردارهای پنهان توسط ساختار تولید می شود. این بردارهای پنهان به عنوان بردار ورودی برای یک ماشین بولتزمن دیگر محسوب می شود که به صورت سری با ماشین اول در ساختار قرار داده شده است، پس از آموزش ماشین

[1] Stack RBMs

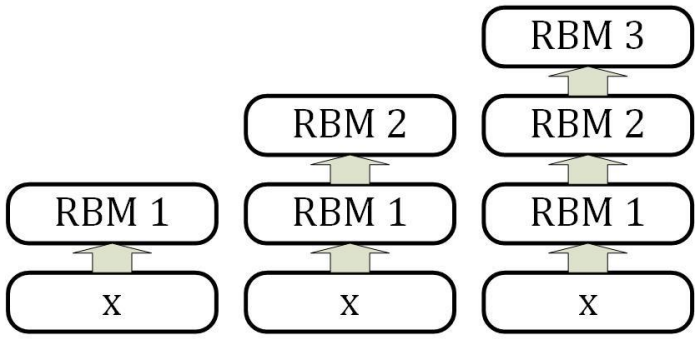
دوم بردار های پنهان آن به عنوان ورودی برای آموزش ماشین بعدی استفاده می شود بدین ترتیب ساختاری متشکل از چندین لایه، ماشین بولتزمن، که هرکدام جداگانه و به صورت بدون نظارت و حریصانه[1] آموزش دیده اند مطابق شکل ۷-۱۰ ایجاد می شود که همان شبکه باور عمیق است. از بردار پنهان نهایی این شبکه می توان برای اهداف دیگر استفاده کرد، برای مثال در مجموعه دادگان یک شبکه عصبی که بعد ورودی بالایی دارند، می توان قبل از وارد کردن دادگان به شبکه عصبی پرسپترون ابتدا آن ها را به عنوان ورودی به یک شبکه باور عمیق آموزش داده شده، داد و سپس بردار های بازنمائی حاصل را که نگاشت های مختلفی در لایه های شبکه باور عمیق بر آن اعمال شده، و بعد آن کاهش یافته، را وارد شبکه پرسپترون کرد، در این صورت نیازی به شبکه عصبی پرسپترون با ظرفیت بالا، تعداد لایه پنهان بالا، نبوده و شبکه پرسپترون با بردار بازنمائی حاصل از شبکه باور عمیق به عنوان ورودی و مقدار مطلوب متناظر در داده های اصلی آموزش داده می شود. این روش در کاهش بعد دادگانی که بعد بسیار بالایی داشته و اعمال کاهش بعد با مدل های کوچک مانند ماشین بولتزمن محدود شده چندان نتیجه مطلوبی ندارد، عملکرد بسیار خوبی دارد.

بدین ترتیب رابطه به روز رسانی واحدهای پنهان و ورودی برای لایه اول یک شبکه باور عمیق به ترتیب مطابق رابطه ۶۹-۱۰ و ۷۰-۱۰ است. در ادامه آموزش ساختار پس از آموزش لایه اول، ماشین بولتزمن محدود، مطابق روابط تابع هزینه و گرادیان های بخش ۳-۲-۱۰ از آن برای تولید بردارهای پنهان متناظر $\boldsymbol{h}$ با دادگان آموزشی $\boldsymbol{x}$ استفاده می شود. این مجموعه بردارهای پنهان $\boldsymbol{h}$ خود به عنوان دادگان آموزشی در واحدهای ورودی، نمایان، لایه دوم استفاده می شوند. بدین ترتیب روابط به روزرسانی واحدهای پنهان و نمایان لایه دوم شبکه باور عمیق به ترتیب مطابق رابطه ۷۳-۱۰ و ۷۴-۱۰ خواهد بود. آموزش این لایه نیز براساس روابط بخش ۳-۲-۱۰ است. بدین ترتیب آموزش لایه به لایه حریصانه مطابق شکل ۸-۱۰ تا آخرین لایه شبکه باور عمیق ادامه داشته و در هر لایه بردارهای پنهان لایه قبلی به عنوان بردار ورودی لایه بعدی در نظر گرفته می شود.

$$P_{h_j^2} = f(\sum_{i=1}^{n} w_{i,j} h_j^1 + b_{h_j}) \qquad (۱۰-۷۳)$$

$$O_{h_j^2} = f(\sum_{i=1}^{n} w_{i,j} h_j^2 + b_{h_j}) \qquad (۱۰-۷۴)$$

[1] Greedy



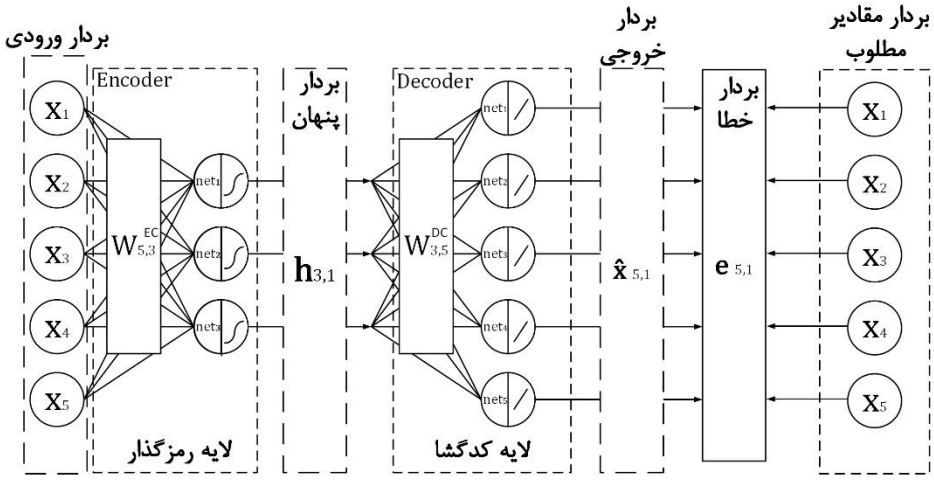
شکل ۸-۱۰: روند آموزش حریصانه شبکه باور عمیق

در آموزش ساختار شبکه باور عمیق باید حتما باید توجه کنیم که بردارهای پنهان تولید شده توسط اولین ماشین بولتزمن محدود شده که به عنوان ورودی ماشین بولتزمن محدود شده دوم استفاده می شود، بعد از اتمام آموزش ماشین اول و تنظیم پارامترهای آن تولید می شود و در طی روند آموزش ماشین اول هیچ برداری وارد ماشین دوم نمی شود. به عبارت دیگر از پارامترهای آموزش داده شده ماشین اول بردارهای ورودی ماشین دوم، پنهان ماشین اول، ساخته می شود.

۱۰.۳ خود رمزگذارها[1]

یکی دیگر از مدل های مطرح در یادگیری بازنمائی خودرمزگذارها هستند. ساختار خود رمزگذارهای مشابه شبکه های عصبی پرسپترون چند لایه است. خودرمزگذارها نشات گرفته از ایده ای مشابه ماشین های بولتزمن محدود شده هستند ولی تفاوت اصلی آن ها با ماشین های بولتزمن محدود شده این است که خودرمزگذار ها مدل های متمایزکننده محسوب می شوند. به عبارت دیگر خودرمزگذار یک شبکه عصبی دو لایه پرسپترون است که بردار مقادیر مطلوب آن همان بردار ورودی آن است از این رو آموزش آن، آموزش بدون سرپرست محسوب می شود. البته همان طور که در ماشین های بولتزمن با اعمال تغییراتی می توانستیم این مدل های را مدل های متمایزکننده در نظر بگیریم، خودرمزگذارها نیز دارای انواع مولد هستند. مطلب حائز اهمیت در اینجا این است همانند ماشین های بولتزمن که فرض مدل متمایزکننده ماهیت اصلی ماشین بولتزمن که مدل مولد است را تغییر نمی داد، در خودرمزگذارها نیز اعمال تغییراتی برای فرض مدل مولد ماهیت متمایزکننده آن را تغییر نمی دهد. شکل ۹-۱۰ یک خودرمزگذار متشکل از لایه کدکننده، لایه پنهان، و لایه کدگشا، لایه خروجی، را نشان می دهد.

[1] Autoencoders



شکل ۹-۱۰: ساختار خودرمزگذار

قبل از توضیح در مورد ساختار خودرمزگذارها ابتدا جزئیات تفاوت ماهیت متمایزکننده این مدل را با ماهیت مولد ماشین های بولتزمن محدود شده بررسی می کنیم. در بخش ۴-۲-۱۰ بررسی کردیم که در صورت اضافه شدن واحدهای پنهان به ساختار ماشین بولتزمن، با توزیع یکنواخت به هر کدام از این واحدها یک مقدار احتمالاتی اولیه بین صفر تا یک نسبت می دهیم که در صورت بزرگتر بودن مقدار ورودی محاسبه شده از رابطه ۶۹-۱۰ برای یک واحد پنهان از این مقدار اولیه آن واحد فعال شده و مقدار متغیر تصادفی آن برابر یک، در صورتی که متغیرها باینری باشند، می شود، همچنین مقادیر احتمالات جدید جایگزین مقادیر اولیه می شوند. بنابراین واحدهای پنهان ماشین بولتزمن ساختار شکل ۵-۱۰ دارای یک توزیع احتمالاتی بر اساس مقادیر احتمال هر واحد بوده و از این رو مدل مولد محسوب می شوند. ماشین های بولتزمن محدود شده بر اساس واحدهای پنهان توزیع احتمالاتی، مقدار متغیرهای تصادفی، واحدهای نمایان را محاسبه کرده و الگوی، بردار، بازسازی را در واحدهای نمایان تشکیل می دهند. این توزیع های ذکر شده در ماشین های بولتزمن در ساختار خودرمزگذارها تعریف نمی شوند و هر واحد در خودرمزگذار رفتاری مشابه نورون های شبکه های عصبی پرسپترون دارد.

۱۰.۳.۱ روابط پیشرو خودرمزگذارها

هدف یک خودرمزگذار شبیه سازی یک تابع همانی برای ورودی است به طوری که مقدار مطلوب خودرمزگذار برای خروجی لایه کدگشا همان بردار ورودی باشد، یادگیری بدون سرپرست. بر این اساس روابط پیشرو واحدهای خودرمزگذار در لایه رمزگذار طبق رابطه ۷۵-۱۰ و در لایه کدگشا طبق رابطه ۷۶-۱۰ مشابه نورون های شبکه های عصبی پرسپترون محاسبه می شود که از این جهت به روابط واحدهای ماشین بولتزمن محدود شده، رابطه ۶۹-۱۰ و ۷۰-۱۰، نیز شباهت

دارد.

$$o_i^{EC} = f(\sum_{j=1}^{n} w_{i,j} x_j + b_i) \quad (۱۰-۷۵)$$

$$o_i^{DC} = f(\sum_{j=1}^{n} w_{i,j} o_j^{EC} + b_i) \quad (۱۰-۷۶)$$

فرم ماتریسی روابط فوق برای لایه رمزگذار و کدگشا به ترتیب طبق روابط ۱۰-۷۷ تا ۱۰-۸۰ است.

$$\mathbf{net}_{1\times m}^{EC} = \mathbf{x}_{n\times 1}^{T} \times W_{n\times m}^{EC} \quad (۱۰-۷۷)$$

$$\mathbf{net}_{1\times n}^{DC} = \mathbf{o}_{1\times m}^{EC} \times W_{m\times n}^{DC} \quad (۱۰-۷۸)$$

$$\mathbf{o}_{1\times m}^{EC} = f(\mathbf{net}^{EC}) \quad (۱۰-۷۹)$$

$$\mathbf{o}_{1\times n}^{DC} = f(\mathbf{net}^{DC}) \quad (۱۰-۸۰)$$

۱۰.۳.۲ تابع هزینه خود رمزگذارها

با توجه به این که خود رمزگذارها به عنوان مدل متمایزکننده مشابه شبکه های عصبی پرسپترون شناخته می شوند، برای این مدل ها نیز از تابع میانگین مجموع مربعات خطا استفاده می شود و هدف آن کاهش فاصله اقلیدسی بین بردار خروجی خودرمزگذار، بردار بازسازی، و بردار مطلوب، بردار ورودی، است. رابطه ۱۰-۸۱ نمایش تابع هزینه میانگین مجموع مربعات خطا به ازای یک نمونه داده ورودی برای یک خودرمزگذار است که در آن و به ترتیب بردار ورودی و بردار بازسازی، خروجی، است.

$$E = \frac{1}{2}(\mathbf{x} - \hat{\mathbf{x}})^2 \quad (۱۰-۸۱)$$

۱۰.۳.۳ آموزش خود رمزگذارها به روش گرادیان نزولی

ساختار خودرمزگذارها مشابه شبکه های عصبی چند لایه پرسپترون است بنابراین اگر بعد ورودی شبکه n و تعداد نورون های لایه کدگذار، پنهان، m باشد در این صورت اندازه بردار وزن های لایه کدگذار برابر $n \times m$ خواهد بود. بر همین اساس چون بعد خروجی و ورودی برابر است بنابراین اندازه ماتریس وزن لایه کدگشا، خروجی، برابر با $m \times n$ خواهد بود. درشکل ۱۰-۹ ساختار یک خودرمزگذار و ماتریس وزن های آن نشان داده شده است که در آن اندازه بردار ورودی ۵، وزن

لایه پنهان ۳×۵، بردار خروجی ۳، وزن لایه کدگشا ۵×۳ و بردار خروجی ۵ است.

از سویی وظایف لایه های رمزگذار و کدگشا در خودرمزگذار معکوس یکدیگر هستند؛ بدین معنی که لایه کدگذار بعد ورودی را به بعد پنهان و لایه کدگشا بعد پنهان را به بعد خروجی نگاشت می کند. از این رو می توان نتیجه گرفت که اهمیت و تاثیر هر بعد از بردار ورودی در تشکیل بردار پنهان، خروجی لایه رمزگذار، برابر تاثیر بعد متناظر آن در بردار بازسازی، خروجی لایه کدگشا، به هنگام نگاشت آن از بردار پنهان است، به عبارت ساده تر وزن متناظر هر ورودی در نورون های لایه رمزگذار با وزن نگاشت آن از خروجی لایه رمزگذار به لایه کدگشا برابر است. در ساختار شکل ۹-۱۰ این مطلب نمایش داده شده است.

بنابراین با توجه به توضیحات فوق در مورد اندازه ماتریس وزن ها و مقادیر آن در هر لایه می توانیم برای ساختار خودرمزگذار مطابق شکل ۹-۱۰ رابطه ۸۲-۱۰ را نتیجه بگیریم.

$$W_{n\times m}^{EC} = (W_{m\times n}^{DC})^T \qquad (۱۰-۸۲)$$

با توجه به رابطه ۸۲-۱۰ می توانیم برای سادگی روند آموزش، ساختار خودرمزگذارها فقط وزن لایه کدگشا را به روش گرادیان نزولی آموزش داده و روابط مشتق های زنجیره ای را طبق رابطه ۸۳-۱۰ برای آن توسعه دهیم. ماتریس حاصل از گام فعلی k و گرادیان آموزشی را به عنوان ماتریس وزن لایه کدگشا در گام آینده $k+1$ طبق رابطه ۸۴-۱۰ در نظر می گیریم. با توجه به رابطه ۸۲-۱۰ می توانیم با محاسبه ترانهاده ماتریس وزن های لایه کدگشا در گام آینده، $k+1$، آن را به عنوان ماتریس وزن های لایه رمزگذار در گام آینده، $k+1$، در نظر بگیریم.

$$\frac{\partial E}{\partial W^{DC}} = \frac{\partial E}{\partial e}\frac{\partial e}{\partial \hat{\mathbf{x}}}\underbrace{\frac{\partial \hat{\mathbf{x}}}{\partial net^{DC}}}_{f'(net^{DC})}\underbrace{\frac{\partial net^{DC}}{\partial W^{DC}}}_{\mathbf{o}^{EN}} \qquad (۱۰-۸۳)$$

$$W_{(k+1)}^{DC} = W_{(k)}^{DC} - \eta \frac{\partial E}{\partial W^{DC}}(k) \qquad (۱۰-۸۴)$$

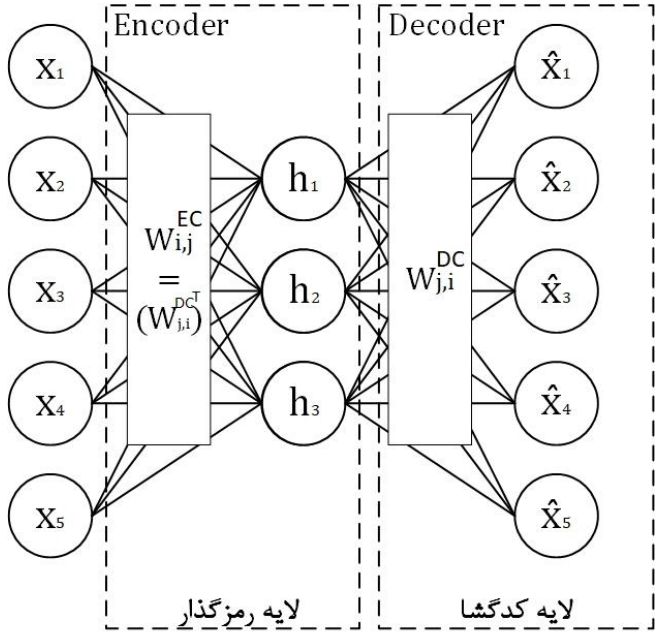
برای آموزش پارامترهای بایاس در لایه کدگذار رابطه ۸۵-۱۰ و ۸۶-۱۰ برقرار است.

$$\frac{\partial E}{\partial \mathbf{b}^{EC}} = \frac{\partial E}{\partial e}\frac{\partial e}{\partial \hat{\mathbf{x}}}\frac{\partial \hat{\mathbf{x}}}{\partial net^{DC}}\underbrace{\frac{\partial net^{DC}}{\partial \mathbf{o}^{EN}}}_{W^{DC}}\underbrace{\frac{\partial \mathbf{o}^{EN}}{\partial net^{EC}}}_{f'(net^{EC})}\underbrace{\frac{\partial net^{EC}}{\partial \mathbf{b}^{EC}}}_{1} \qquad (۱۰-۸۵)$$

$$\mathbf{b}_{(k+1)}^{EC} = \mathbf{b}_{(k)}^{EC} - \eta \frac{\partial E}{\partial \mathbf{b}^{EC}}(k)$$

(۱۰-۸۶)

باید در نظر داشته باشیم که در ساختار خودرمزگذار مفروض، مطابق شکل ۹-۱۰ برای اینکه رابطه ۸۲-۱۰ صدق کند و مقادیر وزنی در لایه های رمزگذار و کدگشا برابر باشند، باید نگاشت لایه کدگشا عکس نگاشت لایه رمزگذار باشد. بنابراین اگر ابعاد ورودی و نورون های خودرمزگذار را به صورت واحدهایی مشابه واحد های ماشین بولتزمن در نظر بگیریم باید ساختار واحدهای لایه کدگشا و ورودی مطابق شکل ۱۰-۱۰ برابر باشند. لذا برای برقراری این شرایط لایه خروجی باید دارای تابع فعال ساز خطی باشد.



شکل ۱۰-۱۰: ساختار متقارن خودرمزگذار

در صورتی که موارد فوق در ساختار خودرمزگذار صدق نکند و رابطه ۸۲-۱۰ برقرار نباشد، در روند آموزش خودرمزگذار به روش گرادیان نزولی در لایه رمزگذار نمی توان از ترانهاده ماتریس وزن های لایه کدگشا استفاده کرد و باید روابط مشتق زنجیره ای را برای این لایه نیز توسعه دهیم. رابطه مشتق زنجیره ای برای لایه رمزگذار مطابق رابطه ۱۰-۸۷ است. در صورت همچنین می توانیم از سایر روش هایی که در فصل معرفی ۷ کردیم مانند استفاده از نورون های انعطاف‌پذیر و... نیز برای بهبود عملکرد هم در لایه رمزگذار و هم در لایه کدگشا استفاده کنیم.

$$\frac{\partial E}{\partial W^{EC}} = \frac{\partial E}{\partial e}\frac{\partial e}{\partial \hat{\mathbf{x}}}\frac{\partial \hat{\mathbf{x}}}{\partial net^{DC}}\frac{\partial net^{DC}}{\partial \mathbf{o}^{EN}}\frac{\partial \mathbf{o}^{EN}}{\partial net^{EC}}\underbrace{\frac{\partial net^{EC}}{\partial W^{EC}}}_{\mathbf{x}} \qquad (۱۰-۸۷)$$

برای آموزش پارامترهای بایاس در لایه کدگشا نیز در صورتی که لایه دارای بایاس باشد از رابطه ۱۰-۸۸ و ۱۰-۸۹ محاسبه می شود.

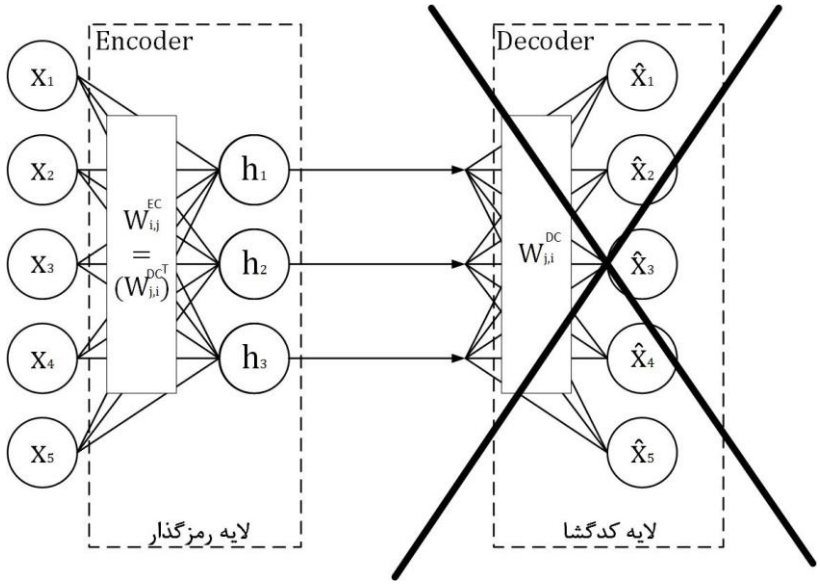
$$\frac{\partial E}{\partial \mathbf{b}^{DC}} = \frac{\partial E}{\partial e}\frac{\partial e}{\partial \hat{\mathbf{x}}}\frac{\partial \hat{\mathbf{x}}}{\partial net^{DC}}\underbrace{\frac{\partial net^{DC}}{\partial \mathbf{b}^{DC}}}_{1} \qquad (۱۰-۸۸)$$

$$\mathbf{b}^{DC}_{(k+1)} = \mathbf{b}^{DC}_{(k)} - \eta \frac{\partial E}{\partial \mathbf{b}^{DC}}(k) \qquad (۱۰-۸۹)$$

مشابه شبکه های عصبی پرسپترون چند لایه در خود رمزگذارها نیز قسمتی از دادگان به عنوان داده های تست در نظر گرفته شده و در هر گام پس از استفاده از مجموعه داده های آموزشی برای آموزش پارامترهای خود رمزگذار عملکرد خودرمزگذار با دادگان تست سنجیده می شود، در این مرحله نیز مانند مرحله آموزش مقدار مطلوب همان مقدار ورودی در نظر گرفته شده و تابع هزینه مطابق رابطه ۱۰-۸۱ براساس فاصله اقلیدسی بین بردار خروجی خودرمزگذار، بردار بازسازی، و بردار ورودی محاسبه می شود.

پس از اتمام روند آموزش و تست خودرمزگذار به ازای تعداد گام[1] های مدنظر، در نهایت لایه خروجی مطابق شکل ۱۰-۱۱ حذف شده و ساختار نهایی فقط شامل لایه رمزگذار خواهد بود. در این حالت بردار پنهان، بردار خروجی لایه رمزگذار، به عنوان بردار بازنمائی متناظر با بردار ورودی در نظر گرفته شده و از آن به جای مقادیر بردار ورودی اصلی که در مجموعه دادگان قرار داشت استفاده می شود.

[1] Epoch

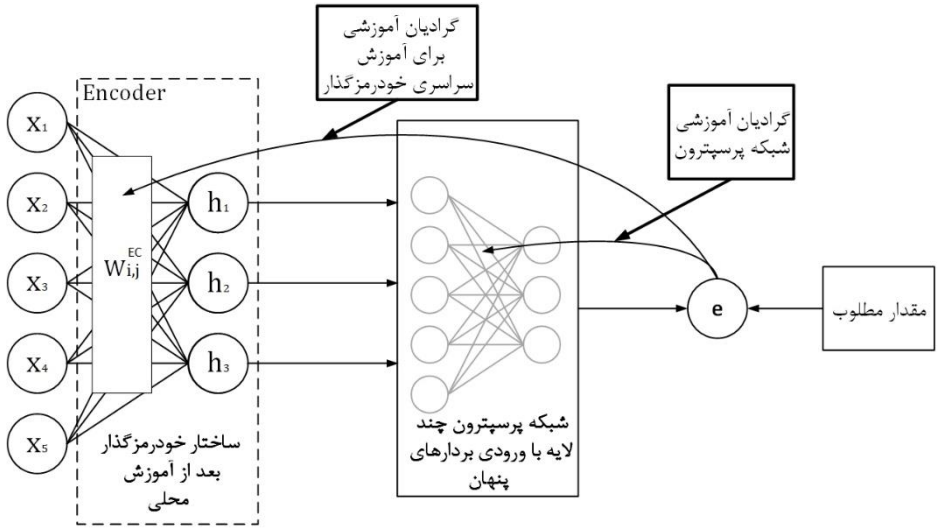


شکل ۱۱-۱۰: حذف لایه کدگشا بعد از آموزش

۱۰.۳.۴ آموزش سراسری[1] و محلی[2] در خودرمزگذارها

به روند آموزش مطرح شـده در قسـمت ۳-۳-۱۰ برای سـاختار خودرمزگذارها آموزش محلی اطلاق می شود، روندی که در آن خودرمزگذار بدون سرپرست با دادگان آموزش داده شده و سپس از آن به عنوان یک مدل بازنمائی استفاده می شود. در صورتی که پس از آموزش محلی خودرمزگذار بردار پنهان به عنوان ورودی برای یک مدل دیگر که به روش گرادیان نزولی آموزش داده می شـود، مانند یک شبکه عصبی پرسپترون، در نظر گرفته شود، آنگاه می توان پارامترهای آموزش داده شده خودرمزگذار در مرحله آموزش محلی را دوباره نسـبت به مقادیر گرادیان های مدل اصـلی، شـبکه پرسپترون، که خروجی خود رمزگذار ورودی آن است آموزش داد که به آن آموزش سراسری اطلاق می شود. در آموزش سراسری پارامترهای آموزش داده شده در آموزش محلی برای خودرمزگذار به عنوان مقادیر اولیه پارامترها برای آموزش سراسری آن در نظر گرفته می شوند. شکل ۱۲-۱۰ مسیر آموزش سراسری یک خودرمزگذار پس از آموزش محلی آن را نشان می دهد.

[1] Global learning
[2] Local learning



شکل ۱۲-۱۰: روند آموزش سراسری

با توجه به ساختار شکل ۱۲-۱۰ می توان چنین استنباط کرد که خودرمزگذار به عنوان یک لایه پنهان به ابتدای ساختار شبکه پرسپترون اضافه شده است که وزن های اولیه برای آن در روند آموزش شبکه همان وزن های حاصل از آموزش محلی است. بدین ترتیب اگر برای شبکه پرسپترون دو لایه فرض کنیم آنگاه مشتق زنجیره‌ای برای آموزش وزن خودرمزگذار مطابق رابطه ۹۰-۱۰ خواهد بود. در رابطه ۹۰-۱۰، $\boldsymbol{x}$ بردار ورودی اصلی و $\boldsymbol{h}$ بردار پنهان حاصل از خودرمزگذار است که به عنوان ورودی شبکه پرسپترون در نظر گرفته می شود.

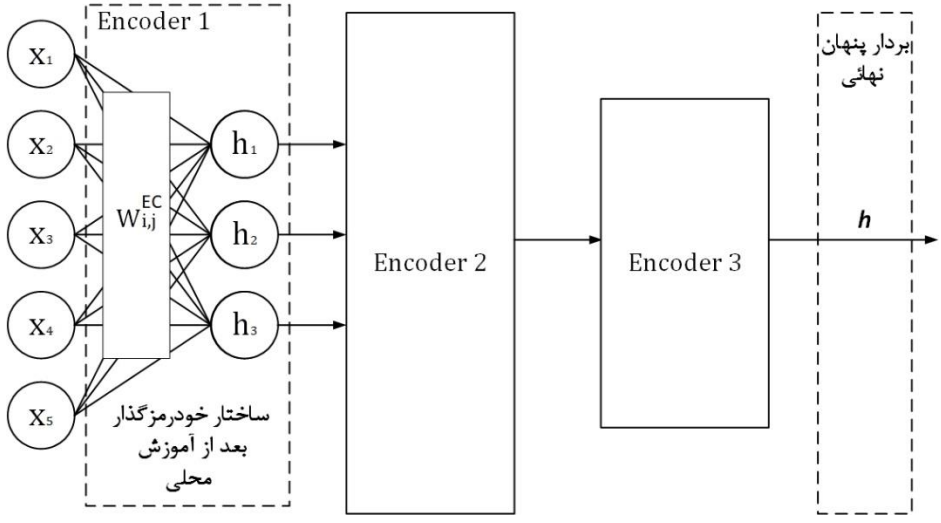
$$\frac{\partial E}{\partial W^{EC}} = \frac{\partial E}{\partial e}\frac{\partial e}{\partial \hat{\mathbf{x}}}\frac{\partial \hat{\mathbf{x}}}{\partial net^{2}}\frac{\partial net^{2}}{\partial \mathbf{o}^{1}}\frac{\partial \mathbf{o}^{1}}{\partial net^{1}}\frac{\partial net^{1}}{\partial \mathbf{h}}\underbrace{\frac{\partial \mathbf{h}}{\partial net^{EC}}}_{f'(net^{EC})}\underbrace{\frac{\partial net^{EC}}{\partial W^{EC}}}_{\mathbf{x}} \qquad (۱۰-۹۰)$$

۱۰.۳.۵ خود رمزگذارهای پشته ای[1]

در معرفی شبکه های باور عمیق بررسی کردیم که استفاده از چند ماشین بولتزمن محدود شده به صورت پشته ای، یک شبکه باور عمیق، ممکن است عملکرد بهتری نسبت به یک ماشین بولتزمن محدود شده داشته باشد. در ساختار خودرمزگذارها نیز همانند شبکه های عصبی پرسپترون چند لایه که در آن لایه های شبکه به صورت سری در کنار هم قرار داشته و خروجی یک لایه به عنوان ورودی لایه بعدی در نظر گرفته می شود، می توان چند ساختار خودرمزگذار را نیز به صورت پشته ای، سری، در کنار هم قرار داد. روند تشکیل یک ساختار خودرمزگذار پشته ای مشابه شبکه باور

[1] Stack autoencoder

عمیق است، بدین صورت که ابتدا یک خودرمزگذار به صورت محلی آموزش داده شده و سپس لایه کدگشای آن حذف می شود. ساختار حاصل به عنوان لایه اول خودرمزگذار پشته ای در نظر گرفته شده و بردارهای پنهان آن برای آموزش محلی ساختار خودرمزگذار دیگری استفاده می شود، خودرمزگذار دوم نیز پس از آموزش و حذف لایه کدگشا به صورت پشته ای درکنار خودرمزگذار اول لایه دوم ساختار پشته ای را تشکیل می دهد. با ادامه این روند در نهایت یک ساختار خودرمزگذار پشته ای خواهیم داشت که یک مدل بازنمائی محسوب شده و از بردار پنهان آن، بردار پنهان آخرین خودرمزگذار، به عنوان بردار بازنمائی استفاده می شود. در این صورت ساختار حاصل خودرمزگذار پشته ای نامیده می شود. شکل ۱۳-۱۰ یک ساختار خودرمزگذار پشته ای با سه لایه رمزگذار را نشان می دهد. در خودرمزگذارهای پشته ای ممکن است ساختار هرلایه متفاوت بوده و دارای تابع هزینه مخصوص به خود باشد که با لایه های دیگر متفاوت است، در ادامه این فصل به بررسی ساختارها و توابع هزینه مختلف در خود رمزگذارها خواهیم پرداخت.



شکل ۱۳-۱۰: ساختار خودرمزگذار پشته‌ای

۱۰.۳.۶ چالش ها و مشکلات آموزش خود رمزگذارها

با توجه به شباهت های بین شبکه های عصبی پرسپترون چند لایه و خودرمزگذارها، چالش های مطرح در روند آموزش شبکه های عصبی پرسپترون که در فصل ۷ بررسی کردیم در مورد خودرمزگذارها نیز وجود دارد و اغلب رویکردهای مشابه رویکرد های مورد استفاده در شبکه های عصبی برای مواجهه با این چالش ها، در خودرمزگذارها نیز مورد استفاده قرار می گیرد. علاوه بر این چالش ها خودرمزگذارها دارای مشکلاتی مختص به خود نیز هستند که به چند مورد از مهم ترین آن ها اشاره می کنیم:

- در روند آموزش خودرمزگذار ها با مجموعه دادگان آموزشی، هدف کمینه کردن خطای ناشی از فاصله اقلیدسی بین بردار بازسازی و بردار ورودی است. در این روند خطای ساختار خودرمزگذار به ازای این نقاط گسسته که همان نمونه های مجموعه دادگان است کاهش می یابد. اما این روند باعث می شود لایه رمزگذار و کدگشا فقط به ازای همان نقاط آموزش دیده و برای فضای بین نقاط این نقاط آموزشی نداشته باشد. از این رو پس از آموزش خودرمزگذار برای بازه های بین این نقاط آموزشی بردار پنهان مناسبی ارائه نمی کند.
- هدف اصلی از توسعه ساختار خود رمزگذارها تبدیل ورودی اصلی به بردار پنهانی است که لایه آن را رمزگذار ایجاد می کند، اما در آموزش خودرمزگذار از آن جایی که تابع هزنیه طبق رابطه ۸۱-۱۰ فقط براساس فاصله بردار بازسازی از بردار ورودی محاسبه شده و مقادیر بردار پنهان تاثیری در آن ندارند. از این رو ممکن است پارامترهای خودرمزگذار به گونه ای آموزش ببینند که نسبت به تغییرات جزئی درمقادیر ورودی، بازه تغییرات مقادیر بردار پنهان بزرگ باشد در صورتی که در یک مدل بازنمائی این حالت مطلوب نیست و انتظار داریم نسبت تغییر مقادیر بردار های ورودی و پنهان به یک اندازه باشد. علاوه بر این ممکن است بردارهای پنهان نسبت داده شده به بردارهای ورودی مختلف نسبت به هم تمایز چندانی نداشته باشند، در صورتی که هدف از آموزش یک مدل بازنمائی نگاشت هر بردار ورودی به یک بردار پنهانی است که نسبت به بردارهای پنهان سایر نمونه های ورودی متمایز باشد.
- در مواردی ممکن است مجموعه دادگانی که خودرمزگذار آن ها را به بردارهای پنهان تبدیل می کند آغشته به نویز باشند. در این صورت در مقادیر بردارهای ورودی نیز عدم قطعیت وجود خواهد داشت و نویز دادگان به بردارهای پنهان نیز منتقل می شود. این نویز در نهایت در عملکرد ساختاری که از این بردارهای پنهان به عنوان ورودی استفاده می کند نیز تاثیر خواهد گذاشت.
- عملکرد خودرمزگذار در کاربرد کاهش بعد دادگان مشابه تحلیل مولفه های اصلی است. بدین صورت که با در مرحله آموزش خودرمزگذار می تواند الگوها و وابستگی بین ابعاد مختلف ورودی را تشخیص داده و آن ها را حذف کند. بدین منظور تعداد نورون های لایه های پنهان همواره باید کوچکتر از بعد ورودی باشد تا ابعاد ورودی کاهش یابد، این قید عملکرد خودرمزگذار را در ایجاد بردارهای پنهان تحت تاثیر قرار می دهد. از طرفی اگر ابعاد وابستگی چندانی به هم نداشته باشند تلاش برای آموزش خودرمزگذار با کاهش بعد چندان موثر نخواهد بود.

۱۰.۳.۷ تنک زایی[1] در خود رمزگذارها

برای رفع مشکل ذکر شده در بخش ۶-۳-۱۰ در مورد عدم عملکرد مناسب خودرمزگذار با قرار دادن قید تعداد نورون های کوچکتر از ورودی در لایه پنهان به ویژه در مواردی که ابعاد مختلف ورودی به هم وابستگی چندانی نداشته باشند، می توان تعداد نورون ها در لایه پنهان را برابر یا بزرگتر از ورودی قرار دهیم، بدین ترتیب بردار های پنهان این خودرمزگذار حاوی الگوهایی بین ابعاد اصلی بردار ورودی است. پس از این مرحله برای کاهش بعد بردار پنهان حاصل می توانیم از یک خودرمزگذار دیگر به صورت پشته ای که از بردار پنهان خودرمزگذار اول به عنوان استفاده می کند و تعداد نورون های لایه پنهان آن کوچکتر از ورودی است استفاده کنیم.

از طرفی اشاره کردیم که هدف یک مدل بازنمائی ایجاد بردارهای پنهان مختص هر ورودی به گونه ای است که نسبت به بردار پنهان سایر بردارهای ورودی حدالامکان متمایز باشد، این مورد به ویژه در حالت فوق الذکر که تعداد نورون های لایه پنهان بزرگتر یا مساوی با بعد بردار ورودی است از مسائل چالش برانگیز در روند آموزش مدل های خودرمزگذار است. برای رفع معضل مربوط به ایجاد تمایز بین بردارهای پنهان ورودی های مختلف در روند آموزش خودرمزگذار تنک زایی تعریف می شود. بدین منظور به تابع هزینه خودرمزگذار یک جمله تنک زایی اعمال می کنیم تا فعالیت نورون های لایه پنهان در طول روند آموزش به حداقل میزان ممکن برسد، خروجی نورون ها تا حد ممکن نزدیک به صفر باشد؛ بدین ترتیب بردار پنهان حاصل به ازای هر بردار ورودی در حالت ایده آل برداری یک بردار باینری خواهد بود بیشترین تمایز را با بردار پنهان سایر ورودی ها داشته و مختص ورودی متناظر با خود است، مشابه بارکد کالا ها. جمله تنک زایی را می توانیم به دو صورت زیر در تابع هزینه آموزش خودرمزگذار تعریف کنیم:

- **میانگین خروجی نورون های فعال:**

هدف از اعمال این روش تنک زائی وادار کردن خودرمزگذار به غیر فعال کردن نورون های لایه پنهان در طول روند آموزش و متعاقبا استفاده از مدل آموزش دیده است. بدین ترتیب با اعمال این روش خروجی نورون های لایه پنهان تا حد ممکن به سمت مقدار صفر سوق داده می شوند. برای اعمال این روش تنک زایی طبق رابطه ۹۱-۱۰ مجموع میانگین خروجی نورون های لایه پنهان را، مشابه جمله تنظیم کننده مقدار وزن ها که در فصل ۸ بررسی کردیم، به تابع هزینه اضافه می کنیم. در رابطه ۹۱-۱۰، a_i^h مقدار خروجی نورون i لایه رمزگذار، λ یک پارامتر اسکالر با مقداری بین صفر تا یک برای تنظیم حساسیت تابع هزینه نسبت به جمله تنک زایی و N تعداد نورون های لایه رمزگذار است.

$$E = \frac{1}{2}(\mathbf{x}-\hat{\mathbf{x}})^2 + \frac{\lambda}{N}\sum_i \left\| a_i^h \right\|_1 \qquad (۹۱-۱۰)$$

[1] Sparce coding

در رابطه ۹۱-۱۰ از نرم اول مقادیر خروجی نورون ها در جمله تنک زایی استفاده شده است، اما همانند جمله تنظیم کننده مقادیر وزن ها که در فصل ۸ بررسی کردیم می توان از نرم دوم نیز برای محاسبه آن استفاده کنیم که نسبت به نرم یک رفتار ملایم تری داشته و باعث افزایش پایداری روند آموزش مدل می شود، در عوض محاسبات مربوط به نرم دوم نسبت به نرم اول پیچیده تر است. با اعمال جمله تنک زایی باید دو لایه رمزگذار و کدگشای ساختار خودرمزگذار به طور مجزا آموزش داده شوند لذا برای آموزش آن نمی توانیم از رابطه ۸۲-۱۰، ترانهاده وزن های کدگشا به عنوان وزن های لایه رمزگذار، استفاده کنیم. در ساختار یک خودرمزگذار مشتقات زنجیره ای لایه کدگشا با تابع هزینه ای مشابه رابطه ۹۱-۱۰ مشابه رابطه ۸۳-۱۰ و مشتقات زنجیره ای لایه رمزگذار مطابق رابطه ۹۲-۱۰ خواهد بود.

$$\frac{\partial E}{\partial W^{EC}} = \Big(\frac{\partial E}{\partial e}\frac{\partial e}{\partial \hat{\mathbf{x}}}\frac{\partial \hat{\mathbf{x}}}{\partial net^{DC}}\frac{\partial net^{DC}}{\partial \mathbf{o}^{EN}} + \underbrace{\frac{\partial E}{\partial \mathbf{o}^{EN}}}_{\frac{N}{\lambda}}\Big)\underbrace{\frac{\partial \mathbf{o}^{EN}}{\partial net^{EC}}}_{f'(net^{EC})}\underbrace{\frac{\partial net^{EC}}{\partial W^{EC}}}_{\mathbf{x}}$$

$$o_i^{EN} = a_i^h \tag{۹۲-۱۰}$$

- **هم پوشانی توزیع برنولی:**

در تنک زایی به روش میانگین خروجی نورون های فعال از آن جایی که هدف سوق دادن مقادیر خروجی همه نورون های لایه رمزگذار به سمت مقدار صفر تعریف می شود، با توجه به رابطه ۹۱-۱۰ ممکن است مقادیر خروجی همه نورون های این لایه یک مقدار کوچک نزدیک به صفر اختیار کنند، همه ابعاد بردار پنهان دارای این مقادیر کوچک باشند، در این صورت بردارهای پنهان شبیه به بردارهای باینری که به برای هر بردار ورودی منحصر بفرد باشند ایجاد نخواهند شد، بدین ترتیب معضل مربوط به تمایز بین بردارهای پنهان که هدف از تعریف تنک زایی رفع آن بود همچنان وجود خواهد داشت. برای رفع این معضل به جای استفاده از روش تنک زایی مقدار میانگین خروجی ها که پیشتر اشاره کردیم، می توانیم از هم پوشانی توزیع برنولی برای برخی، از نورون های لایه رمزگذار، و یا همه آن ها، به عنوان جمله تنک زایی در تابع هزینه استفاده کنیم بدین ترتیب فعالیت هر نورون به طور مستقل در تابع هزینه اعمال شده و معضل ذکر شده رفع می شود.

برای توسعه این تابع هزینه ابتدا یک توزیع برنولی مطلوب $\hat{p}_{(x)}$ با مقدار احتمال $\hat{\rho}$ برای برخی از نورون های لایه پنهان، و یا همه آن ها، در نظر می گیریم تا طبق آن تعداد نورون های فعال را برای لایه رمزگذار تعیین کنیم از آنجایی که هدف فعالیت حداقلی برای نورون های تعیین شده در لایه رمزگذار است مقدار احتمال $\hat{\rho}$ را برای نورون j-ام، نورونی که میخواهیم فعالیت آن محدود

شود، مقداری کوچک در نظر می گیریم و تابع هزینه خودرمزگذار را به گونه ای توسعه می دهیم تا در روند آموزش خودرمزگذار را وادار کنیم که توزیع برنولی این نورون را در لایه رمزگذار $P_{j(x)}$ با احتمال ρ_j را به توزیع هدف $\hat{P}_{(x)}$ با احتمال $\hat{\rho}$ نزدیک کند.

در فصل ۳ توزیع برنولی را معرفی کردیم، این توزیع برای مقادیر باینری که در اینجا بر اساس فعال بودن و یا غیرفعال بودن نورون های لایه رمزگذار است تعریف می شود. تابع جرم احتمال[1] این توزیع مطابق رابطه ۹۳-۱۰ است که در آن ρ احتمال فعال بودن نورون در لایه رمزگذار بوده و x به ازای نورون های فعال برابر یک و به ازای نورن های غیرفعال برابر صفر است. در این قسمت نیز مشابه حالت قبل از حد آستانه نصف بازه خروجی نورون برای تعیین فعال یا غیرفعال بودن آن استفاده می کنیم.

$$P_{(x)} = \rho^x (1-\rho)^{1-x} \tag{۹۳-۱۰}$$

در ساختار خودرمزگذار توزیع برنولی هدف و نورون های لایه رمزگذار هردو دارای تابع جرم احتمال مشابه رابطه ۹۳-۱۰ هستند، هدف ما این است که این دو توزیع نسبت به هم بیشترین هم پوشانی ممکن را داشته باشند، حدالامکان ρ_j و $\hat{\rho}$ دارای مقادیر نزدیک به هم باشند، بنابراین طبق توضیحات فصل ۸ معیار واگرائی KL این دو توزیع را طبق رابطه ۹۴-۱۰ به تابع هزینه خودرمزگذار اضافه می کنیم که در آن λ یک پارامتر با مقداری بین صفر تا یک است که حساسیت تابع هزینه نسبت به جمله تنک زایی تعیین می کند.

$$E = \frac{1}{2}(\mathbf{x}-\hat{\mathbf{x}})^2 + \lambda KL(P_{(x)} \,||\, \hat{P}_{(x)}) \tag{۹۴-۱۰}$$

در یک توزیع برنولی با مقدار احتمالی ρ، مشابه رابطه ۹۳-۱۰، مقدار امید ریاضی، میانگین، توزیع برابر با مقدار ρ است، $E_{[x]} = \rho$، بنابراین مقدار احتمال توزیع برنولی حاصل از مقادیر خروجی لایه رمزگذار مطابق رابطه ۹۵-۱۰ با استفاده از مقدار میانگین خروجی نورون j-ام لایه رمزگذار به ازای نمونه های مختلف ورودی تعریف می شود. با توجه به نحوه پیاده سازی الگوریتم آموزشی تعداد نمونه های N در رابطه ۹۵-۱۰ می تواند یک نمونه، نمونه های دسته و یا همه نمونه های آموزشی به ترتیب برای روش های تصادفی، دسته ای کوچک و دسته ای که در فصل ۹ بررسی کردیم باشد.

$$\rho_j = \frac{1}{N}\sum_{i=1}^{N} \left\| a^h_{j(i)} \right\|_1 \tag{۹۵-۱۰}$$

[1] Probability mass function (PMF)

با این توضیحات معیار واگرائی KL تعریف شده در رابطه ۹۴-۱۰ مطابق رابطه ۱۰-۹۶ تعریف می شود.

$$KL(P_{(x)} \,||\, \hat{P}_{j(x)}) = KL(\rho \,||\, \hat{\rho}_j) = \rho \log \frac{\rho}{\hat{\rho}_j} + (1-\rho) \log \frac{1-\rho}{1-\hat{\rho}_j} \qquad (۱۰-۹۶)$$

رابطه ۹۶-۱۰ معیار واگرائی KL را به ازای یک نورون لایه پنهان تعریف می کند اما همان طور که پیشتر اشاره کردیم، این روش تنک زایی را می توانیم برای برخی و یا همه نورون های لایه پنهان اعمال کنیم؛ اعمال این روش به همه نورون ها بیشتر مرسوم بوده و باعث ایجاد بردارهای پنهانی با تمایز بالاتر نسبت به سایر می شود، بنابراین رابطه ۹۶-۱۰ را به ازای همه نورون های لایه پنهان به صورت رابطه ۹۷-۱۰ تعریف می کنیم. در رابطه ۹۷-۱۰، n_1 تعداد نورون های لایه پنهان است.

$$KL(\rho \,||\, \hat{\rho}) = \sum_{j=1}^{n_1} \rho \log \frac{\rho}{\hat{\rho}_j} + (1-\rho) \log \frac{1-\rho}{1-\hat{\rho}_j} \qquad (۱۰-۹۷)$$

با جایگذاری رابطه ۹۵-۱۰ در رابطه ۹۷-۱۰ و سپس جایگذاری رابطه ۹۷-۱۰ در رابطه ۹۴-۱۰ می توانیم تابع هزینه این خودرمزگذار را محاسبه کنیم.

در هر دو روش ذکر شده در این بخش برای اعمال تنک زایی در ساختار خودرمزگذارها فعالیت نورون ها بر اساس مقدار خروجی تعریف شده است، بنابراین در تعریف روابط این روش ها فرض بر این است که بازه خروجی توابع فعال ساز لایه رمزگذار در ساختار بین صفر تا یک بوده و غیرفعال بودن نورون در صورتی اتفاق می افتد که خروجی آن صفر، یا مقداری نزدیک به صفر، باشد.

۱۰.۳.۸ خود رمزگذار نوییززدا[1]

اشاره کردیم که یکی از مشکلات مربوط به خودرمزگذارها انتقال نویز و عدم قطعیت موجود در دادگان ورودی به بردار پنهان است. برای مرتفع کردن این مورد در مرحله آموزش خودرمزگذار با مجموعه دادگانی فاقد نویز محسوس، مقادیر نویز به ورودی طبق رابطه ۹۸-۱۰ اضافه می کنیم ولی مقدار مطلوب خودرمزگذار را همان ورودی اصلی فاقد نویز در نظر می گیریم. در رابطه ۹۸-۱۰، δ مقادیر کوچکی هستند که به عنوان نویز به ورودی اضافه، یا از آن کم، می شوند. با توجه به این که در حالت کلی در آموزش شبکه های عصبی و خودرمزگذارها توزیع نویز موجود در دادگان آموزشی را توزیع گوسی فرض کرده و از نرم[2] دو، میانگین مجموع مربعات خطا، برای محاسبه تابع هزینه این ساختارها استفاده می کنیم در اینجا نیز با توجه به تابع هزینه میانگین مجموع مربعات

[1] Denoising autoencoder
[2] Norm

خطا، توزیع نویزهای اضافه شده را توزیع گوسی با میانگین صفر، $\mu = 0$، و واریانس دلخواه σ مطابق رابطه ۹۹-۱۰ در نظر می گیریم. بدیهی است که هرچه مقدار واریانس توزیع مقدار بزرگ تری باشد تاثیر نویز در داده ها بیشتر خواهد بود.

$$\mathbf{x} := \mathbf{x} \pm \delta \tag{۱۰-۹۸}$$

$$\mathbf{x} \sim \mathcal{N}(0, \sigma^2) \tag{۱۰-۹۹}$$

در این صورت مدل آموزش داده شده به ازای مقادیر مختلف ورودی با تغییرات جزئی مقادیر تقریبا ثابتی را در خروجی، بردار پنهان، ایجاد کرده و نسبت به این تغییرات جزئی حساسیتی ندارد و نسبت به نویز وارد شده به آن در روند آموزش مقاوم شده است. بنابراین می توان از این مدل بازنمائی به عنوان یک فیلتر نیز استفاده کرد.

۱۰.۳.۹ خود رمزگذار انقباضی[1]

در ساختار خودرمزگذار نویززدا هدف مقاوم کردن خودرمزگذار نسبت به نویز است، به عبارت دیگر در ساختار نویززدا هدف حذف تغییرات بردار خروجی و پنهان ناشی تغییرات مقادیر ورودی است. این هدف با افزودن نویز گوسی به دادگان آموزش خودرمزگذار با مقادیر مطلوب اصلی محقق می شود که در بخش ۸-۳-۱۰ مورد بررسی قرار دادیم، در حالی که این افزودن نویز به دادگان اثرات نامطلوبی به همراه داشته و ممکن است توزیع دادگان آموزشی با اضافه شدن آن تغییر کند. بدین ترتیب روند آموزش خودرمزگذار با دادگانی با توزیعی متفاوت از توزیع اولیه دادگان انجام شده و بردارهای پنهانی که با پارامترهای این ساختار تولید می شوند نیز تحت تاثیر این تغییر قرار خواهند گرفت. بنابراین اگر توزیع حاصل از افزودن نویز به دادگان متفاوت از توزیع دادگان اصلی باشد عملکرد مدل بازنمائی آموزش داده شده کاهش خواهد یافت که این کاهش با میزان مغایرت دو توزیع دادگان رابطه مستقیم دارد.

بنابر توضیحات فوق لزوم توسعه خودرمزگذاری که نسبت به نویز مقاوم باشد ولی توزیع دادگان را تغییر ندهد قابل توجیح است. بدین منظور ساختار خودرمزگذار انقباضی را مورد بررسی قرار می دهیم. در ساختار خودرمزگذار انقباضی همچون خودرمزگذار نویززدا هدف کاهش تغییرات بردار پنهان نسبت به تغییرات کوچک در ورودی است ولی در این ساختار به جای افزودن نویز به دادگان، در رابطه تابع هزینه ساختار مطابق رابطه ۱۰۰-۱۰ جمله ای اضافه می کنیم تا ساختار نسبت به نویز مقاوم شود. این جمله همان نرم فربینوس ماتریس ژاکوبین بردار پنهان نسبت به بردار ورودی است. با افزودن این جمله به تابع هزینه ساختار را وادار می کنیم تا نسبت به تغییرات بردار ورودی کمترین تغییر را در بردار پنهان داشته باشد، به عبارت دیگر گردایان ابعاد بردار پنهان نسبت به

[1] Contractive autoencoder

ورودی به حداقل ممکن برسد. در رابطه ۱۰-۱۰۰، λ پارامتر حساسیت با مقداری بین صفر تا یک است که حساسیت تابع هزینه نسبت به جمله مجموع گرادیان ها را تعیین می کند.

$$E = \frac{1}{2}(\mathbf{x} - \hat{\mathbf{x}})^2 + \lambda \left\| J_{(\mathbf{o}^h)} \right\|_F \qquad (۱۰-۱۰۰)$$

ماتریس ژاکوبین برای یک خودرمزگذار انقباضی با بردار پنهانی با بعد n طبق رابطه ۱۰-۱۰۱ تعریف می شود.

$$J_{(\mathbf{o}^h)} = \begin{bmatrix} \frac{\partial o_1^h}{\partial x_1} & \frac{\partial o_1^h}{\partial x_2} & \cdots & \frac{\partial o_1^h}{\partial x_n} \\ \frac{\partial o_2^h}{\partial x_1} & \ddots & \cdots & \vdots \\ \vdots & \cdots & \ddots & \vdots \\ \frac{\partial o_n^h}{\partial x_1} & \cdots & \cdots & \frac{\partial o_n^h}{\partial x_n} \end{bmatrix} \qquad (۱۰-۱۰۱)$$

برای محاسبه هر یک از درایه های ماتریس ژاکوبین رابطه ۱۰-۱۰۱ رابطه مشتق زنجیره ای ۱۰-۱۰۲ برقرار است.

$$\frac{\partial o_i^h}{\partial x_j} = \frac{\partial o_i^h}{\partial net_i^h} \frac{\partial net_i^h}{\partial x_j} = f'_{(net_i^h)} w_{i,j} \qquad (۱۰-۱۰۲)$$

در نهلیت برای جمله نرم فربینوس ماتریس ژاکوبین بردار پنهان رابطه ۱۰-۱۰۳ را داریم که در آن m بعد ورودی است.

$$\left\| J_{(\mathbf{o}^h)} \right\|_F = \sqrt{\sum_j^m \sum_i^n f'_{(net_i^h)} w_{i,j}} \qquad (۱۰-۱۰۳)$$

مزیت دیگر ساختار خودرمزگذار انقباضی نسبت به ساختار خودرمزگذار نویززدا این است که در ساختار نویززدا مقاومت به نویز در بردار خروجی، بازسازی، اعمال می شد و نسبت به بردار پنهان نظارتی اعمال نمی شد حال آن که هدف نهایی در این مدل بازنمائی تولید بردارهای پنهان است. در حالی که در ساختار خودرمزگذار انقباضی مقاومت نسبت به نویز در بردار پنهان اعمال می شود. در این ساختار نمی توان از ترانهاده وزن های کدگشا برای وزن های کدگذار، رابطه ۱۰-۸۲، در روند آموزش استفاده کرد و باید برای هر لایه روابط آموزشی مستقل تعریف کنیم. افزودن ترم

انقباضی تاثیری در آموزش وزن های لایه کدگشا ندارد ولی رابطه مشتق زنجیره ای لایه کدگذار مطابق رابطه ۱۰۴-۱۰ محاسبه می شود.

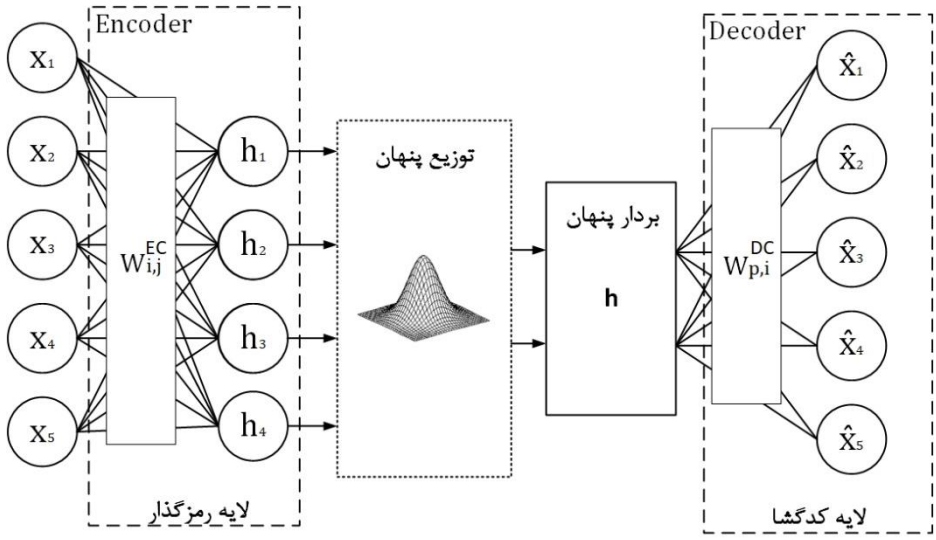
$$\frac{\partial E}{\partial W^{EC}} = \frac{\partial E'}{\partial e}\frac{\partial e}{\partial \hat{\mathbf{x}}}\frac{\partial \hat{\mathbf{x}}}{\partial \mathbf{net}^{DC}}\frac{\partial \mathbf{net}^{DC}}{\partial \mathbf{o}^{EN}}\frac{\partial \mathbf{o}^{EN}}{\partial \mathbf{net}^{EC}}\frac{\partial \mathbf{net}^{EC}}{\partial W^{EC}} + [1] f'_{(\mathbf{net}^h)} + \frac{f'_{(\mathbf{net}^h)}}{\partial \mathbf{net}^h}\frac{\partial \mathbf{net}^h}{\partial W^{EC}}(W^{EC})^T \quad (۱۰-۱۰۴)$$

در رابطه ۱۰۴-۱۰ منظور از [1] یک یک ماتریس هم اندازه با ماتریس وزن های لایه رمزگذار است که مقادیر همه درایه های آن یک است.

۱۰.۳.۱۰ خود رمزگذار متغیر[1]

یکی از چالش های عمده اشاره شده در بخش ۶-۳-۱۰ در ساختار خودرمزگذارهایی که تا به اینجا بررسی کردیم، مربوط به وجود بازه هایی در دامنه بردار پنهان است که خودرمزگذار نسبت به آن آموزشی نداشته است و بردار پنهان مناسبی در این بازه ها ایجاد نمی شود. همه این ساختارها از بردارهای پنهان گسسته برای تولید بردار بازسازی، خروجی، استفاده می کنند. برای رفع این چالش ساختار خودرمزگذارهای متغیر معرفی شده اند که یک توزیع احتمالاتی پنهان با یادگیری بدون نظارت برای مجموعه دادگان آموزشی ایجاد کرده و سپس از آن توزیع برای تولید بردارهای پنهان استفاده می کنند. در این ساختار نیز همانند ساختارهای خودرمزگذار پیشین خروجی مطلوب همان ورودی تعریف می شود که براساس بردار پنهان تولید می شود، بدین ترتیب هدف لایه رمزگذار محاسبه توزیع بردارهای پنهان بر اساس دادگان ورودی $P_{(z|x)}$ و هدف بردار کدگشا تولید بردار بازسازی (خروجی) بر اساس نمونه های تصادفی توزیع پنهان است. در این ساختار خروجی لایه رمزگذار پارامترهای توزیع پنهان هستند که بردارهای پنهان $\mathbf{z}\sim P_{(\mathbf{z})}$ از آن توزیع پنهان به صورت نمونه تصادفی انتخاب می شوند. هدف آموزش بدون نظارت ساختار خودرمزگذار متغیر همانند ساختارهای پیشین تولید بردار خروجی با حداکثر تشابه به بردار ورودی متناظر با آن ورودی است که برای تحقق آن باید توزیع مناسبی برای بردار های پنهان تعریف شود، بنابراین تنظیم پارامترهای توزیع پنهان نیز در کنار کاهش خطای بین بردار ورودی و بازسازی به عنوان هدف آموزش این ساختار در نظر گرفته می شود. شکل ۱۰-۱۴ ساختار کلی یک خودرمزگذار متغیر را نشان می دهد که در آن توزیع پنهان مشخص شده است.

[1] Variational autoencoder



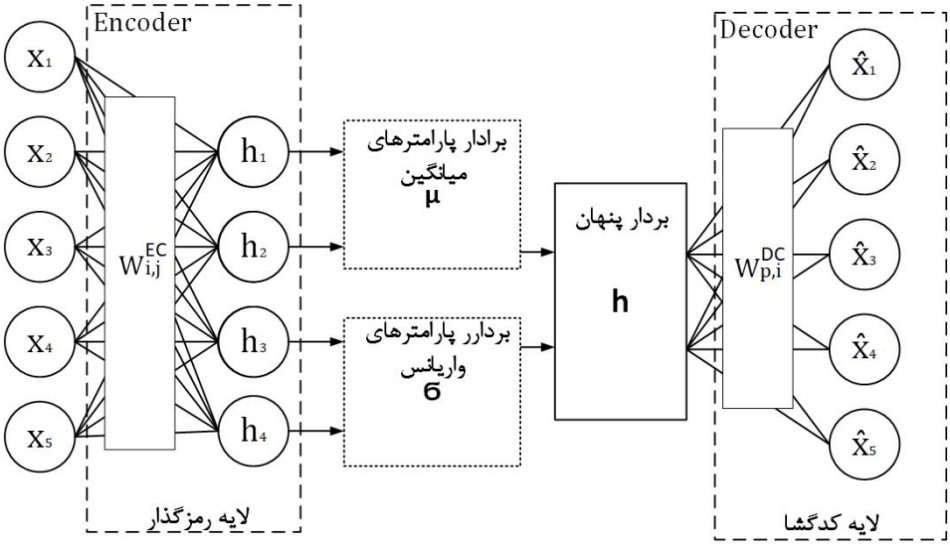
شکل ۱۴-۱۰: ساختار کلی خودرمزگذار متغیر

همان طور که در ماشین های بولتزمن با مدل کردن توزیع دادگان می توانستیم پس از آموزش ماشین از توزیع مدل شده آن برای ایجاد نمونه های تصادفی جدیدی که توزیعی مشابه توزیع داده های آموزش دارند تولید کنیم، در این ساختار نیز با مدل کردن توزیع پنهان می توانیم از آن توزیع نمونه های تصادفی به عنوان نمونه هایی که توزیعی بازنمائی از توزیع داده های اصلی دارند برای مقاصد دیگر استفاده کرد. بدین ترتیب قسمتی از فضا که در آن با استفاده از ساختارهای پیشین خودرمزگذار که تا کنون معرفی شده است، بردارهای پنهان به صورت نقاط گسسته متناظر با هر ورودی تعریف می شدند، تبدیل به یک دامنه احتمالاتی پیوسته شده و مشکل مربوط به وجود بازه های بین این نقاط گسسته که آموزشی در آن ها صورت نمی گرفت برطرف شده است. در ادامه به بررسی مثال هایی از کاربرد نمونه های تصادفی که از توزیع پنهان تولید می شوند خواهیم پرداخت.

برای روشن شدن بهتر عملکرد ساختار خودرمزگذار متغیر روند پیشرو و آموزش را در هر قسمت از ساختار به ترتیب زیر بررسی می کنیم:

۱- در ابتدا ساختار خودرمزگذار متغیر را تعریف می کنیم. در این ساختار تعداد نورون های لایه پنهان دو برابر بعد مدنظر برای بردار پنهان انتخاب می شود، دلیل این موضوع این است که برای مدل کردن توزیع پنهان، توزیع هر بعد از بردارهای پنهان $P(\boldsymbol{z}|\boldsymbol{x})$ را یک توزیع گوسی فرض می کنیم که هر بعد آن از سایر ابعاد مستقل است، توزیع پنهان دارای ماتریس کواریانس قطری است، و برای شبیه سازی هر یک از بعدهای این توزیع، از یک نورون برای پارامتر میانگین μ و از نورونی دیگر برای پارامتر واریانس σ توزیع استفاده می شود. در مورد دلایل فرض توزیع گوسی برای بردارهای پنهان را در ادامه با بررسی ساختار از دیدگاه احتمالاتی بیشتر بحث خواهیم کرد.

شـکل ۱۵-۱۰ یک خودرمزگذار متغیر را نشـان می دهد که دارای بردار پنهان با بعد $n = 2$ و تعداد نورون پنهان $2n = 4$ برای مدل کردن توزیع گوسی پنهان است.



شکل ۱۵-۱۰: ساختار خودرمزگذار متغیر با توزیع پنهان

۲- پس از توسـعه سـاختار از نورون های لایه پنهان که در مرحله قبل ایجاد شـد برای شـبیه سـازی توزیع پنهان مطابق رابطه ۱۰۵-۱۰ اسـتفاده می کنیم که در آن مقادیر میانگین و واریانس هر بعد از توزیع همان خروجی نورون های پنهان هسـتند. باید توجه داشته باشیم که ساختار خودرمزگذارهای متغیر از ساختار شبکه های عصبی با توابع فعال ساز شعاعی[1] متفاوت است. در شبکه های عصبی با توابع فعال ساز شعاعی مقدار احتمال محاسـبه شـده از توزیع گوسـی به ازای ورودی به عنوان خروجی نورون در نظر گرفته شـده و واریانس و میانگین توزیع، مانند وزن های شـبکه عصـبی پرسپترون، پارامترهای آموزشی شبکه هستند؛ در حالی که در ساختار خودرمزگذار متغیر خروجی نورون پارامترهای میانگین و واریانس توزیع هسـتند که برای ایجاد آن ها براساس ورودی از وزن ها و توابع فعال ساز مشابه شبکه های عصبی پرسپترون چند لایه استفاده می شود.

$$\mathbf{z} \sim \mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\sigma}^2) \qquad (۱۰-۱۰۵)$$

۳- از توزیع مدل شـده در مرحله قبل، نمونه بردار تصـادفی پنهان ایجاد می شـود، برای

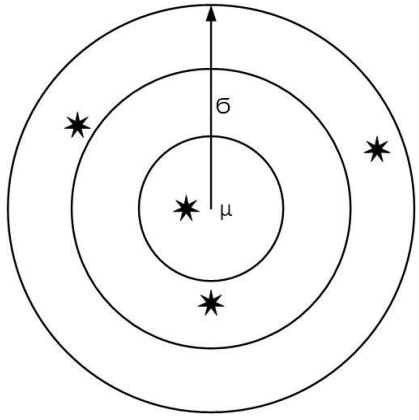
[1] Radial based function (RBF)

ایجاد این بردار پنهان، طبق رابطه ۱۰۶-۱۰ از توزیع مرحله قبل استفاده می کنیم، در رابطه ۱۰۶-۱۰، $\boldsymbol{\varepsilon}$ بردار نمونه‌ی تصادفی از یک نویز گوسی با میانگین صفر و وارایانس دلخواه $\mathcal{N}(0,\sigma^2)$ است. استفاده از مقادیر نویز برای تولید نمونه های تصادفی علاوه بر مقاوم کردن ساختار نسبت به نویز دادگان مشابه ساختار خودرمزگذارهای نویززدا، بازه مقادیر بردار پنهان نیز تعمیم می دهد، بدین ترتیب مشکل وجود بازه های بدون آموزش در سایر ساختارهای خودرمز گذار در این ساختار برطرف می شود. بدیهی است که با کنترل مقدار واریانس نویز گوسی در هر بعد، می توان بازه احتمالی انتخاب نمونه تصادفی از توزیع پنهان را بین مرکز توزیع و انتهای شعاع توزیع مطابق شکل ۱۶-۱۰ کنترل کرد. پس از تولید بردار پنهان در مرحله بعد لایه کدگشا بردار خروجی متناظر را برای این بردار تصادفی تولید می کند. بدین ترتیب روابط پیشرو خودرمزگذار متغیر با بعد ورودی n و بعد بردار پنهان m در لایه رمزگذار و کدگشا به ترتیب مطابق روابط ۱۰۸-۱۰ تا ۱۱۱-۱۰ است.

$$\mathbf{z} = \boldsymbol{\mu} + \boldsymbol{\sigma} \times \boldsymbol{\varepsilon} \qquad (۱۰-۱۰۶)$$

در رابطه ۱۰۶-۱۰ هر یک از پارامترهای میانگین و واریانس به صورت برداری تعریف می شوند که با ضرب در بردار نویز گوسی نرمال $\boldsymbol{\varepsilon}$ که به صورت یک نمونه تصادفی از توزیع رابطه ۱۰۷-۱۰ تعریف می شود.

$$\varepsilon \sim \mathcal{N}(0,1^2) \qquad (۱۰-۱۰۷)$$



شکل ۱۶-۱۰: تاثیر نویز در انتخاب نمونه از توزیع پنهان

$$\mathbf{net}^{EC}_{1\times 2m} = \mathbf{x}^{T}_{n\times 1} \times W^{EC}_{n\times 2m} \qquad (۱۰-۱۰۸)$$

$$\mathbf{net}^{DC}_{1\times n} = \mathbf{z}_{1\times m} \times W^{DC}_{m\times n} \qquad (۱۰-۱۰۹)$$

$$\mathbf{o}_{1\times 2m}^{EC} = f(\mathbf{net}^{EC}) \quad (۱۰-۱۱۰)$$

$$\mathbf{o}_{1\times n}^{DC} = f(\mathbf{net}^{DC}) \quad (۱۰-۱۱۱)$$

۴- برای بردار پنهان تصادفی ایجاد شده از یک بردار ورودی مقدار مطلوب همان بردار ورودی اصلی است بنابراین برای تعریف تابع هزینه خودرمزگذار متغیر از همان فاصله اقلیدسی بین بردار ورودی و بردار بازسازی استفاده می کنیم. از طرفی در تعریف ساختار این خودرمزگذار، پارامترهای یک توزیع را به عنوان خروجی لایه رمزگذار در نظر گرفتیم. مسئله قابل تامل در اینجا این است که علیرغم فرض ما برای خروجی لایه رمزگذار به عنوان پارامترهای توزیع این پارامترها در عمل مشابه یک لایه پرسپترون عمل کرده و برای هر بردار ورودی یک بردار پنهان نسبت داده و تفاوتی با ساختارهای پیشین ندارند، برای رفع این مشکل و تبدیل خروجی لایه رمزگذار به پارامترهای توزیع یک توزیع هدف $p_{(\mathbf{z})}$ برای بردارهای پنهان در نظر گرفته و تفاوت پارامترهای لایه پنهان، خروجی لایه رمزگذار، با این توزیع هدف را به تابع هزینه وارد می کنیم. بدین ترتیب لایه رمزگذار به جای آن که به هر بردار ورودی یک بردار پنهان نسبت دهد، مطابق شکل ۱۴-۱۰ و ۱۵-۱۰ یک توزیع گوسی نسبت می دهد. با توجه به فرض توزیع گوسی برای لایه رمزگذار توزیع هدف را در ساده ترین حالت یک توزیع گوسی نرمال که ابعاد آن مستقل از هم هستند $P_{(\mathbf{z})} = \mathcal{N}(0,1)$ تعریف می کنیم، البته می توان برای توزیع از توزیع ساده دیگری مانند توزیع یکنواخت نیز استفاده کرد ولی استفاده از توزیع گوسی نرمال علاوه بر این که تناسب بیشتری با توزیع پنهان مدل $P_{(\mathbf{z}|\mathbf{x})}$ دارد باعث می شود تا بردارهای پنهان در قسمتی از فضا، مبدا، تجمع بیشتری داشته و نمونه تصادفی بردار پنهان انتخابی همواره از بازه کوچکتری انتخاب شود تا روند آموزش پایدارتر باشد. با این توضیحات یک جمله مربوط به هم پوشانی توزیع لایه رمزگذار و توزیع هدف که با معیار KL سنجیده می شود به تابع هزینه اضافه می کنیم. بدین ترتیب باید خروجی نورون های لایه پنهان در طی روند آموزش به سمت مقادیر مطلوب خود، $\mu \to 0, \sigma \to 1$، سوق داده شوند. بدین ترتیب تابع هزینه نهایی مطابق رابطه ۱۱۲-۱۰ خواهد بود.

$$E = \frac{1}{2}(\mathbf{x}-\hat{\mathbf{x}})^2 + KL(P_{(\mathbf{z})} \,||\, P_{(\mathbf{z}|\mathbf{x})}) \quad (۱۰-۱۱۲)$$

۵- پس از محاسبه مقدار تابع هزینه به ازای بردار ورودی روابط آموزش به روش گرادیان

نزولی را برای لایه کدگشـــا و رمز گذار طبق روابط ۱۱۳-۱۰ تا ۱۱۶-۱۰ توســعه می دهیم. لازم به ذکر اســـت که با توجه به این که لایه رمزگذار پارامترهای توزیع را به عنوان خروجی ســـاخته و لایه کدگشـــا براســـاس یک نمونه تصـــادفی از توزیع بردار بازســـازی، خروجی، را می ســـازد، علاوه بر این که ماهیت عملکردی دو لایه برخلاف ساختارهایی که قبلا معرفی شده اند برعکس هم نیست، اندازه ماتریس وزن های لایه رمزگذار و کدگشـــا نیز به ترتیب و بوده و با هم برابر نیســـتند، بنابراین نمی توان از رابطه ۸۲-۱۰ در این ساختار استفاده کرد و باید روابط مشتق زنجیره ای برای هر دو لایه رمزگذار و کدگشا تعریف شود.

$$\frac{\partial E}{\partial W^{DC}} = \frac{\partial E}{\partial e}\frac{\partial e}{\partial \hat{\mathbf{x}}}\frac{\partial \hat{\mathbf{x}}}{\partial net^{DC}}\frac{\partial net^{DC}}{\partial W^{EC}} \qquad (۱۰-۱۱۳)$$

$$W^{DC}_{(k+1)} = W^{DC}_{(k)} - \eta \frac{\partial E}{\partial W^{DC}}(k) \qquad (۱۰-۱۱۴)$$

$$\frac{\partial E}{\partial W^{EC}} = \frac{\partial E}{\partial e}\frac{\partial e}{\partial \hat{\mathbf{x}}}\frac{\partial \hat{\mathbf{x}}}{\partial net^{DC}}\frac{\partial net^{DC}}{\partial \mathbf{z}}\frac{\partial \mathbf{z}}{\partial \mathbf{o}^{EN}}\frac{\partial \mathbf{o}^{EN}}{\partial net^{EC}}\frac{\partial net^{EC}}{\partial W^{EC}} \qquad (۱۰-۱۱۵)$$

$1+\varepsilon$

$$W^{EC}_{(k+1)} = W^{EC}_{(k)} - \eta \frac{\partial E}{\partial W^{EC}}(k) \qquad (۱۰-۱۱۶)$$

همچنین برای آموزش بایاس های لایه کدگشـــا و رمزگذار روابط ۱۱۷-۱۰ تا ۱۲۰-۱۰ برقرار است:

$$\frac{\partial E}{\partial \mathbf{b}^{DC}} = \frac{\partial E}{\partial e}\frac{\partial e}{\partial \hat{\mathbf{x}}}\frac{\partial \hat{\mathbf{x}}}{\partial net^{DC}}\frac{\partial net^{DC}}{\partial \mathbf{b}^{DC}} \qquad (۱۰-۱۱۷)$$

$$\mathbf{b}^{DC}_{(k+1)} = \mathbf{b}^{DC}_{(k)} - \eta \frac{\partial E}{\partial \mathbf{b}^{DC}}(k) \qquad (۱۰-۱۱۸)$$

$$\frac{\partial E}{\partial \mathbf{b}^{EC}} = \frac{\partial E}{\partial e}\frac{\partial e}{\partial \hat{\mathbf{x}}}\frac{\partial \hat{\mathbf{x}}}{\partial net^{DC}}\frac{\partial net^{DC}}{\partial \mathbf{z}}\frac{\partial \mathbf{z}}{\partial \mathbf{o}^{EN}}\frac{\partial \mathbf{o}^{EN}}{\partial net^{EC}}\frac{\partial net^{EC}}{\partial \mathbf{b}^{EC}} \qquad (۱۰-۱۱۹)$$

$1+\varepsilon$

$$\mathbf{b}^{EC}_{(k+1)} = \mathbf{b}^{EC}_{(k)} - \eta \frac{\partial E}{\partial \mathbf{b}^{EC}}(k) \qquad (۱۰-۱۲۰)$$

۱۰.۳.۱۰.۱ خودرمزگذار متغیر از دیدگاه احتمالاتی

هدف یک خودرمزگذار ایجاد بردارهای بازسازی $\hat{\boldsymbol{x}}$ براساس بردارهای $\boldsymbol{z}$ پنهان است. بر این اساس می توانیم مدل خودرمزگذار را با استفاده از قاعده بیز[1] که در فصل ۶ معرفی کردیم طبق رابطه ۱۲۱-۱۰ بنویسیم.

$$P_{(\mathbf{z}|\mathbf{x})} = \frac{P_{(\mathbf{x}|\mathbf{z})} P_{(\mathbf{z})}}{P_{(\mathbf{x})}} \tag{۱۰-۱۲۱}$$

در رابطه ۱۲۱-۱۰ برای محاسبه احتمال شواهد[2] $P_{(x)}$ در مخرج رابطه ۱۲۲-۱۰ را داریم:

$$P_{(\mathbf{x})} = \int P_{(\mathbf{x}|\mathbf{z})} P_{(\mathbf{z})} d\mathbf{z} \tag{۱۰-۱۲۲}$$

در رابطه ۱۲۲-۱۰ انتگرال فرم بسته نداشته و محاسبه آن بسیار دشوار است، از این رو به جای محاسبه توزیع $P_{(\boldsymbol{z}|\boldsymbol{x})}$ رابطه ۱۲۱-۱۰ برای توزیع پنهان یک توزیع گوسی $q_{(\mathbf{z}|\mathbf{x})} = \mathcal{N}(\mu, \sigma^2)$ تقریب می زنیم که این توزیع برای هر بعد از بردار پنهان تعریف می شود؛ هر یک از ابعاد بردار پنهان دارای توزیع گوسی است که مستقل از بعد دیگر بوده و ماتریس کواریانس توزیع قطری است. این تقریب گوسی همان توزیعی است که خروجی لایه رمزگذار در ساختار پارامترهای آن را شبیه سازی می کند.

از طرفی هدف از آموزش ساختار خودرمزگذار ایجاد بردارهای خروجی به گونه ای است که این بردارهای بیشترین شباهت را به بردار ورودی داشته باشند، به بیان احتمالاتی هدف آموزش به حداکثر رساندن لگاریتم تابع درست نمایی برای بردار $\boldsymbol{x}$ در خروجی مطابق رابطه ۱۲۳-۱۰ است.

$$\log P_{\theta(\mathbf{x})} = \log \mathbb{E}_{p_{(\mathbf{z})}} (P_{\theta(\mathbf{z}|\mathbf{x})}) \tag{۱۰-۱۲۳}$$

برای هر تابع محدب، تابعی که نقطه کمینه محلی[3] و سراسری[4] آن یکسان باشد، نامساوی جنسون[5] را می توان به دو صورت طبق روابط ۱۲۴-۱۰ و ۱۲۵-۱۰ تعریف کرد. شکل ۱۷-۱۰ نمایش نامساوی جنسون برای یک تابع محدب است. در رابطه ۱۲۴-۱۰، t یک اسکالر بین صفر تا یک α, β دو مقدار برای ورودی تابع و در رابطه ۱۲۵-۱۰، E_x امید ریاضی شرطی نسبت به ورودی های x تابع است.

$$f_{(t\alpha+(1-t)\beta)} \geq t f_{(\alpha)} + (1-t) f_{(\beta)} \tag{۱۰-۱۲۴}$$

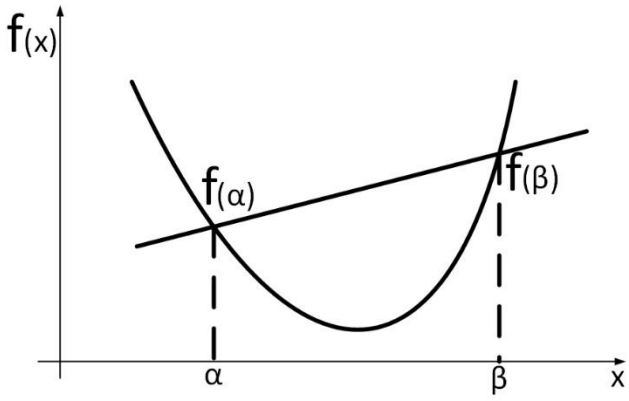
[1] Bayes' theorem
[2] Evidence probability
[3] Local minimum
[4] Global minimum
[5] Jenson's inequality

$$f_{(\mathrm{E}_x(x))} \geq \mathrm{E}_x f_{(x)} \qquad (۱۰-۱۲۵)$$

برای توابع مقعر (تابعی که نقطه بیشینه محلی[1] و سراسری[2] آن یکسان باشد) نامساوی جنسون به صورت قرینه روابط ۱۰-۱۲۴ و ۱۰-۱۲۵، ضرب طرفین در منفی یک، به صورت رابطه ۱۰-۱۲۶ و ۱۰-۱۲۷ تعریف می شـــود. بدین ترتیب می توان چنین نتیجه گرفت که تابع مقعر قرینه تابع محدب متناظر با آن است.

$$f_{(t\alpha+(1-t)\beta)} \leq t f_{(\alpha)} + (1-t) f_{(\beta)} \qquad (۱۰-۱۲۶)$$

$$f_{(\mathrm{E}_x(x))} \leq \mathrm{E}_x f_{(x)} \qquad (۱۰-۱۲۷)$$



شکل ۱۷-۱۰: نامساوی جنسون

رابطه ۱۰-۱۲۳ یک تابع درسـت نمایی توزیع احتمالاتی اسـت که رفتار آن در بازه صـفر تا یک است، بنابراین این تابع یک تابع مقعر بوده و برای تعریف نامساوی جنسون برای آن از رابطه ۱۰-۱۲۷ اســـتفاده می کنیم که مطابق رابطه ۱۰-۱۲۸ تعریف می شـــود. با توجه به این که هدف به حداکثر رساندن تابع درست نمایی در طول روند آموزش ساختار است، با توجه به رابطه ۱۰-۱۲۸ با به افزایش دادن سمت راست نامساوی به سمت مقدار حداکثری یک مقدار حداقل سمت چپ نیز تحت تاثیر سـمت راسـت نامسـاوی به سـمت یک میل می کند. در این رویکرد بهینه سـازی سـمت

[1] Local maximum
[2] Global maximum

راست نامساوی جنسون با نام باند پایین شاهد[1] شناخته می شود که هدف به حداکثر رساندن آن است.

$$\log P_{\theta(\mathbf{x})} = \log \mathrm{E}_{p_{(\mathbf{z})}}(P_{\theta(\mathbf{z}|\mathbf{x})}) \geq \mathrm{E}_{p_{(\mathbf{z})}}(\log(P_{\theta(\mathbf{z}|\mathbf{x})})) \quad (۱۰-۱۲۸)$$

برای این که رابطه ۱۰-۱۲۸ همواره در قسمت مثبت فضا باشد قرینه آن را طبق رابطه ۱۰-۱۲۹ تعریف کرده و هدف آموزش ساختار را به حداقل رساندن آن در نظر می گیریم:

$$\arg\max_{} \mathrm{E}_{p_{(\mathbf{z})}}(\log(P_{\theta(\mathbf{z}|\mathbf{x})})) \rightarrow \arg\min_{} \mathrm{E}_{p_{(\mathbf{z})}}(-\log(P_{\theta(\mathbf{z}|\mathbf{x})})) \quad (۱۰-۱۲۹)$$

از طرفی در قسمت قبل یک توزیع گوسی نرمال $p_{(\mathbf{z})} = \mathcal{N}(0,1)$ به عنوان توزیع هدف برای توزیع پنهان مدل مطرح کردیم در اینجا نیز هم پوشانی توزیع مدل $Q_{(\mathbf{z}|\mathbf{x})} = \mathcal{N}(\mu, \sigma^2)$ با معیار KL محاسبه می شود. این معیار نیز همواره مثبت بوده و هدف به حداقل رساندن آن است، بنابراین این معیار را نیز به رابطه ۱۰-۱۲۹ اضافه کرده در نهایت تابع هزینه ساختار را مطابق رابطه ۱۰-۱۳۰ تعریف می کنیم.

$$E = \mathbf{E}_{p_{(\mathbf{z})}}(-\log(P_{\theta(\mathbf{z}|\mathbf{x})})) + KL(P_{(\mathbf{z})} \,||\, Q_{(\mathbf{z}|\mathbf{x})}) \quad (۱۰-۱۳۰)$$

از طرفی لگاریتم تابع درست نمایی برای این ساختار را می توانیم به صورت رابطه ۱۰-۱۳۱ و ۱۰-۱۳۲ تعریف کنیم، در رابطه ۱۰-۱۳۱ و ۱۰-۱۳۲، μ مقدار میانگین و σ واریانس دادگان است که با توجه به این که برای دادگان نویز گوسی با میانگین صفر و واریانس فرضی در نظر گرفته ایم، دو جمله اول رابطه ۱۰-۱۳۲ همواره مقدار ثابتی دارند و فقط جمله سوم که همان مجموع مربعات خطا است در مقدار نهایی تابع هزینه تاثیر دارد (می توان از واریانس مخرج جمله سوم از آنجایی که همواره مقدار ثابتی دارد صرف نظر کرد)، بنابراین رابطه تابع هزینه ۷-۱۲۲ را به صورت رابطه ۱۰-۱۳۳ بازنویسی می کنیم. با مقایسه رابطه ۷-۱۳۳ و ۷-۱۱۲ می توانیم برابری این دو رابطه را مشاهده کنیم.

$$\log(P_{\theta(\mathbf{z}|\mathbf{x})}) = \log\left(\frac{1}{\sqrt{2\pi\sigma^2}} e^{\left(-\frac{1}{2}\frac{(\mathbf{x}-\hat{\mathbf{x}})^2}{\sigma^2}\right)}\right) \quad (۱۰-۱۳۱)$$

$$\log\left(\frac{1}{\sqrt{2\pi\sigma^2}} e^{\left(-\frac{1}{2}\frac{(\mathbf{x}-\hat{\mathbf{x}})^2}{\sigma^2}\right)}\right) = -\log\sigma - \log 2\pi - \frac{(\mathbf{x}-\hat{\mathbf{x}})^2}{2\sigma^2} \quad (۱۰-۱۳۲)$$

[1] Evidence Lower Bound (ELBO)

$$E = \frac{1}{2}(\mathbf{x} - \hat{\mathbf{x}})^2 + KL(P_{(\mathbf{z})} \,||\, Q_{(\mathbf{z}|\mathbf{x})})$$

(۱۰-۱۳۳)

۱۰.۳.۱۰.۲ بهبود عملکرد در خودرمزگذارهای متغیر

برای بهبود عملکرد ساختار خودرمزگذارهای متغیر راهکارهایی مطرح شده است که به چند مورد از مهم ترین آنها اشاره می کنیم:

۱- استفاده از توزیع غیرصریح برای تقریب توزیع شرطی $p_{(z|x)}$:

پیشتر بررسی کردیم که به دلیل دشواری محاسبه انتگرال رابطه ۱۰-۱۲۲ در ساختار خودرمزگذارهای متغیر از یک توزیع تقریب گوسی استفاده می کنیم. برای بهبود عملکرد ساختار و افزایش دقت تقریب توزیع می توان به جای توزیع گوسی از توزیع های ضمنی[1]، غیر پارامتری[2]، استفاده کرد. در این صورت نورون های لایه رمزگذار توزیع غیرپارمتری را شبیه سازی می کنند. به دلیل ماهیت این توزیع و فقدان پارامترهای مشخص، مانند میانگین و واریانس، رابطه ای نمی توان برای توزیع ضمنی توسعه داد از این رو نمی توان از معیار واگرائی KL برای مقایسه آن با توزیع هدف $p_{(\mathbf{z})}$ در تابع هزینه استفاده کرد. برای رفع این مشکل از یک متمایزکننده[3] مشابه آنچه در شبکه های مولد متخاصم[4] که در فصل ۱۳ بررسی شده است، استفاده می شود برای تشخیص هم پوشانی توزیع هدف و توزیع ضمنی مطابق رابطه ۱۰-۱۳۴ استفاده می کنیم. در رابطه ۱۰-۱۳۴، یک متمایزکننده است. مقدار محاسبه شده از رابطه ۱۰-۱۳۴ به جای جمله KL در رابطه ۱۰-۱۱۲ تابع هزینه ساختار استفاده می شود.

$$T = \log D(P_{(\mathbf{z})}) + \log(1 - D(Q_{(\mathbf{z}|\mathbf{x})}))$$

(۱۰-۱۳۴)

در این حالت متمایزکننده D یک مدل است که می تواند دارای پارامترهای آموزشی نیز باشد، شبکه عصبی پرسپترون چند لایه، در این صورت برای سنجش بهتر هم پوشانی پس از اینکه مقدار رابطه ۱۰-۱۳۴ در تابع هزینه خودرمزگذار اعمال شد، پارامترهای متمایزکننده نیز در کنار پارامترهای ساختار خودرمزگذار آموزش داده می شود تا در طی روند آموزش سنجش ارائه شده توسط آن بهبود یابد. مقدار کمینه تابع مدل متمایزکننده براساس رابطه ۱۰-۱۳۴ به عنوان تابع هزینه برای مدل متمایزکننده شناخته می شود.

---

[1] Implicit distribution
[2] Non-parametric
[3] Discriminator
[4] Generative adversarial networks (GAN)

۲- استفاده از توزیع های پیچیده تر برای توزیع پیشین هدف $P_{(z)}$ با نرمال سازی جریان[1]:

در توضیحات قبلی اشاره کردیم که برای توزیع هدف بردارهای پنهان $P_{(z)}$ از یک توزیع ساده گوسی نرمال استفاده می کنیم، برای ارتقا عملکرد ساختار می توان از توزیع پیچیده تری برای این توزیع استفاده کرد. برای انتقال توزیع هدف $P_{(z)}$ به یک توزیع دیگر رابطه ۱۳۵-۱۰ را داریم. در رابطه ۱۳۵-۱۰، $f_{(.)}$ یک تابع معکوس پذیر، تابع یک به یک، و $f_{(.)}^{-1}$ معکوس آن است، همچنین $\left|\frac{\partial f_{(\mathbf{z}')}}{\partial \mathbf{z}'}\right|^{-1}$ معکوس دترمینان ماتریس ژاکوبین $f_{(.)}$ نسبت به ورودی، نمونه های توزیع حاصل $P_{(\acute{z})}$، است. معکوس دترمینان ماتریس ژاکوبین در رابطه نسبت حجم[2] توزیع اولیه $P_{(z)}$ و حاصل $P_{(\acute{z})}$ است که برای تغییر حجم توزیع اولیه به توزیع حاصل اعمال می شود. در صورتی مقدار این دترمینان یک باشد حجم توزیع اولیه و حاصل با هم برابر است. همچنین بعد توزیع اولیه و حاصل نیز با هم برابر است.

$$P_{(\mathbf{z}')} = P_{(\mathbf{z})}\left|\frac{\partial f^{-1}_{(\mathbf{z}')}}{\partial \mathbf{z}'}\right| = P_{(\mathbf{z})}\left|\frac{\partial f_{(\mathbf{z}')}}{\partial \mathbf{z}'}\right|^{-1} \qquad (۱۰-۱۳۵)$$

رابطه ۱۲۷-۷ با عنوان نرمال سازی جریان شناخته می شود. دلیل این نام گذاری این است که این روش در ابتدا برای محاسبات تغییرات ایجاد شده با تغییر سطح مقطع مسیرهای عبوری سیالات ارائه شده بود، بدین ترتیب منظور از جریان اعمال نگاشت های برگشت پذیر به دادگان است به صورتی که پس از اعمال نگاشت توزیع حاصل نیز با اعمال نسبت حجم دو توزیع نرمال سازی شود.

- **نرمال سازی جریان به روش Planar Flow (PF)**

برای نرمال سازی به روش PF رابطه ۱۳۶-۱۰ را به عنوان تابع معکوس پذیر تعریف می کنیم. در رابطه ۱۳۶-۱۰، $f_{(.)}$ یک تابع معکوس پذیر، برای مثال یک تابع تانژانت هیپربولیک $f_{(.)} = \tan^{-1}(.)$ است، و u, b, w پارامترهایی با ابعاد مناسب برای بردار نمونه‌ها هستند که باید به گونه ای تنظیم شوند تا شرط معکوس پذیری رابطه همواره برقرار باشد.

$$\mathbf{z}' = \mathbf{z} + uf(w\mathbf{z} + b) \qquad (۱۰-۱۳۶)$$

$$\left|\frac{\partial f_{(\mathbf{z}')}}{\partial \mathbf{z}'}\right| = |1 + f'(wz + b)uw| \qquad (۱۰-۱۳۷)$$

[1] Flow normalization
[2] Distribution volume

رابطه ۱۰-۱۳۷ نیز دترمینان ماتریس ژاکوبین برای روش PF است. برای نرمال سازی جریان به روش PF رابطه ۱۰-۱۳۶ را در رابطه ۱۰-۱۳۵ به کار برده و توزیع حاصل از توزیع اولیه گوسی نرمال را به عنوان توزیع هدف در رابطه تابع هزینه ساختار استفاده می کنیم.

- **نرمال سازی جریان به روش Nonlinear Independent Components Estimation (NICE)**

در رابطه ۱۰-۱۳۷ با وجود این که از نظر تئوری می توان معکوس تابع تانژانت هیپربولیک را محاسبه کرد اما محاسبات مربوط به معکوس دترمینان ماتریس ژاکوبین طبق رابطه دشوار است برای رفع این مشکل روش $NICE$ ارائه شده است. در این روش توزیع اولیه ابتدا به دو زیرمجموعه جدا از هم افراز می شود، مجموعه بردارهای پنهان $\mathbf{z}$ را به دو زیرمجموعه تعداد عضو برابر $\mathbf{z_1}$ و $\mathbf{z_2}$ تقسیم می کنیم به طوری که $\mathbf{z_1} \cup \mathbf{z_2} = \mathbf{z}$. سپس به جای استفاده از رابطه ۱۰-۱۳۶ برای هر یک از دو مجموعه با تعداد عضو برابر $\mathbf{z_1}$ و $\mathbf{z_2}$ نگاشت های معکوس پذیر زیر به ترتیب طبق روابط ۱۰-۱۳۸ و ۱۰-۱۳۹ اعمال می شود. در رابطه ۱۰-۱۳۹، m_θ یک تابع پیچیده، یک شبکه عصبی با توابع فعال ساز $ReLU$، است که معکوس آن نیز قابل محاسبه است. در این روش به جای اعمال نگاشت به هر نمونه جدا از توزیع اولیه ($\mathbf{z} \to \acute{\mathbf{z}}$) توزیع به جفت نمونه اعمال شده و متعاقباً دو نمونه از آن حاصل می شود ($\langle \mathbf{z_1}, \mathbf{z_2} \rangle \to \langle \acute{\mathbf{z}_1}, \acute{\mathbf{z}_2} \rangle$).

$$\mathbf{z}'_1 = \mathbf{z_1} \qquad (۱۰-۱۳۸)$$

$$\mathbf{z}'_2 = \mathbf{z}_2 + m_\theta(\mathbf{z_1}) \qquad (۱۰-۱۳۹)$$

در روش $NICE$ حالت کلی ماتریس ژاکوبین مطابق رابطه ۱۰-۱۴۰ است که دشواری محاسباتی روش گذشته را ندارد.

$$\left|\frac{\partial f_{(\mathbf{z}')}}{\partial \mathbf{z}'}\right| = \begin{vmatrix} I_{\mathbf{z_1}} & 0 \\ \frac{\partial m_{(\mathbf{z}'_1)}}{\partial \mathbf{z}'_1} & I_{\mathbf{z_2}} \end{vmatrix} \qquad (۱۰-۱۴۰)$$

- **نرمال سازی جریان به روش Real Non-Volume Preserving (RealNVP)**

روش $RealNVP$ شباهت زیادی به روش $NICE$ دارد و مانند آن نمونه های توزیع اولیه را به دو زیرمجموعه با تعداد عضو برابر افراز می کند و روابط آن برای هر جفت نمونه به صورت $\langle \mathbf{z_1}, \mathbf{z_2} \rangle \to \langle \acute{\mathbf{z}_1}, \acute{\mathbf{z}_2} \rangle$ تعریف می شود . در روش $RealNVP$ برای محاسبه برای نگاشت مجموعه

$\mathbf{z_1}$ همان رابطه ۱۳۸-۱۰ روش $NICE$ استفاده می شود. برای نگاشت مجموعه $\mathbf{z_2}$ نیز رابطه ۱۴۱-۱۰ را داریم. در رابطه ۱۴۱-۱۰، $\odot$ نماد بیانگر ضرب درایه به درایه، هادامارد[1]، m_θ و s_θ توابع پیچیده ای، شبکه های عصبی پیچشی، هستند که اعضای مجموعه به عنوان ورودی آن ها در نظر گرفته می شود.

$$\mathbf{z}'_2 = e^{s_\theta(\mathbf{z}_1)} \odot \mathbf{z}_2 + m_\theta(\mathbf{z_1}) \tag{۱۰-۱۴۱}$$

برای حالت کلی دترمینان ژاکوبین روش $RealNVP$ رابطه ۱۴۲-۱۰ را داریم:

$$\left|\frac{\partial f_{(\mathbf{z}')}}{\partial \mathbf{z}'}\right| = \begin{vmatrix} I_{\mathbf{z_1}} & 0 \\ \frac{\partial s_{\theta(\mathbf{z}'_1)}}{\partial \mathbf{z}'_1} e^{s_\theta(\mathbf{z}_1)} \odot \mathbf{z}_2 + \frac{\partial m_{\theta(\mathbf{z}'_1)}}{\partial \mathbf{z}'_1} & e^{s_\theta(\mathbf{z}_1)} \end{vmatrix} \tag{۱۰-۱۴۲}$$

۳- استفاده از بردارهای پنهان تصادفی وزن دار:

در ساختار خودرمزگذار متغیر پس از ایجاد نمونه بردار پنهان تصادفی از توزیع احتمالاتی پنهان، لایه کدگشا از نمونه برای ساخت بردار بازسازی استفاده کرده و در نهایت مجموع مربعات خطا، فاصله اقلیدسی بین بردار بازسازی و ورودی، در تابع هزینه محاسبه می شد. در این روش وزن مقدار خطا، فاصله اقلیدسی، برای همه بردارهای حاصل از نمونه های تصادفی که از توزیع پنهان انتخاب می شدند یکسان در نظر گرفته می شد، در حالی که در واقع نمونه های تصادفی که از فاصله نزدیک تری نسبت به مرکز توزیع گوسی پنهان انتخاب می شوند، دارای مقدار احتمال $P_{(\mathbf{z}|x)}$ بزرگتر هستند، معمولا دارای خطای بازسازی کمتری نسبت به نمونه هایی که از فاصله دورتری نسبت به مرکز انتخاب می شوند هستند زیرا مقادیر بردار پنهان این نمونه های به مقادیر خروجی لایه رمزگذار نزدیکتر هستند. بر همین اساس کاهش خطای حاصل از نمونه های بردار پنهان با احتمال بزرگتر، نسبت به نمونه بردارهایی که از مقدار احتمال کوچکتری برخوردار هستند اهمیت کمتری دارد و برای داشتن توزیع احتمالاتی پنهان مناسبی که بازه گسترده‌ای را برای بردارهای پنهان پوشش دهد لازم است بر کاهش خطای نمونه هایی با احتمال پنهان کمتر، دورتر از مرکز، تمرکز کنیم و به خطای بازسازی آن ها اهمیت بیشتری در تابع هزینه بدهیم. برای در نظر گرفتن این موضوع در تابع هزینه در محاسبه میانگین مجموع مربعات خطا از میانگین وزن دار استفاده می کنیم. وزن هر بردار خروجی مطابق رابطه ۱۴۳-۱۰ که در رابطه ۱۰-

[1] Hadamard's product

۱۴۴ نسبت به وزن همه نمونه های تابع نرمال می شود محاسبه می شود که طبق رابطه ۱۰-۱۴۵ به بردارهای خروجی که احتمال پنهان بزرگتری دارند وزن کمتر و به بردارها با احتمال پنهان کمتر وزن بیشتری در تابع هزینه اعمال می شود، بدین ترتیب تابع هزینه ساختار مطابق رابطه ۱۰-۱۴۶ خواهد بود که در آن $\mathbf{E}_{\hat{w}}$ امید ریاضی شرطی نسبت به وزن های اعمال به ازای همه نمونه‌ها شده است.

$$w = \frac{1}{q_{(\mathbf{z}|\mathbf{x})}} \tag{۱۰-۱۴۳}$$

$$\hat{w} = \frac{w}{\sum_{n} w} \tag{۱۰-۱۴۴}$$

$$e = \hat{w}(\mathbf{x} - \hat{\mathbf{x}})^2 \tag{۱۰-۱۴۵}$$

$$E = \frac{1}{2}\mathbf{E}_{\hat{w}}(\mathbf{x} - \hat{\mathbf{x}})^2 + KL(P_{(\mathbf{z})} \,||\, q_{(\mathbf{z}|\mathbf{x})}) \tag{۱۰-۱۴۶}$$

با توجه به ماهیت آماری و استفاده از امید ریاضی در تابع هزینه بهتر است به جای استفاده از روش گرادیان نزولی از روش های آموزش احتمالاتی که از مقدار خطای حاصل چند نمونه برای آموزش استفاده می کنند، مانند گرادیان نزولی تصادفی[1]، که در فصل ۹ بررسی شده است استفاده کرد تا مقدار وزن های اعمال شده به خطاهای بازسازی هر نمونه به ازای جامعه آماری بزرگتری محاسبه شده و تاثیر بیشتری داشته باشند.

۴- تقویت لایه کدگشا و وابسته کردن ابعاد بردار خروجی به هم:

برای بهبود عملکرد ساختار خودرمزگذار متغیر می توان در لایه کدگشا تغییراتی اعمال کنیم تا بردارهای بازسازی با خطای کمتری ایجاد شوند. یک از راهکارهای ارائه شده بدین منظور این است که به جای اینکه همه ابعاد یک بردار خروجی را با اعمال وزن ها و نگاشت تابع فعال ساز در لایه کدگشا ایجاد کنیم، ابعاد بردارباز‌سازی را به هم وابسته ساخته و هر بعد از بردار خروجی را نسبت به بعدهای قبل و نمونه بردار پنهان به صورت سلسله مراتبی تولید می کنیم. در این روش ابتدا بعد اول بردار خروجی را بر اساس بردار پنهان تولید کرده و سپس بعد های دیگر بردار پنهان را براساس ابعاد قبلی و بردار پنهان ایجاد می کنیم. بدین ترتیب توزیع لایه کدگشا که به جای $P_{(\mathbf{x}|\mathbf{z})}$ از رابطه ۱۰-۱۴۷ تبعیت می کند. بدیهی است که در رابطه ۱۰-۱۴۷ بعد اول بردار بازسازی فقط به بردار پنهان وابسته بوده و براساس آن تشکیل می شود. استفاده از این روش به ویژه در مواردی که ابعاد نمونه ها در

[1] Stochastic gradient descent (SGD)

دادگان به هم وابسته هستند مانند دادگان تصویری و یا سری های زمانی (ابعاد بردار خروجی دینامیک های زمانی مختلف یک سری زمانی هستند) باعث بهبود چشم گیر در عملکرد ساختار می شود.

$$P_{(x_i|x_1,...,x_{i-1},\mathbf{z})} \quad (۱۰-۱۴۷)$$

$$\mathbf{x}=[x_1,...,x_n], i=1,...,n \quad (۱۰-۱۴۸)$$

روش مشابه دیگر برای تقویت لایه کدگشا وابسته کردن بردار خروجی به بردار خروجی مرحله قبل است. این روش در مواردی که نمونه های مختلف دادگان آموزشی به هم وابسته هستند موثر است در حالی که روش قبل ابعاد یک نمونه به هم وابستگی داشتند. در این صورت لایه کدگشا تابع توزیع رابطه ۱۰-۱۴۹ خواهد بود که در آن $\boldsymbol{h}$ مطابق رابطه ۱۰-۱۵۰ بردار خروجی مرحله قبل است. این روش در دادگان برخط[1]، پردازش جریان[2]، که در آن داده ها به صورت سریال، سری زمانی از نمونه های مجموعه دادگان، وارد مدل می شوند می تواند عملکرد مناسبی داشته باشد.

$$P_{(\mathbf{x}|\mathbf{h}_{(k)},\mathbf{h}_{(k-1)},...,\mathbf{h}_{(1)},\mathbf{z})} \quad (۱۰-۱۴۹)$$

$$\mathbf{h}_{(k)}=\hat{\mathbf{x}}_{(k-1)} \quad (۱۰-۱۵۰)$$

همچنین می توانیم در مواردی ساختار هایی را توسعه دهیم که از هر دو روش، رابطه ۱۰-۱۴۷ و ۱۰-۱۴۹، برای ایجاد وابستگی بین دادگان استفاده می کنند. انتخاب استفاده از روش وابستگی بین نمونه ای یا بین ابعاد نمونه به الگوی دادگان آموزشی وابسته بوده و بسته به رفتار داده هر کدام از روش ها می تواند نسبت به روش دیگر ارجح باشد.

۱۰.۳.۱۰.۳ مشکل غیرفعال شدن نورون های پنهان[3] و از بین رفتن وابستگی به توزیع پنهان در ساختار خودرمزگذار متغیر

در توضیحات فوق الذکر روش هایی را پیشنهاد کردیم که با استفاده از آن ها می توانستیم لایه کدگشا را در ساختار تقویت کرده و به نتایج مطلوب تری در ایجاد بردار بازسازی برسیم. اما ارتقا عملکرد لایه کدگشا ممکن است باعث بروز نقص هایی در ساختار بشود؛ بدین صورت که اگر لایه کدگشا ارتقا پیدا کرده، برای مثال از یکی از توزیع های روابط ۱۰-۱۴۷ یا ۱۰-۱۴۹ استفاده کند، عملکرد بسیار خوبی داشته باشد در این صورت می تواند به صورت مستقل از توزیع پنهان و نمونه بردار پنهان، $P_{(x)}$، بردار خروجی را تولید کند، مشابه ساختار *DC-GAN* که در فصل ۱۳ بررسی شده است. در این صورت اگرچه بردارهای خروجی مطلوبی داشته و مقدار جمله اول در رابطه

[1] Online
[2] Stream processing
[3] Dying units problem

۱۰-۱۰۴ تابع هزینه کوچک خواهد بود ولی باید در نظر داشته باشیم که هدف از توسعه مدل های بازنمائی ایجاد مدل هایی است که بتواند داده های اصلی را به داده های مناسب تری (ابعاد کمتر، حذف نویز و...) تبدیل کند که در این ساختار این وظیفه برعهده توزیع پنهان است. استقلال لایه کدگشا از توزیع پنهان باعث می شود آموزش مناسبی برای توزیع پنهان صورت نگرفته و مدل بازنمائی مطلوبی توسعه پیدا نکند. برای درک بهتر موضوع رابطه تابع هزینه ۱۰-۱۳۰ را در نظر می گیریم، این رابطه را می توانیم به صورت رابطه ۱۰-۱۵۱ بازنویسی کنیم.

$$E = KL(P_{(\mathbf{x})} \,||\, P_{\theta(\mathbf{x}|\mathbf{z})})) + KL(P_{(\mathbf{z})} \,||\, Q_{\theta(\mathbf{z}|\mathbf{x})}) \tag{۱۰-۱۵۱}$$

در صورت استقلال لایه کدگشا از توزیع پنهان رابطه ۱۰-۱۵۱ تبدیل به رابطه ۱۰-۱۵۲ می شود که این حالت مطلوب نیست. رابطه ۱۰-۱۵۲ در واقع یک حالت فرضی بوده و به طور صریح در ساختار تعریف نمی شود.

$$E = KL(P_{(\mathbf{x})} \,||\, P_{\theta(\mathbf{x})})) + KL(P_{(\mathbf{z})} \,||\, Q_{\theta(\mathbf{z}|\mathbf{x})}) \tag{۱۰-۱۵۲}$$

در این حالت مقدار جمله اول رابطه ۱۰-۱۵۲ با توجه به ساختار تقویت شده لایه کدگشا و خطای کوچک بردار های خروجی، مقدار بسیار کوچکی خواهد بود. بنابراین در طول روند آموزش برای حداقل کردن تابع هزینه ساختار مقدار جمله دوم را کاهش می دهد. با کاهش جمله دوم توزیع پنهان بر توزیع هدف، نرمال گوسی، منطبق شده و عملا به ازای هر بردار ورودی خروجی بردار رمزگذار یکسان خواهد بود. در این صورت بردار های تصادفی پنهان انتخاب شده از توزیع نیز تقریبا مشابه هم بوده و مدل بازنمائی مناسبی برای دادگان توسعه نخواهد یافت.

برای رفع این مشکل در صورتی که برای تقویت لایه کدگشا از روابط ۱۰-۱۴۷ یا ۱۰-۱۴۹ استفاده کرده باشیم. می توانیم از وابستگی محلی به جای وابستگی سراسری استفاده کنیم. در این صورت رابطه ۱۰-۱۴۷ و ۱۰-۱۴۹ به ترتیب به صورت روابط ۱۰-۱۵۳ و ۱۰-۱۵۴ خواهند بود.

$$P_{(x_i|x_m,...,x_{i-1},\mathbf{z})}, m \neq 1 \tag{۱۰-۱۵۳}$$

$$P_{(\mathbf{x}|\mathbf{h}_{(k)},\mathbf{h}_{(k-1)},...,\mathbf{h}_{(k-m)},\mathbf{z})}, m \neq 1 \tag{۱۰-۱۵۴}$$

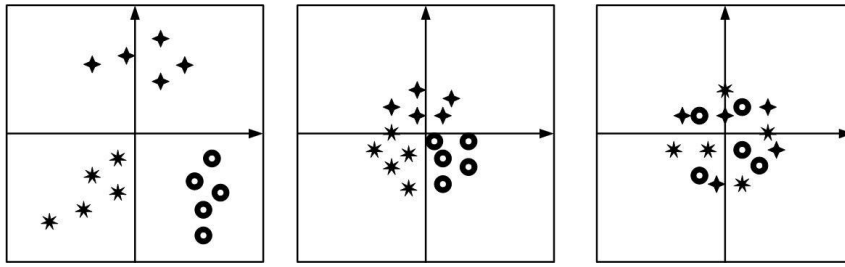
در رابطه ۱۰-۱۵۳ هریک از ابعاد بردار خروجی به جای وابستگی به همه ابعاد پیش از خود تنها به چند بعد قبل از خود وابسته خواهد بود. برای مثال در دادگان تصویری به جای وابستگی همه پیکسل ها به هم، هر پیکسل فقط به پیکسل های اطراف خود وابسته خواهد بود و یا در دادگان سری زمانی هربعد (دینامیک) به جای همه دینامیک های قبل از خود فقط یک یا چند دینامیک قبل از خود وابسته خواهد بود. به طور مشابه برای رابطه ۱۰-۱۵۴ نیز هر نمونه به جای وابستگی به

همه نمونه های پیش از خود فقط به تعدادی نمونه محدود پیش از خود وابسته خواهد بود.

اشاره کردیم که بروز مشکل غیرفعال شدن نورون های لایه پنهان در طول روند آموزش مربوط به انطباق کامل توزیع پنهان، بایاس شـــدن، به توزیع نرمال هدف و کاهش بیش از اندازه مقدار جمله دوم رابطه ۱۵۱-۱۰ تابع هزینه است. بنابراین ممکن است در مواردی که حتی لایه کدگشا تقویت نشـــده باشـــد نیز این مشـــکل به ویژه در گام های اول آموزش که مقدار تابع هزینه کاهش محسوسی دارد بروز پیدا کند. در این صورت راهکار کلی برای رفع این مشکل استفاده از ضرایب میرایی[1] برای جمله هم پوشـانی توزیع پنهان و توزیع هدف در تابع هزینه اسـت. در این روش تابع هزینه رابطه ۱۵۱-۱۰ را به صورت رابطه ۱۵۵-۱۰ توسعه می دهیم که در آن β ضریب اسکالری بین صفر تا یک است که مقدار آن با افزایش گام آموزشی افزایش می یابد. در این رویکرد در گام های اولیه جمله دوم معیار KL در رابطه ۱۵۱-۱۰ تاثیر کمتری مقدار نهایی تابع هزینه داشـــته و متعاقبا با تولید گرادیان های کوچک تر از بایاس شـــدن توزیع پنهان به توزیع هدف جلوگیری می کند.

$$E = KL(P_{(\mathbf{x})} \,||\, P_{\theta(\mathbf{x}|\mathbf{z})})) + \beta KL(P_{(\mathbf{z})} \,||\, Q_{\theta(\mathbf{z}|\mathbf{x})}) \qquad (۱۰-۱۵۵)$$

در شکل ۱۸-۱۰ تاثیر هر یک از دو جمله تابع هزینه در توزیع پنهان ساختار با فرض استفاده از دادگان آموزشی طبقه‌بندی با سه کلاس بررسی شده است، شکل الف تابع هزینه ساختار فقط براساس میانگین مربعات خطای بردار بازسازی، شکل ب ترکیب میانگین مربعات خطا و واگرائی توزیع پنهان و توزیع هدف و شکل ج فقط براساس واگرائی بین توزیع پنهان و توزیع هدف تعریف شده است.



شکل ۱۸-۱۰: بررسی فضای توزیع پنهان با تغییر تابع هزینه

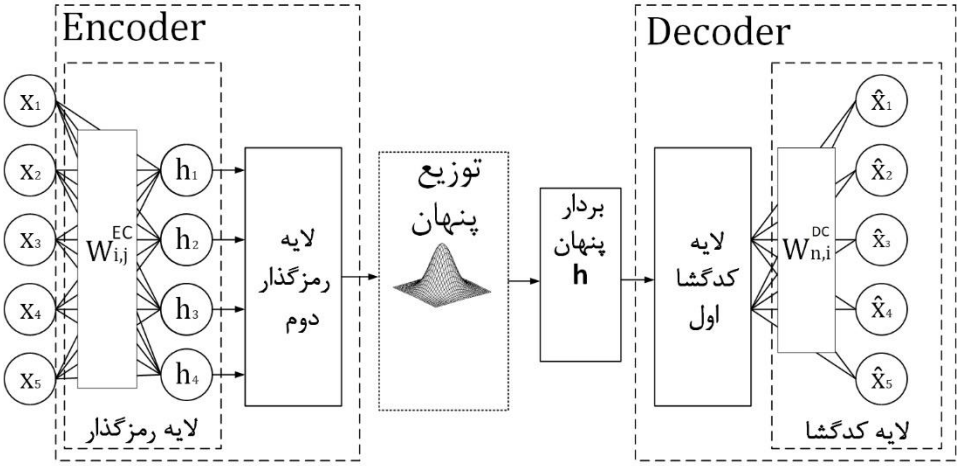
۱۰.۳.۱۰.۴ کاربرد خودرمزگذار متغیر

برای ساختار خودرمزگذار متغیر علاوه بر مدل بازنمائی کاربردهای دیگری نیز تعریف می شود.

[1] Decay factor

در صورتی که این ساختار به عنوان مدل بازنمائی استفاده شود پس از آموزش لایه کدگشا از ساختار حذف شده از توزیع پنهان و نمونه های تصادفی آن برای ایجاد بردارهای بازنمائی مانند سایر ساختارهای خودرمزگذارها استفاده می شود. اما با توجه به ماهیت مولد ساختار که یک توزیع را شبیه سازی می کند می توان پس از آموزش لایه کدگشا را در ساختار باقی گذاشته و با تولید بردارهای پنهان مختلف به ازای بردار ورودی داده هایی مشابه بردار ورودی در خروجی لایه کدگشا تولید کرد. با توجه به یادگیری بدون نظارت در این روش، در مواردی که دادگان آموزشی محدود باشد این روش، روشی موثر برای تولید[1] و افزایش تعداد دادگان خواهد بود.

برای افزایش ظرفیت و بهبود عملکرد ساختار می توان در هر دو کاربرد مدل بازنمائی و مدل مولد از تعدادی لایه سری به جای یک لایه رمزگذار و کدگشا استفاده کنیم، تعداد لایه های رمزگذار و کدگشا لزوما برابر نیستند. شکل ۱۹-۱۰ نمایش یک ساختار خودرمزگذار متغیر با دولایه کدگذار و دولایه کدگشا است.



شکل ۱۹-۱۰: ساختار خودرمزگذار متغیر با چند لایه رمزگذار و کدگشا

۱۰.۳.۱۱ خود رمزگذار واسا اشتاین[2]

در ساختار خودرمزگذارهایی که جمله اصلی تابع هزینه آن ها براساس نرم، فاصله اقلیدسی، بین بردار ورودی و بردار بازسازی تعریف می شود، مشابه همه ساختارهایی که تا کنون بررسی کردیم، در روند آموزش به پارامترهای ساختار، به ویژه لایه کدگشا، تحمیل می شود که براساس یک بردار متفاوت با بردار ورودی، و معمولا با بعدی کمتر از آن، که همان بردار پنهان است برداری عینا مشابه بردار ورودی ایجاد کند. این موضوع می تواند باعث مشکل بیش برازش[3] در ساختار

[1] Data synthesizing
[2] Wasserstein autoencoder (WAE)
[3] Over fitting

شود، علاوه بر بروز این مشکل در مواردی که هدف از توسعه ساختار خودرمزگذار تولید دادگانی با توزیع دادگان اصلی آموزشی است، مانند آنچه در بخش ۴-۱۰-۳-۱۰ معرفی شد، استفاده از تابع هزینه براساس فاصله اقلیدسی بین بردار ورودی و بازسازی باعث می شود تا داده هایی که پس از آموزش ساختار تولید می شوند شباهت بسیار زیادی به داده های اصلی داشته باشند، دادگان اصلی با تغییر اندکی کپی شده باشند، حال آنکه هدف از توسعه چنین ساختاری تولید دادگانی با توزیع مشابه دادگان اصلی ولی متفاوت بود تا نمونه هایی که در مجموعه دادگان وجود دارند از سراسر بازه توزیع احتمالاتی دادگان انتخاب شده و محدوده بیشتری را پوشش دهند تا در ادامه آموزش ساختار هایی که از آن مجموعه داده استفاده می کنند نسبت به مجموعه دادگان اصلی با نمونه های کمتر، بهتر باشد.

با توجه به توضیحات فوق برای رفع معضلات ذکر شده در خودرمزگذارها می توان از تابع هزینه‌ای استفاده کرد که به جای اندازه‌گیری فاصله بین دو نمونه، بردار ورودی و بازسازی، فاصله بین دو توزیع دادگان اصلی $P_{(\boldsymbol{x})}$ و دادگان تولید شده توسط ساختار $P_{(\hat{\boldsymbol{x}})}$ را اندازه گیری کند، در این صورت حالت ایده‌آل برای چنین تابع هزینه‌ای حالتی است که دو توزیع دقیقا با هم، هم‌پوشانی داشته باشند در حالی که نمونه‌هایی که از آن ها انتخاب شده تا میزان هم پوشانی اندازه گیری شود از قسمت های مختلفی از بازه انتخاب شده باشند. برای توسعه چنین تابع هزینه‌ای می توانیم از معیارهای واگرایی که در فصل ۸ تعریف کردیم مانند معیار واگرایی KL یا معیار واگرایی f استفاده کنیم اما با توجه به فرمول این معیارها برای محاسبه میزان هم پوشانی داشتن پارامترهای توزیع ها ضروری است در حالی که در رابطه ۱۲۲-۱۰ بررسی کردیم که محاسبه توزیع $P_{(\boldsymbol{x})}$ دشوار است به همین دلیل از معیاری استفاده می کنیم که هم پوشانی را برای توزیع های غیرپارامتری نیز بتواند محاسبه کند که با توجه به توضیحات فصل ۸ از فاصله واسااشتاین بدین منظور استفاده می کنیم. ساختارخودرمزگذار واسااشتاین ساختاری مشابه ساختار خودرمزگذار متغیر است که در آن به جای استفاده از میانگین مربعات خطا به عنوان تابع هزینه از فاصله واسااشتاین استفاده شده است. تابع هزینه ساختار خودرمزگذار واسااشتاین برای محاسبه هم پوشانی بین توزیع مجموعه دادگان ورودی و تولید شده مطابق رابطه ۱۵۶-۱۰ است.

$$W_c(P_{(\mathbf{x})}, P_{(\hat{\mathbf{x}})}) = \inf_{\Gamma \in p(\mathbf{x}\sim P_{(\mathbf{x})}, \hat{\mathbf{x}}\sim P_{(\hat{\mathbf{x}})})} \mathbb{E}_{(\mathbf{x},\hat{\mathbf{x}})\sim\Gamma}[c(\mathbf{x},\hat{\mathbf{x}})] \tag{۱۰-۱۵۶}$$

در رابطه ۱۵۶-۱۰، $c(,)$ فاصله بین دو بردار است. در صورت استفاده از فاصله اقلیدسی، نرم دوم (W_2)، تابع هزینه به تابع هزینه مرسوم در ساختار شبکه های متخاصم مولد که در فصل ۱۳ بررسی شده است شباهت خواهد داشت. در این ساختار از نرم اول استفاده شده و رابطه ۱۵۶-۱۰ برای ساختار مطابق رابطه ۱۵۷-۱۰ خواهد بود.

$$W_1(P_{(\mathbf{x})}, P_{(\hat{\mathbf{x}})}) = \sup_{f\in F_L} \mathbb{E}_{\mathbf{x}\sim P_{(\mathbf{x})}}[f(\mathbf{x})] - \mathbb{E}_{\hat{\mathbf{x}}\sim P_{(\hat{\mathbf{x}})}}[f(\hat{\mathbf{x}})] \tag{۱۰-۱۵۷}$$

رابطه ۱۰-۱۵۷ فرم دوگانه کانتورویچ-رابینشتاین[1] فاصله واسااشتاین است که در آن F_L مجموعه توابع n-لیپشیتس[2] است. در اینجا از تابع ۱- لیپشیتس استفاده می کنیم که ساده ترین نوع آن یک تابع همانی، $y = x$، است.

توابع n-لیپشیتس به توابعی گفته می شود که در آن ها طبق رابطه نامساوی ۱۰-۱۵۸ شیب تابع محدود شده باشد.

$$| f(x_1) - f(x_2) | \leq n | x_1 - x_2 | \tag{۱۰-۱۵۸}$$

مشکل دیگر خودرمزگذارهای متغیر که در ساختار واسااشتاین برطرف شده است بدین شرح است؛ در خودرمزگذار های متغیر برای هر نمونه ورودی به یک توزیع تقریبی گوسی نسبت داده می شد که پارامترهای توزیع همان بردار خروجی لایه رمزگذار در ساختار بود. در ادامه انحراف این توزیع منسوب به نمونه از توزیع نرمال هدف در تابع هدف لحاظ شده و در طی روند آموزش پارامترهای هر یک از توزیع های تقریبی حاصل از بردار ورودی مشابه شکل ۱۰-۲۰ الف به سمت پارامترهای توزیع نرمال هدف سوق داده می شدند. با فرض ادامه روند آموزش به ازای همه بردار های ورودی پارامترهایی مشابه پارامترهای توزیع نرمال بوده و تفاوت ورودی تاثیر چندانی در پارامترهای توزیع پنهان ایجاد نمی کرد[3]. به همین دلیل بردار تصادفی پنهان حاصل نیز تقریبا برای همه ورودی ها مشابه بوده و از این رو بردار بازسازی حاصل از آن محدوده کوچکی را پوشش می داد. در شکل ۱۰-۲۰ الف تشکیل بردار بازسازی از توزیع پنهان نشان داده شده است.

برای رفع مشکل ذکر شده در خودرمزگذار های متغیر راهکارهایی ارائه شده است که در بخش قبل به چند مورد از آن ها اشاره کردیم. اما راهکارهای ارائه شده صرفا باعث بهبود نسبی شده و اصل مشکل ناشی از نسبت دادن توزیع هدف به هر یک از توزیع های حاصل از نمونه های ورودی همچنان در ساختار مطرح است. ساختار خودرمزگذارهای واسا اشتاین این مشکل مطرح در ساختارهای خودرمزگذار متغیر را بدین صورت برطرف می کند؛ در این ساختار به جای نسبت دادن توزیع هدف به توزیع هر نمونه ورودی که در شکل ۱۰-۲۰ الف نشان داده شده است، توزیع هدف مشابه شکل ۱۰-۲۰ ب برای توزیع کل داده های ورودی نسبت داده می شود. ساختار کلی در این روش، چیدمان لایه ها و...، مشابه خودرمزگذار متغیر است.

چالش مطرح در این ساختار پیشنهادی محاسبه توزیع کلی براساس همه نمونه ها است. که برای محاسبه آن از رابطه ۱۰-۱۵۹ استفاده می کنیم.

$$Q_{(\mathbf{z}|\mathbf{x})} = \int P_{(\mathbf{z}|\mathbf{x})} dP_{(\mathbf{x})} \tag{۱۰-۱۵۹}$$

برای محاسبه انتگرال رابطه ۱۰-۱۵۹ توزیع پنهان هر نمونه ورودی، $P_{(\mathbf{z}|\mathbf{x})}$، به ازای توزیع

[1] Kantorvich-Rubistein's duality
[2] n-Lipshitz functions
[3] Dying units problem

شواهد $P_{(\mathbf{x})}$ در هر مرحله که یک نمونه از دادگان آموزشی وارد ساختار می شود محاسبه و به مقدار قبلی اضافه می شود. بدین ترتیبس روند محاسبه توزیع پنهان کلی $Q_{(\mathbf{z}|\mathbf{x})}$ در طول آموزش ساختار ادامه داشته و به ازای ورود هر نمونه به روز می شود. با این توضیحات می توان رابطه ۱۵۹-۱۰ را به صورت رابطه ۱۶۰-۱۰ نیز نشان دهیم که در آن امید ریاضی شرطی به ازای همه نمونه های وارد شده از ابتدای گام آموزشی محاسبه می شود.

$$Q_{(\mathbf{z}|\mathbf{x})} = E_{P_{(\mathbf{x})}} \left[P_{(\mathbf{Z}|\mathbf{X})} \right] \tag{۱۰-۱۶۰}$$

بنابراین می توانیم میزان واگرایی توزیع $Q_{(\mathbf{z}|\mathbf{x})}$ و $P_{(\mathbf{Z})}$ را نیز همانند معیار KL در ساختار های پیش تر بررسی شده به تابع هزینه اضافه کنیم. در این صورت رابطه ۱۵۶-۱۰ را به صورت رابطه ۱۶۱-۱۰ بازنویسی می کنیم. در این ساختار نیز توزیع هدف $P_{(\mathbf{Z})}$ را همان توزیع گوسی نرمال در ساختار متغیر در نظر می گیریم.

$$W_c(P_{(\mathbf{x})}, P_{(\hat{\mathbf{x}})}) = \inf_{Q:Q_{(\mathbf{z})}=P_{(\mathbf{z})}} \mathrm{E}_{(\mathbf{x},\hat{\mathbf{x}})\sim\Gamma} \mathrm{E}_{Q_{(\mathbf{z}|\mathbf{x})}} [c(\mathbf{x},\hat{\mathbf{x}})] \tag{۱۰-۱۶۱}$$

با توجه به پیچیدگی رابطه ۱۶۱-۱۰ می توانیم آن را به صورت رابطه ۱۶۲-۱۰ که در آن هم پوشانی توزیع پنهان به صورت جمله جداگانه اعمال شده است استفاده کنیم.

$$W_c(P_{(\mathbf{x})}, P_{(\hat{\mathbf{x}})}) = \inf_{Q_{(\mathbf{z}|\mathbf{x})}\in} \mathrm{E}_{(\mathbf{x},\hat{\mathbf{x}})\sim\Gamma} \mathrm{E}_{Q_{(\mathbf{z}|\mathbf{x})}} [c(\mathbf{x},\hat{\mathbf{x}})] + \lambda T \tag{۱۰-۱۶۲}$$

در رابطه ۱۶۲-۱۰، λ یک اسکالر بین صفر تا یک که حساسیت جمله هم پوشانی توزیع پنهان را تعیین می کند و T یک متمایزکننده مشابه رابطه ۱۳۴-۱۰ است که هم پوشانی بین توزیع $Q_{(\mathbf{z}|\mathbf{x})}$ و $P_{(\mathbf{Z})}$ را محاسبه کرده و خود دارای پارامترهای آموزشی است که همراه پارامترهای ساختار به روز می شود. همچنین می توان رابطه ۱۶۲-۱۰ را به صورت رابطه ۱۶۳-۱۰ نیز توسعه دهیم. در این صورت به جای استفاده از متمایزکننده میزان واگرایی توزیع پنهان و هدف را با استفاده از روش حداکثر اختلاف متوسط[1] که در فصل ۸ معرفی کردیم، استفاده کنیم. حداکثر اختلاف متوسط برای دو توزیع $Q_{(\mathbf{z}|\mathbf{x})}$ و $P_{(\mathbf{Z})}$ به صورت رابطه ۱۶۳-۱۰ تعریف می شود.

$$MMD(P_{(\mathbf{z})}, Q_{(\mathbf{z}|\mathbf{x})}) = \left\| \int_{\mathbf{z}} k(\mathbf{z},.) dP_{(\mathbf{z})} - \int_{\mathbf{z}} k(\mathbf{z},.) dQ_{(\mathbf{z})} \right\|_{H_k} \tag{۱۰-۱۶۳}$$

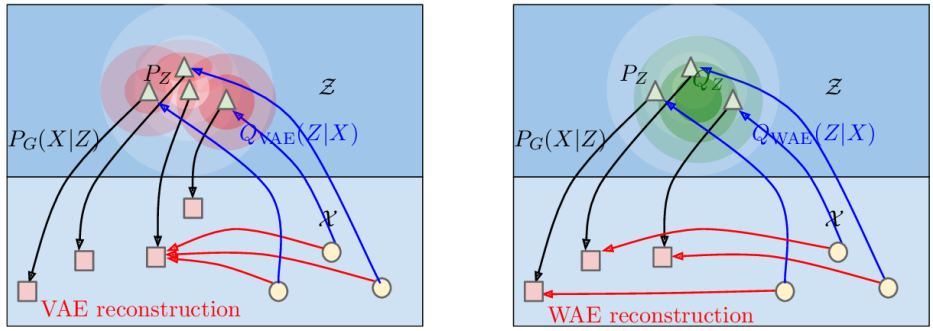
[1] Maximum mean discrepancy (MMD)

رابطه ۱۰-۱۶۳ نمایش تئوریکال بوده و برای استفاده برای نمونه های برداری از توزیع پنهان باید به صورت گسسته، مجموع مقادیر، مطابق رابطه ۱۰-۱۶۴ توسعه داده شود. در رابطه ۱۰-۱۶۳ و ۱۰-۱۶۴ استفاده از کرنل شعاعی[1]، گوسی، می تواند باعث غیر فعال شدن نورون های لایه پنهان شود بنابراین استفاده از کرنل درجه n-ام معکوس مناسب تر است.

$$MMD(P_{(\mathbf{z})}, Q_{(\mathbf{z}|\mathbf{x})}) = \left\| \mathrm{E}_{\mathbf{x}\sim P}[k(\mathbf{z},.)] - \mathrm{E}_{\mathbf{y}\sim Q}[k(\mathbf{z},.)] \right\|_{H_k} \quad (۱۰-۱۶۴)$$

بدین ترتیب رابطه ۱۰-۱۶۲ با استفاده از روش حداکثر اختلاف متوسط به جای متمایزکننده به صورت رابطه ۱۰-۱۶۵ خواهد بود.

$$W_c(P_{(\mathbf{x})}, P_{(\hat{\mathbf{x}})}) = \inf_{Q_{(\mathbf{z}|\mathbf{x})}\in} \mathrm{E}_{(\mathbf{x},\hat{\mathbf{x}})\sim\Gamma} \mathrm{E}_{Q_{(\mathbf{z}|\mathbf{x})}}[c(\mathbf{x},\hat{\mathbf{x}})] + \lambda MMD(P_{(\mathbf{z})}, Q_{(\mathbf{z}|\mathbf{x})}) \quad (۱۰-۱۶۵)$$



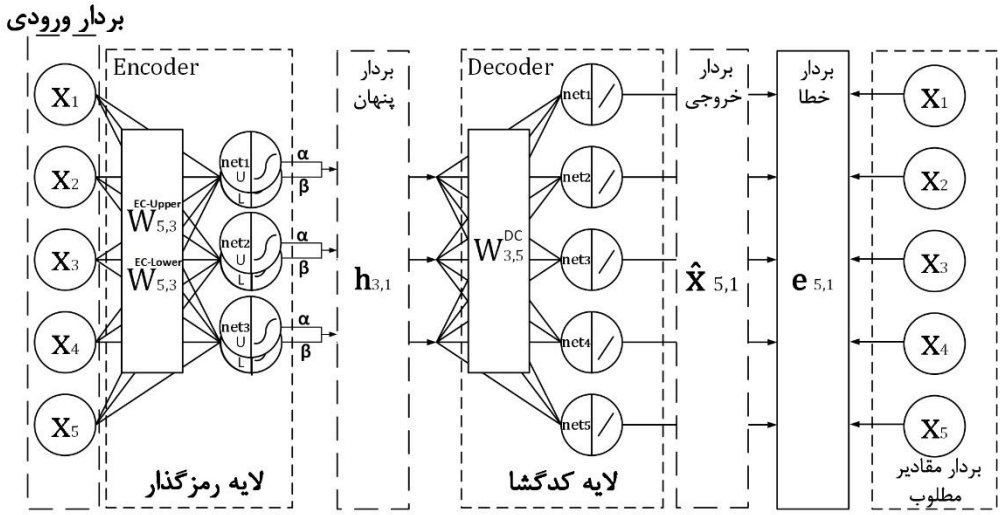
الف) خودرمزگذار متغیر ب) خودرمزگذار واسااشتاین

شکل ۲۰-۱۰: تفاوت توزیع پنهان خودرمزگذار متغیر و واسااشتاین[۸]

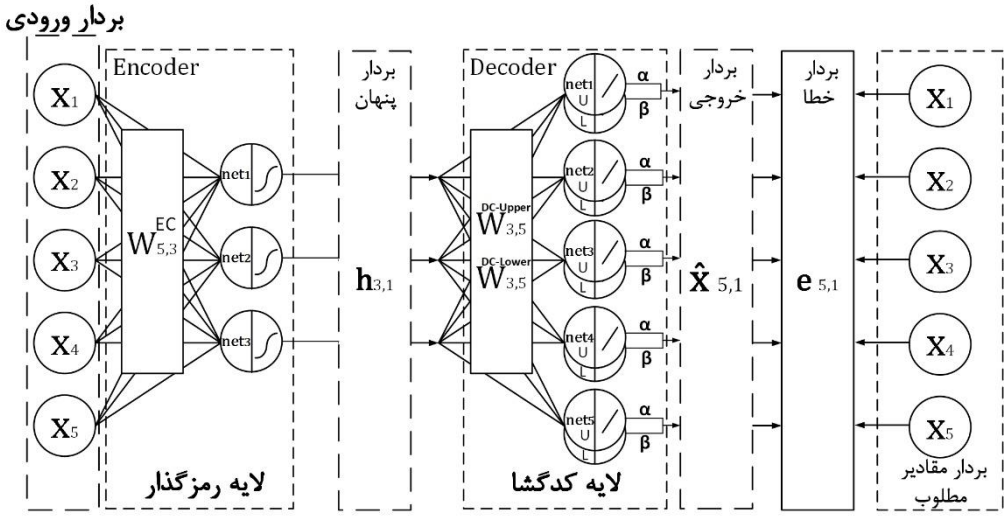
۱۰.۳.۱۲ خود رمزگذارهای راف[2]

از آنجایی که ساختار خودرمزگذار مشابه یک شبکه پرسپترون دو لایه است می توان از نورون های راف استفاده شده در ساختار های شبکه های عصبی راف که در فصل ۷ بررسی کردیم در ساختار خودرمزگذارها نیز بهره گرفت، در این صورت می توان هر دو ساختار راف نوع اول، لایه رمزگذار راف، و دوم، لایه کدگشا راف، را برای خودرمزگذارها پیاده سازی کرد. ساختار حاصل تعریف شده در نسبت به نویز دادگان مقاوم تر خواهد بود. شکل های ۱۰-۲۱ و ۱۰-۲۲ به ترتیب نشان دهنده ساختار راف نوع اول و دوم برای خودرمزگذار هستند که در آن ها امکان استفاده از رابطه ۱۰-۸۲ برای آموزش لایه رمزگذار وجود نداشته و برای هر یک از لایه ها باید روابط مشتق زنجیره ای که در فصل ۷ برای ساختارهای راف بررسی شد توسعه داده شود.

[1] Radial based function (RBF)
[2] Rough autoencoder

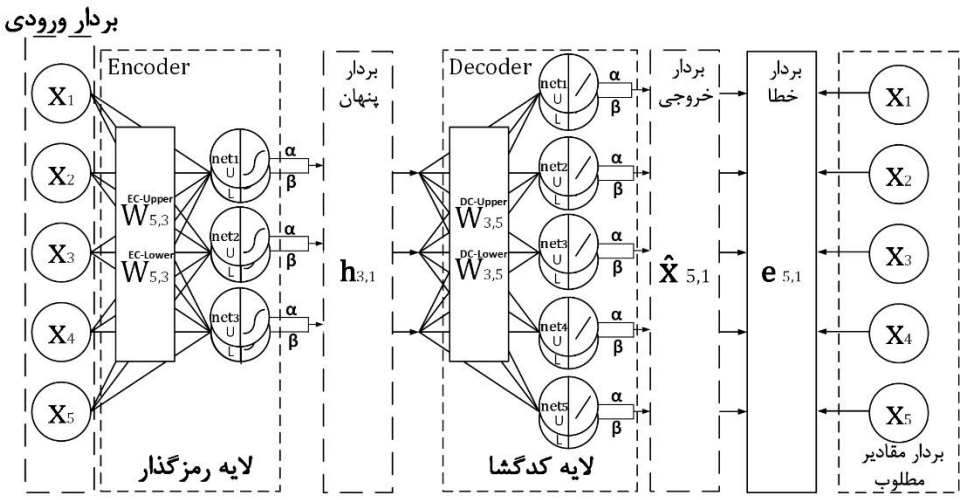


شکل ۲۱-۱۰: خودرمزگذار راف نوع اول



شکل ۲۲-۱۰: خودرمزگذار راف نوع دوم

شکل ۲۳-۱۰ ساختار یک خودرمزگذار راف را نشان می دهد که در آن هر دو لایه رمزگذار و کدگشا راف بوده و با تعریف روابط مشتق زنجیره ای آموزش برای لایه کدگشا از رابطه۸۲-۱۰ برای آموزش لایه رمزگذار استفاده کنیم، در این صورت آموزش وزن های باند بالا و پایین لایه رمزگذار به ترتیب طبق رابطه ۸۲-۱۰ متاثر از باند بالا و پایین لایه کدگشا خواهد بود.



شکل ۲۳-۱۰: خودرمزگذاربا هردو لایه راف

۱۰.۳.۱۳ خود رمزگذارها با آموزش عاطفی[1]

در روند آموزش خودرمزگذارها می توان مقادیر خطا های نمونه های گذشته را نیز در محاسبه تابع هزینه نمونه فعلی وارد کنیم. در این صورت در محاسبه می توانیم با اضافه کردن مشتق خطا که تفاضل خطای گام گذشته و فعلی است به تابع هزینه از آموزش عاطفی استفاده کنیم. در این صورت جمله مربوط به خطای بازسازی در تابع هزینه براساس رابطه ۱۶۶-۱۰ خواهد بود. در رابطه ۱۶۶-۱۰، k_1 k_2 مقادیر اسکالر کنترلی هستند.

$$E = \frac{1}{2}(k_1 e + k_2 \dot{e})^2$$

(۱۰-۱۶۶)

۱۰.۳.۱۴ خود رمزگذارهای بازگشتی[2]

در صورتی که نمونه های دادگان آموزشی که می خواهیم برای آن ها مدل بازنمائی توسعه دهیم وابستگی زمانی به هم داشته باشند استفاده از می توانیم از ساختارهای بازگشتی استفاده کنیم که این وابستگی را در ایجاد نمونه های بازنمائی از دادگان اصلی اعمال کند. استفاده از ساختار بازگشتی در چنین دادگانی می تواند دادگان بازنمائی مطلوب تری نسبت به سایر ساختارها تولید کند. ساختارهای بازگشتی در فصل ۱۱ بررسی شده اند.

برای توسعه با مقایسه ساختارهای بازگشتی بررسی شده در فصل ۱۱ و ساختار یک خودرمزگذار می توان به وجود شباهت هایی بین ساختارهای بازگشتی المن، جردن و المن-جردن

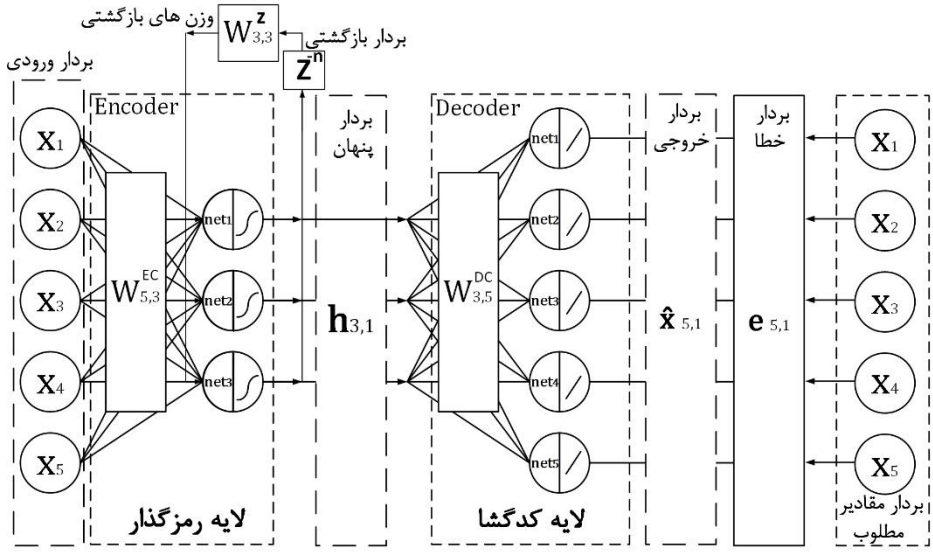
[1] Emotional learning
[2] Recurrent autoencoder

پی برد. همان طور که بررسی کردیم ساختار یک خودرمزگذار متشکل از دو لایه رمزگذار و کدگشا است. به عبارت دیگر خودرمزگذار یک شبکه پرسپترون دو لایه است که مقدار مطلوب برای آن همان ورودی آن است. از طرف دیگر ساختارهای بازگشتی المن، جردن و المن-جردن نیز دارای ساختار دولایه هستند که با افزودن اتصالاتی به یک شبکه پرسپترون دو لایه تعریف می شوند. با این توضیحات می توانیم ساختارهای خودرمزگذارهای بازگشتی را به شرح ذیل تعریف کنیم:

۱۰.۳.۱۴.۱ خود رمزگذار بازگشتی المن[1]

در خودرمزگذار بازگشتی المن خروجی لایه رمزگذار نمونه قبلی با ضرب در وزن های بازگشتی وارد لایه رمزگذار شده و در محاسبه بردار پنهان دخیل می شود.

شکل ۲۴-۱۰ ساختار یک خودرمزگذار بازگشتی المن را نشان می دهد.

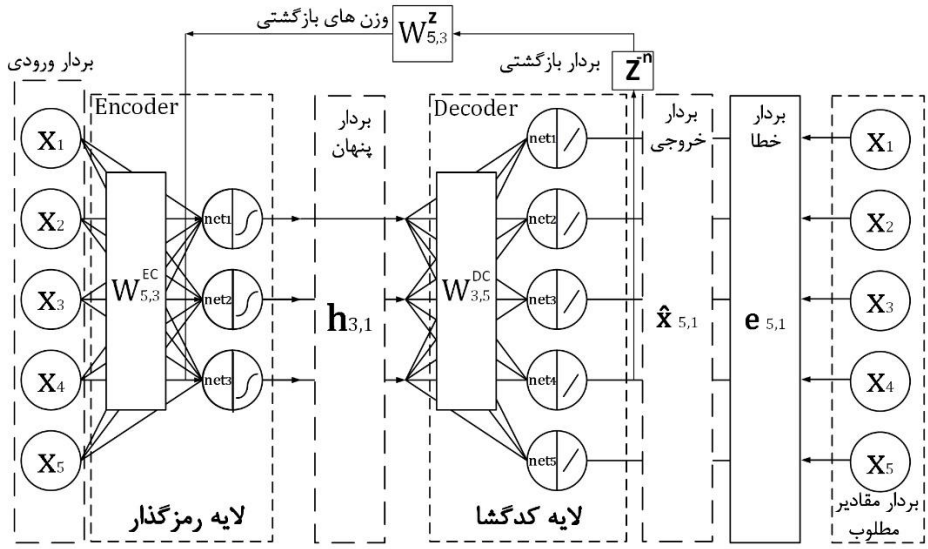


شکل ۲۴-۱۰: ساختار خودرمزگذار المن

۱۰.۳.۱۴.۲ خود رمزگذار بازگشتی جردن[2]

در ساختار خودرمزگذار بازگشتی جردن خروجی لایه کدگشا نمونه قبلی با ضرب در وزن های بازگشتی وارد لایه رمزگذار شده و در محاسبه بردار پنهان دخیل می شود. شکل ۲۵-۱۰ ساختار یک خودرمزگذار بازگشتی جردن را نشان می دهد.

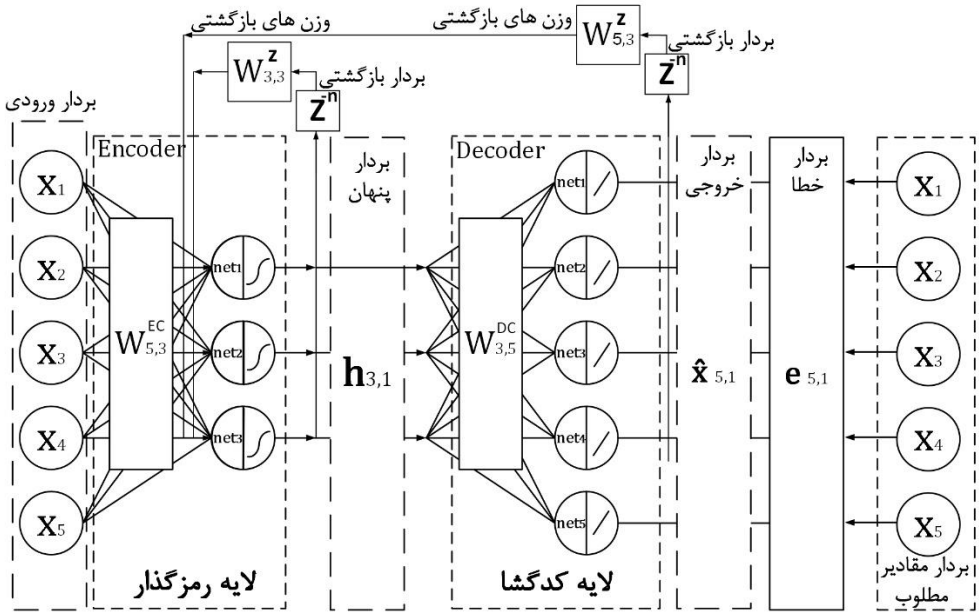
[1] Elman autoencoder
[2] Jordan autoencoder



شکل ۲۵-۱۰: خودرمزگذار جردن

۱۰.۳.۱۴.۳ خود رمزگذار بازگشتی المن-جردن

ساختار خودرمزگذار المن-جردن با ادغام ساختار المن و جردن تعریف می شود و در محاسبه بردار پنهان آن خروجی هر دو لایه رمزگذار و کدگشا برای نمونه قبل دخیل هستند. شکل ۲۶-۱۰ ساختار خودرمزگذار المن-جردن را نشان می دهد.



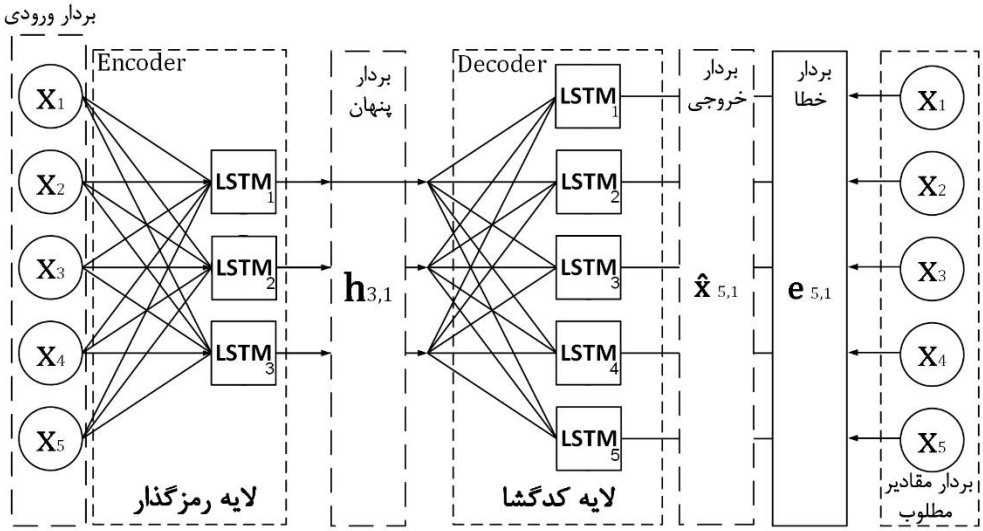
شکل ۲۶-۱۰: خودرمزگذار المن-جردن

در ساختارهای جردن و المن-جردن برای محاسبه بردار پنهان از خروجی لایه کدگشا استفاده می شود در حالی که پیش تر اشاره کردیم که به منطور توسعه یک مدل بازنمائی پس از آموزش لایه کدگشا از ساختار حذف می شود. در این صورت اتصالات بازگشتی نیز با حذف لایه کدگشا حذف شده و در تولید بردارهای پنهان پس از آموزش ساختار نقشی نخواهند داشت. در این صورت با تغییر اتصالات ساختار عملکرد مدل بازنمائی آموزش داده شده کاهش محسوسی خواهد داشت، بنابراین استفاده از این ساختارها در صورتی که بخواهیم از آن به عنوان مدلی برای تولید بردارهای بازنمائی استفاده کنیم مناسب نیست اما در کاربردهایی مانند حذف نویز از دادگان که لایه کدگشا از ساختار حذف نمی شود، مشابه شکل در خودرمزگذارهای متغیر، استفاده از این ساختار مناسب است.

۱۰.۳.۱۴.۴ استفاده از واحدهای بازگشتی در ساختار خودرمزگذار

برای توسعه ساختار بازگشتی در خودرمزگذارها علاوه بر ساختارهای ذکر شده می توان از واحدهای بازگشتی مانند $LSTM$، GRU برای تعریف نورون های ساختار خودرمزگذار استفاده کنیم. در این صورت وزن های ساختار مطابق واحدهای بازگشتی مورد استفاده تعریف می شود و در صورتی که از ساختار هر دو لایه رمزگذار و کدگشا در این ساختار یکسان باشد می توان برای هر وزن متناظر در لایه رمزگذار از رابطه ۸۲-۱۰ استفاده کرده و روابط آموزشی را فقط برای یک لایه توسعه دهیم. شکل ۲۷-۱۰ ساختار یک خودرمزگذار با واحدهای $LSTM$ در واحد رمزگذار را نشان

می دهد. در ساختار شکل ۲۷-۱۰ با توجه به عدم تطابق وزن های دو لایه و یکسان نبودن وزن ها نمی توان از رابطه ۸۲-۱۰ اســتفاده کرد و باید روابط آموزشــی را برای وزن های هر بخش جداگانه تعریف کنیم. روابط و ساختارهای بازگشتی در فصل ۱۱ بررسی شده اند.



شکل ۲۷-۱۰: خودرمزگذار با واحدهای LSTM

۱۰.۴ مسائل

۱. مقادیر قطر اصلی ماتریس وزن های رابطه ۲-۱۰ را محاسبه کنید.

۲. برای آموزش ماشین های بولتزمن روش گرادیان نزولی تصادفی را پیاده سازی کرده و با روش گرادیان نزولی مقایسه کنید.

۳. برای یک مجموعه دادگان آموزشی، یک شبکه باور عمیق و یک خودرمزگذار پشته ای توسعه داده و بردارهای پنهان حاصل از هر ساختار را برای آموزش یک شبکه پرسپترون دو لایه استفاده کرده و نتایج را با هم مقایسه کنید.

۴. ساختار خودرمرگذار متغیر را با تولید تعداد بیشتر نمونه تصادفی از توزیع پنهان، به جای یک بردار، آموزش دهید و نتایج را با حالتی که از یک بردار استفاده می شود مقایسه کنید.

۵. با توجه به اینکه در خودرمزگذارها پس از آموزش محلی لایه کدگشا حذف می شود، در مورد نحوه تولید بردار بازگشتی در خودرمزگذار جردن پس از آموزش محلی و حذف لایه کدگشا بحث کنید.

فصل ۱۱
سری های زمانی و شبکه های عصبی بازگشتی[1]

۱۱.۱ مقدمه

با توجه به رشـد رویکردها و روش های کاربردی در پردازش دادگان سـری های زمانی[2] و ارائه روش های نوین برای پیش بینی[3] وضعیت متغیرها در افق های آینده براساس الگوی رفتاری آن ها در گذشـته، مانند وضـعیت آب و هوا، میزان آلودگی، رفتار بورس و...، تمرکز بر توسـعه مدل های مبتنی بر یادگیری ماشین و یادگیری عمیق نیز با توجه به پتانسیل این روش ها در ارائه مدل های پیش بین در این حوزه گسترش یافته است. براین اساس مدل ها و ساختارهای مبتنی بر یادگیری عمیق در کنار سـایر رویکردها با هدف ارائه مدل های پیش بینی کننده توسـعه یافته‌اند. در اغلب موارد دادگان مربوط به یک سری زمانی حجم بالائی داشته و در مقایسه با سایر مجموعه دادگان آموزشـی از تعداد نمونه های بیشـتری تشـکیل شـده اند، به همین علت در توسـعه مدل ها و سـاختارهای مربوط که برای چنین کاربردهایی توسـعه داده شـده‌اند این مورد همواره مورد توجه محققین این حوزه بوده اسـت. بدین ترتیب سـاختارهای ارائه شـده قادر هسـتند با پردازش حجم بالائی از دادگان مربوط به الگوی رفتاری گذشـته یک متغیر پیش بینی مناسـبی را از کمیت آن در آینده ارائه کنند. برای مثال یک مدل بیش بینی کننده می تولند وضـعیت یک متغیر مانند آب و هوا، میزان آلودگی هوا، وضعیت بورس و... در روز های آینده براساس مقادیر فعلی و گذشته متغیر (آب و هوای فعلی و روزهای گذشته) پیش بینی کند.

در این فصـل ابتدا به بررسـی سـاختار دادگان آموزشـی که دارای وابسـتگی زمانی بوده و برای آموزش چنین مدل هایی مورد اسـتفاده قرار می گیرند پرداخته و سـپس سـاختارهایی را که برای پردازش چنین دادگانی توسـعه داده شـده اند را معرفی می کنیم. در ادامه رویکرد هایی را بررسـی می کنیم که با اعمال آن ها در ساختار های مرسومی که تا کنون بررسی کردیم، شبکه های عصبی پرسـپترون، خودرمزگذار و...، به سـاختارها قابلیت پردازش دادگان سـری زمانی افزوده شـده و بدین ترتیب امکان توسعه مدل های پیش بینی کننده عمومی[4] برای سری های زمانی فراهم می شود. در

---

[1] Recurrent neural networks (RNNs)
[2] Time series
[3] Prediction
[4] General

این صورت می توان از روش هایی که برای بهبود عملکرد ساختارهای مرسوم ارائه شده است برای مدل های مربوط به سری های زمانی نیز استفاده کرده و عملکرد مدل را ارتقا داد. در هر قسمت پس از معرفی ساختار روش آموزش و روابط آن که عمدتا براساس گرادیان نزولی تعریف شده است را نیز بررسی کرده و معایب و مزایای هر ساختار و روش را نیز بررسی می کنیم. در نهایت در انتهای فصل مسائلی برای درک بهتر موضوع فصل و ارتباط آن با سایر قسمت ها در نظر گرفته شده است.

۱۱.۲ ساختار دادگان با وابستگی زمانی

در فصل۶ در مورد مدل های خود هم بسته[1] بحث کردیم، در این مدل ها برای یک سری زمانی مدلی تعریف می کردیم که در آن مقدار متغیر در هر گام زمانی براساس مقادیر آن در گام های گذشته مطابق رابطه۲۰-۶ فصل ۶ محاسبه می شد. اگر فرض کنیم خروجی رابطه۲۰-۶ مقدار متغیر سری زمانی در گام آینده،$k+1$، باشد در این صورت رابطه۲۰-۶ یک پیش بینی کننده براساس مقادیر گام فعلی، k، و گام های گذشته، $k-i, i=1,\dots,k-1$، برای گام آینده متغیر است. گام آینده که خروجی مدل است لزوما یک گام بعد از گام فعلی نبوده و می تواند چند گام بعدتر، $k+n, n>1$، از گام فعلی باشد، البته محاسبه مقدار خروجی در این حالت معمولا دشوارتر از حالتی است که هدف پیش بینی یک گام جلوتر است. حال اگر الگوی رفتاری این سری زمانی درجه غیرخطی بالائی داشته باشد به طوری که استفاده از رابطه۲۰-۶ همواره خطای بزرگی به همراه دارد. در این صورت می توانیم از یک مدل پیچیده‌تر، مانند یک شبکه عصبی پرسپترون، به جای رابطه۲۰-۶ به عنوان مدل پیش بینی کننده استفاده کنیم تا عملکرد مدل بهبود پیدا کند. بنابر این توضیحات در این صورت مقادیر متغیر سری زمانی برای گام فعلی و گام های گذشته در کنار هم به صورت برداری مطابق رابطه ۱-۱۱ به عنوان ورودی شبکه و مقدار گام آینده مطابق رابطه ۲-۱۱ به عنوان خروجی شبکه عصبی پرسپترون که هدف از آموزش شبکه محاسبه آن است، مقدار مطلوب، در نظر می گیریم. در روابط ۱-۱۱ و ۲-۱۱، m,n مقادیر اسکالر مثبت برای تعیین گام های زمانی به ترتیب برای متغیرهای ورودی و خروجی هستند.

$$\mathbf{x}=\begin{bmatrix} x_{k-m-1} \\ x_{k-m} \\ \vdots \\ x_k \end{bmatrix} \qquad (۱۱-۱)$$

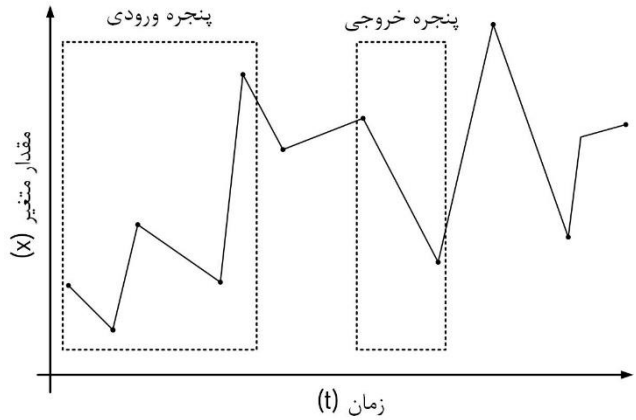
$$y_{k+n} \qquad (۱۱-۲)$$

[1] Autoregressive models

با توجه به ظرفیت مدل های پیچیده تر مانند شبکه های عصبی پرسپترون برخلاف مدل های خود هم بسته که در آن ها خروجی مدل معمولا برای یک بعد، اسکالر، تعریف شده و برای خروجی های چند بعدی، برداری، معمولا عملکرد مطلوبی از مدل خود هم بسته حاصل نمی شد، در مدل های پیچیده مانند شبکه های عصبی پرسپترون می توان خروجی شبکه را به صورت بردار تعریف کرده و مقدار متغیر را در بازه ای مطابق رابطه ۳-۱۱ از گام های آینده تعریف کنیم. در رابطه ۳-۱۱، n,l مقادیر اسکالر طبیعی برای تعیین گام های زمانی هستند.

$$\mathbf{y}=\begin{bmatrix} y_{k+n+1} \\ y_{k+n+2} \\ \vdots \\ y_{k+n+l} \end{bmatrix}, n>0, l>0 \qquad (۱۱-۳)$$

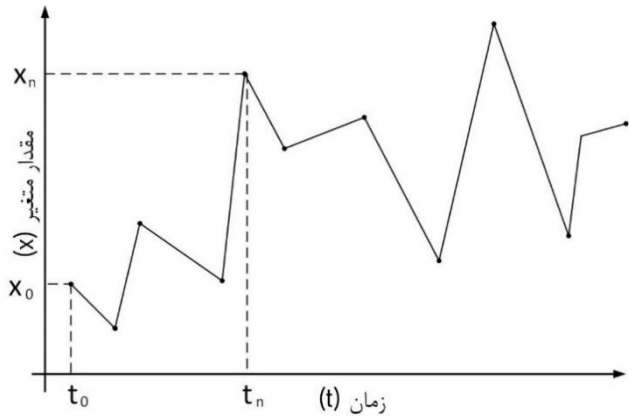
شکل ۱-۱۱ یک سری زمانی برای یک متغیر را نشان می دهد که در آن بردار ورودی رابطه ۱-۱۱ و بردار خروجی رابطه ۳-۱۱ به صورت پنجره هایی نشان داده شده است. بدیهی است که اندازه پنجره ورودی و پنجره خروجی به ترتیب برابر اندازه بردار ورودی و خروجی مدل طبق روابط ۱-۱۱ و ۳-۱۱ است. علاوه بر این می توان از مدل های تعریف شده برای سری زمانی برای طبقه بندی نیز استفاده کنیم در این صورت به ازای هر پنجره ورودی مطابق شکل ۱-۱۱، یک بردار مقدار مطلوب، برچسب، برای طبقه بندی نسبت داده می شود؛ این بردار در صورتی که تعداد کلاس های مسئله بیش از دو کلاس باشد به صورت بردار OneHot و در صورتی که تعداد کلاس ها دو یا کمتر باشد به صورت یک بردار یک بعدی، اسکالر، تعریف می شود.



شکل ۱-۱۱: پنجره ورودی با بعد پنج و خروجی با بعد دو تعریف شده بر دادگان یک سری زمانی

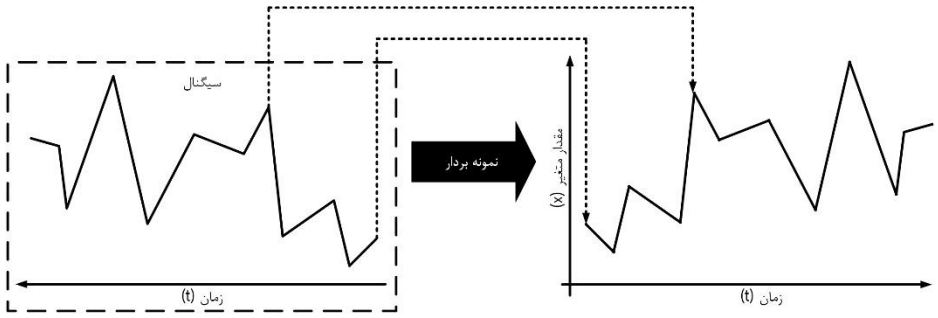
۱۱.۲.۱ ایجاد مجموعه دادگان آموزشی با استفاده از نمونه های یک سری زمانی

نمونه های یک سری زمانی برای یک پارامتر متغیر با زمان، با بسامد معین، فرکانس نمونه برداری، مطابق شکل ۲-۱۱ از یک مبدا زمانی t_0 ثبت می شوند. هر گام زمانی پارامتر متغیر با زمان یک مقدار، x_n، عددی اختیار می کند که توالی آن ها یک سری زمانی مطابق شکل ۲-۱۱ تشکیل می دهد.



شکل ۲-۱۱: نمونه های یک سری زمانی

اگر نمونه های سری زمانی را از دیدگاه پردازش سیگنال بررسی کنیم، پارامتر سری زمانی مانند یک سیگنال است که مقادیر آن به صور گسسته با بسامدی مشخص توسط یک نمونه بردار مطابق شکل ۳-۱۱ ثبت شده و در نهایت مجموعه متشکل از توالی این مقادیر دادگان سری زمانی را تشکیل می دهند.

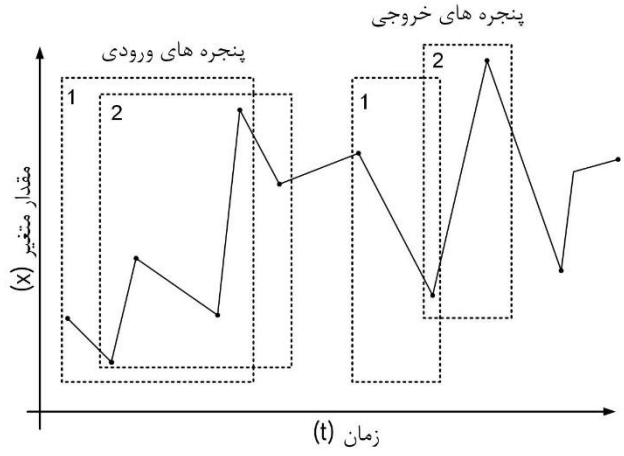


شکل ۳-۱۱: نمونه برداری از سیگنال

برای اینکه از این مجموعه داده مشابه شکل برای آموزش، آموزش با سرپرست، یک مدل مانند شبکه عصبی استفاده کنیم باید، آن را به مجموعه داده ای مشابه آنچه در فصل برای آموزش شبکه

های عصبی پرسپترون چند لایه استفاده کردیم تبدیل کنیم، بدین منظور ابتدا تعداد گام هایی را که براساس آن ها می خواهیم شبکه آموزش داده شده مقدار گام(ها) آینده را پیش بینی کند، بعد ورودی، به صورت شکل ۴-۱۱ با استفاده از پنجره گذاری نمونه ها تعیین می کنیم. نمونه های داخل این پنجره همان بردار ورودی یک نمونه از مجموعه دادگان آموزشی را تشکیل می دهند. از آنجایی که در آموزش با سرپرست هر نمونه ورودی دارای یک مقدار مطلوب است، برای تعیین بردار مقدار مطلوب پنجره دیگری بر روی دادگان سری زمانی تعریف می کنیم، مقدار مطلوب بازه ای از گام های جلوتر از پنجره ورودی است که هدف از آموزش مدل پیش بینی مقادیر نمونه های آن بازه بر اساس مقادیر پنجره ورودی است. مطابق شکل ۴-۱۱ برای سری زمانی مذکور یک پنجره خروجی با اندازه دو تعریف می کنیم که مقادیر نمونه یک و دو گام بعد از بردار ورودی را تعیین کند.

برای تعریف نمونه های دیگر مجموعه دادگان پنجره های بردار ورودی و مطلوب را با گام حرکتی[1] یک مطابق شکل حرکت می دهیم. بدین ترتیب نمونه دوم مجموعه دادگان آموزشی مطابق شکل ۴-۱۱ حاصل می شود. همان طور که در شکل ۴-۱۱ نیز مشخص است اندازه پنجره ها در طول روند حرکت بر روی نمونه ها ثابت است.



شکل ۴-۱۱: حرکت پنجره ها بر روی نمونه های یک سری زماتی

برای حرکت پنجره ها می توانیم در مواردی که در ادامه به آن ها اشاره خواهیم کرد از گام حرکتی بزرگتر از یک استفاده کنیم، اما بیشترین تعداد نمونه برای مجموعه دادگان آموزشی تنها در صورتی که مقدار گام حرکتی پنجره ها برابر یک باشد حاصل می شود. بدیهی است که مقدار گام حرکتی برای هر دو پنجره ورودی و مقدار مطلوب باید برابر باشد. بر این اساس مجموعه دادگان

[1] Stride

آموزشـــی به صـــورت بردارهایی مطابق رابطه ۴-۱۱ می شـــود. در رابطه ۴-۱۱، m,n,l,k مقادیر اسکالر هستند.

$$\mathbf{x}_1=[x_1,x_2,...,x_n],\mathbf{y}_1=[y_{n+m},y_{n+m+1},...,y_{n+m+l}]$$
$$\mathbf{x}_2=[x_2,x_3,...,x_{n+1}],\mathbf{y}_2=[y_{n+m+1},y_{n+m+2},...,y_{n+m+l+1}]$$
$$\vdots$$
$$\mathbf{x}_k=[x_k,x_{k+1},...,x_{n+k-1}],\mathbf{y}_k=[y_{n+m+k-1},y_{n+m+k},...,y_{n+m+l++k-1}] \quad (۱۱-۴)$$

به این روش تولید دادگان روش n-gram اطلاق می شـــود منظور از n همان لندازه بردار، بردار ورودی$\boldsymbol{x}$، در مجموعه دادگان اســـت. به عنوان مثال اگر در رابطه ۴-۱۱ لندازه هریک از بردارهای ورودی سه باشد مجموعه دادگان آموزشی به روش 3-gram ایجاد شده اند.

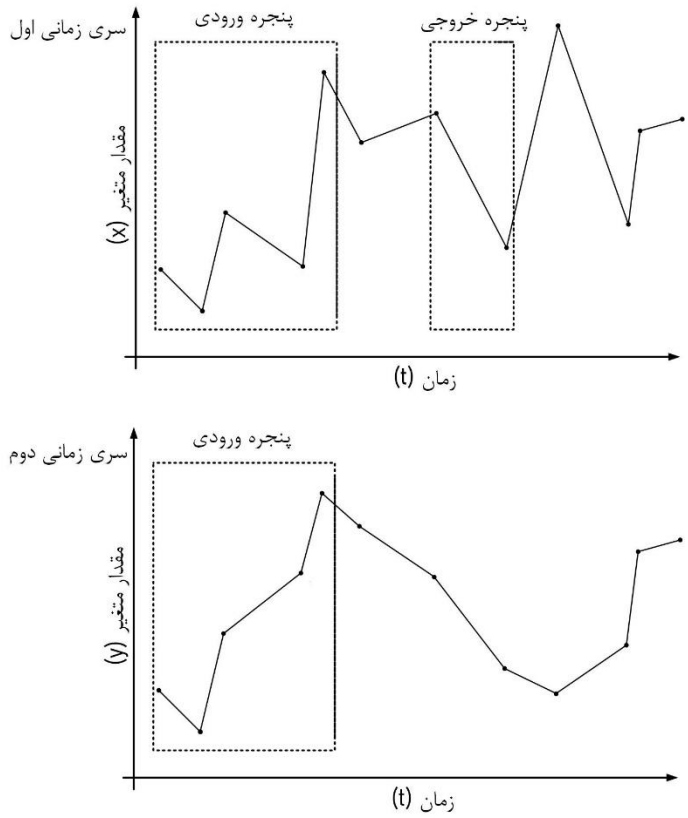
۱۱.۲.۲ استفاده از چند سری زمانی در مدل‌های خود هم بسته

با توجه به آنچه در مورد مدل های هم بسـته مشـابه رابطه ۲۰-۶ در فصـل ۶ بررسـی کردیم. در مواردی برای پیش بینی مقادیر متغیر یک سری زمانی در گام های آینده می توانیم علاوه بر مقادیر آن متغیر در گام های گذشته از مقادیر متغیرهای سری های زمانی دیگری نیز در کنار سری زمانی اصلی استفاده کنیم. در صورتی که تعداد نمونه های این سری های زمانی با هم برابر بوده و دارای بسـامد نمونه برداری[1] برابری داشـته باشـند می توانیم علاوه بر پنجره ورودی بر روی سـری زمانی اولیه بر روی سری های زمانی دیگر نیز مطابق شکل ۵-۱۱ پنجره ورودی تعریف کرده و از آن برای پیش بینی مقادیر گام آینده سری زمانی اولیه استفاده کنیم. در این صورت مدل توسعه داده شده علاوه بر مقادیر گام های گذشـته سـری زمانی اولیه، از مقادیر گام های گذشـته دیگر سـری ها نیز برای محاسـبه خروجی گام آینده سری اولیه به طوری که در شکل نشان داده شده است، استفاده می کند.

در حالت ذکر شـــده که در آن مدل از چند ســـری زمانی اســـتفاده می کند، برای تعریف مدل بسامد سری های زمانی حتما باید مشخص باشد، همچنین ضروری است که زمان ثبت اولین نمونه در هر سـری نیز ثبت شـده و سـری زمانی حداقل برای اولین نمونه دارای برچسـب زمانی[2] باشـد. با داشتن اطلاعات مربوط به بسامد و برچسب زمانی ثبت اولین نمونه برای هر یک از سری های زمانی ممکن است یکی از حالت های ذیل برای سری های زمانی صدق کند، که شـرایط هرکدام از آن ها را بررسـی می کنیم. بررسـی حالت های مختلف برای سـری های زمانی به ترتیب زیر را به منظور سادگی با فرض دو سری زمانی انجام می دهیم که قابل تعمیم به تعداد بیشتر از سری های زمانی است:

[1] Sampling frequency
[2] Timestamp

۱. **زمان ثبت اولین نمونه و بسامد برای همه سری های زمانی یکسان باشد:** این حالت برای تعریف مدل حالت ایده آل است. در این حالت پنجره گذاری را با اندازه پنجره های یکسان برای همه سری ها از اولین نمونه هر سری مطابق شکل ۵-۱۱ انجام می دهیم. همان طور که در شکل ۵-۱۱ نیز مشخص شده است پنجره خروجی نیز که همان مقدار مطلوب مدل، آموزش با سرپرست، است روی سری که هدف محاسبه خروجی آن است در کنار پنجره ورودی تعریف می شود. فاصله پنجره ورودی و خروجی ممکن است صفر یا مقداری بزرگتر از صفر باشد. در صورتی که این فاصله بزرگ باشد در این صورت مدل برای پیش بینی افق[1] های دورتری آموزش داده می شود.

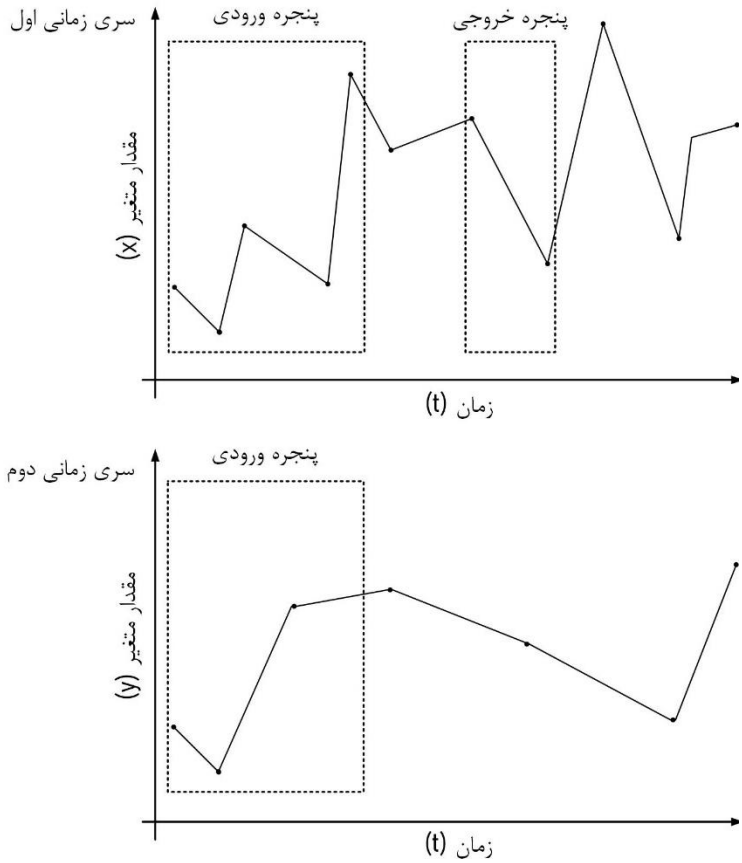


شکل ۵-۱۱: دو سری زمانی با زمان ثبت اولین نمونه و بسامد یکسان

۲. **زمان ثبت اولین نمونه یکسان بوده ولی بسامد برای هر سری ها متفاوت باشد:** در این

[1] Horizon

صـورت اگر اولین زمان ثبت نمونه را به عنوان مرجع در نظر گرفته و نمونه های دو سـری زمانی را مطابق شـکل ۶-۱۱ در کنار هم قرار دهیم، مشـاهده خواهیم کرد که سـری که دارای بسـامد کمتری اسـت تعداد نمونه های کمتری دارد و در صـورت پنجره گذاری با پنجره هایی با اندازه یکسـان، بازه زمانی، اندازه بردار ورودی برای سـری زمانی با بسـامد کمتر کوچکتر خواهد بود. شکل ۶-۱۱ نمایش مقادیر هر سری در این وضعیت است که در آن مشاهده می کنیم که با اعمال پنجره های با اندازه، بازه زمانی، یکسان تعداد نمونه های متفاوتی در پنجره هر سری قرار داده شده است.



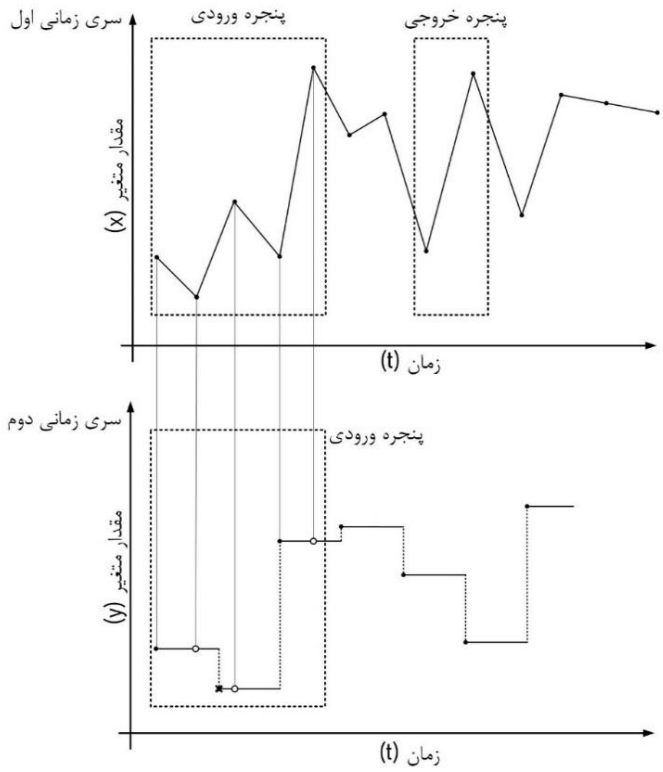
شکل ۶-۱۱: دو سری زمانی با بسامد نمونه برداری متفاوت

در صورتی که بخواهیم تعداد نمونه های سری با بسامد کمتر با تعداد نمونه های سری با بسامد بیشتر برابر شود می توانیم از درون یابی[1] درجه n-ام برای تولید نمونه های بیشتر بین هر دو نمونه از سـری زمانی اسـتفاده کنیم. برای درک بهتر مطالب روش های درون یابی مثال هایی از آن را به

[1] Interpolation

صورت زیر در نظر می گیریم:

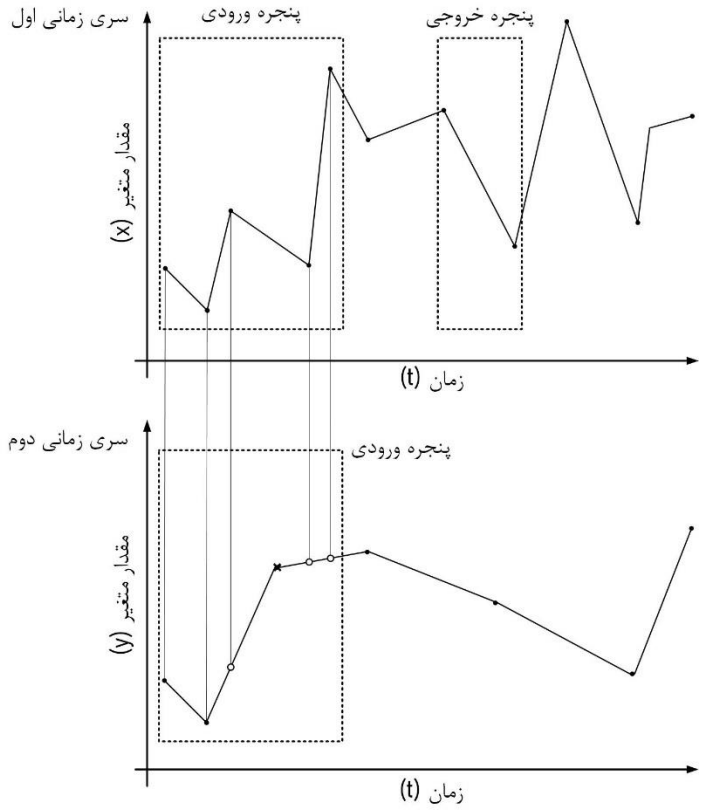
- **درون یابی درجه صـــفر:** در این روش مقدار مربوط به نمونه اول در هر بازه به عنوان مقادیر بین دو بازه قرار گرفته و با توجه به بســـامد مدنظر تعدادی نمونه با همان مقدار نمونه اول تولید می شـود. شـکل ۷-۱۱ این روش درون یابی اسـت. همان طور که در شـکل ۷-۱۱ نیز مشـخص شـده اسـت تعداد نمونه ها و زمان های ثبت نمونه های سری زمانی با بسامد بیشتر به عنوان مرجع در نظر گرفته شده و مقادیر متناظر با این زمان های مرجع برای سـری زمانی دیگر مشـخص می شـوند. بسـته به زمان های مرجع بعضی از این نمونه ها برای سری زمانی با بسامد کمتر ممکن است نمونه های ثبت شده اصلی و بعضی دیگر به روش درون یابی ایجاد شده باشند، سایر نمونه های ثبت شـده سـری زمانی با بسـامد کمتر که زمان ثبت آن ها با زمان های مرجع سـری اول مطابقت ندارند حذف شـده و در تشـکیل بردار پنجره ورودی دخیل نمی شوند. در شکل ۷-۱۱ نمونه های مشخص شده با نماد O نمونه های حاصل از درون یابی، نمونه هایی که با نماد ● مشـخص شـده اند نمونه های اصـلی سـری زمانی و نمونه هایی که با نماد × مشخص شده اند نمونه های اصلی سری زمانی دوم هستند که با زمان های مرجع از سری اول تطابق نداشته و حذف شده اند.



شکل ۷-۱۱: درون یابی درجه صفر

- **درون یابی درجه اول (خطی):** در این روش یک خط بین نمونه اول و دوم در هر بازه تعریف می شود و نمونه های تولید شده مقادیری را که از معادله خط نسبت به موقعیت نمونه، فاصله از نمونه اول بازه و انتهای بازه، مطابق رابطه حاصل می شود ۵ -۱۱ دریافت می کنند، به عبارت دیگر مقادیر نمونه های تولید شده همان نقاط روی خط تعریف شده هستند که در رسم نمودار سری زمانی استفاده می شود. شکل ۸-۱۱ نمایش نموداری این روش درون یابی است. در رابطه ۵-۱۱، x_a، x_b و x_n به ترتیب نمونه اول بازه، انتهای بازه و نمونه ای که مقدار آن را می خواهیم به روش درون یابی حساب کنیم بوده و t_a، t_b و t_n زمان های ثبت هر یک از این نمونه ها است. در شکل ۸-۱۱ نمونه های مشخص شده با نماد O نمونه های حاصل از درون یابی، نمونه هایی که با نماد ● مشخص شده اند نمونه های اصلی سری زمانی و نمونه هایی که با نماد × مشخص شده اند نمونه های اصلی سری زمانی دوم هستند که با زمان های مرجع از سری اول تقاطبق نداشته و حذف شده اند.

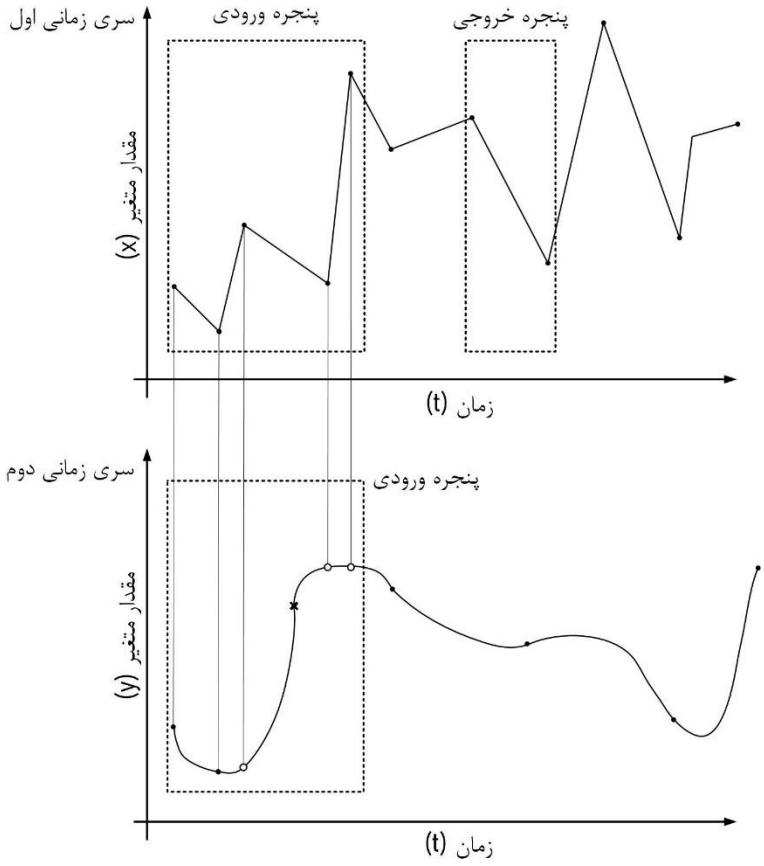
$$x_n = (\frac{x_b - x_a}{t_b - t_a})t_n + x_a \qquad (۱۱-۵)$$



شکل ۸-۱۱: درون یابی درجه اول

- **درون یابی درجه دوم و بالاتر:** این روش نیز مشـابه روش قبلی بوده با این تفاوت که در این روش به جای اسـتفاده از خط بین دو نمونه از یک منحنی درجه دوم یا بالاتر مطابق شکل ۹-۱۱ استفاده می شود. انتخاب درجه مناسب روش درون یابی وابسته به الگوی رفتاری داده ها بوده و بلید درجه درون یابی به گونه ای انتخاب شـود که بیشترین برازش را با منحنی دادگان اصلی داشته باشد البته استفاده از درون یابی با درجات بالاتر نسبت به درون یابی خطی مستلزم هزینه محاسباتی بیشتری بوده و به همین علت در مواردی با وجود برازش بهتر منحنی از روش خطی یا حتی درجه صفرم استفاده می شود. در شکل ۹-۱۱ نمونه های مشخص شده با نماد O نمونه های حاصـل از درون یابی، نمونه هایی که با نماد ● مشـخص شـده اند نمونه های

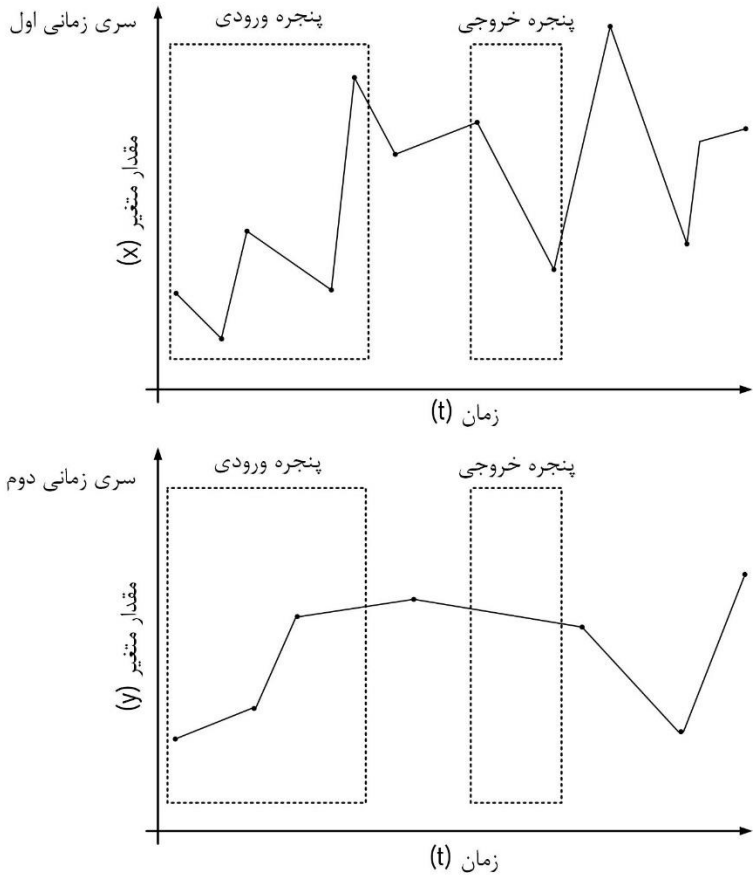
اصـلی سـری زمانی و نمونه هایی که با نماد × مشـخص شـده اند نمونه های اصـلی سری زمانی دوم هستند که با زمان های مرجع از سری اول تقاطبق نداشته و حذف شده اند.



شکل ۹-۱۱: درون یابی درجات بالاتر

حالتی را فرض کنید که در آن پنجره خروجی نیز مطابق شـــکل ۱۰-۱۱ برای دو خروجی، یا فقط خروجی سـری با بسـامد کمتر، تعریف شـده و بسـامد دو سـری برابر نباشـد. در این صـورت با حرکت پنجره ها در طول زمان ممکن اســـت پنجره خروجی برای یک ســـری فاقد نمونه و متعاقبا فاقد مقدار مطلوب برای آموزش با سـرپرسـت مدل باشـد. در این صـورت امکان آموزش مدل وجود ندارد. برای رفع این معضل دو راهکار وجود دارد. نخست آنکه گام حرکتی پنجره به گونه ای نسـبت به بسامد سری زمانی با بسامد کمتر تنظیم شود که همواره پنجره خروجی برای هر دو سری دارای نمونه باشـد، اسـتفاده از گام حرکتی بزرگتر از یک، مشـکل به وجود آمده در صـورت اسـتفاده از این

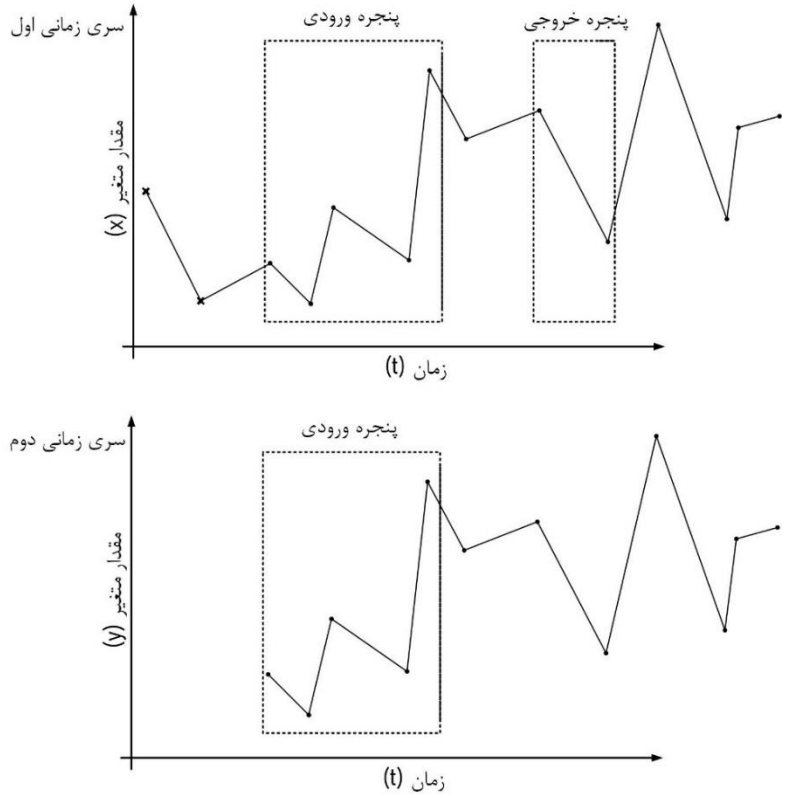
راهکار این است که تعداد زیادی از نمونه ها به دلیل استفاده از گام حرکتی بزرگ در روند آموزش مدل استفاده نشده و مجموعه دادگان آموزشی کوچکی تشکیل می شود. راهکار دوم استفاده از روش های درون یابی اشاره شده برای سری با بسامد کمتر است، بنابراین لزوم استفاده از روش درون یابی در چنین حالتی بیشتر مشهود است.



شکل ۱۰-۱۱: تعریف پنجره خروجی برای دو سری با بسامد متفاوت

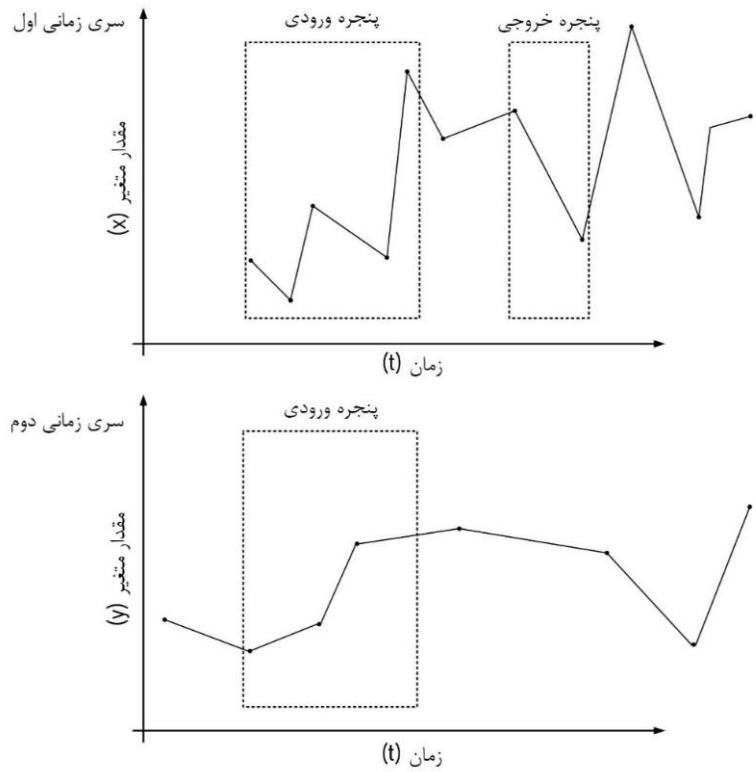
۳. **زمان ثبت اولین نمونه هر سری متفاوت بوده ولی بسامد برای همه سری های زمانی یکسان باشد:** در این صورت برای تعریف پنجره های ورودی و خروجی اولین نمونه را برای همه سری های زمانی، زمان ثبت اولین نمونه سری در نظر می گیریم که نمونه آن نسبت به سری های دیگر دیرتر ثبت شده است. نمونه هایی که زمان ثبت آن ها نسبت به نمونه اول سری زمانی با آخرین زمان ثبت نمونه اول، قبل تر است برای تعریف مجموعه دادگان مطابق شکل ۱۱-۱۱ در نظر گرفته نمی شود. در شکل ۱۱-۱۱ دو نمونه از سری زمانی

اول که پیش از ثبت اولین نمونه سـری دوم ثبت شـده اند برای تولید مجموعه دادگان در نظر گرفته نمی شوند.



شکل ۱۱-۱۱: سری های زمانی با زمان ثبت اولین نمونه متفاوت

۴. **هر دو پارامتر ثبت اولین نمونه و بسـامد برای هر سـری متفاوت باشـد:** این حالت ترکیب حللت های ۲ و ۳ اسـت. در این حللت علاوه بر اینکه زمان ثبت اولین نمونه را برای همه سـری ها مطابق حالت ۳ زمان ثبت اولین نمونه سـری که زمان ثبت نمونه اول آن آخرین اسـت در نظر می گیریم، برای برابر کردن تعداد نمونه ها نیز از روش های درون یابی اسـتفاده کرده و یا گام حرکتی پنجره را با بسـامد سـری زمانی که کمترین بسـامد را دارد تنظیم می کنیم. شکل ۱۲-۱۱ نمایش این حالت برای دو سری زمانی است.



شکل ۱۲-۱۱: دو سری زمانی با بسامد و زمان ثبت اولین نمونه متفاوت

۱۱.۲.۳ انتخاب مدل مناسب برای آموزش مدل پیش بینی کننده سری زمانی

برای انتخاب مدل مناسب برای آموزش مدل پیش بینی کننده دادگانی که دارای وابستگی زمانی هستند؛ دادگانی که مقادیر مطلوب مدل در آن ها وابسته به ورودی فعلی بلکه ورودی های گذشته است. ابتدا یک دسته بندی کلی برای این نوع از داده ها در نظر می گیریم. در اصل این دسته بندی یک روش تجربی برای درک بهتر بوده و عملا مرز مشخصی بین دسته ها نمی توان تعریف کرد. براساس این دسته بندی دادگان سری زمانی را به سه دسته مطابق شکل ۱۳-۱۱ تقسیم کرده و ویژگی ها و چالش های هرکدام از آن ها را بررسی می کنیم:

شکل ۱۳-۱۱: مقایسه حافظه تعریف شده برای سری های زمانی

۱. **سری های زمانی با حافظه کوتاه مدت[1]:** در این سری های زمانی مقدار گام آینده متغیر، $k+1$، سری از مقدار متغیر، k، در گام فعلی و تعدادی از گام های قبل تر از آن، $k-i, i=1,\dots,n$، محاسبه می شود. به عبارت دیگر پارامتر سری زمانی به تعدادی از گام های، دینامیک های، قبل از خود وابسته است. برای توسعه مدل پیش بینی کننده مناسب برای چنین سری های زمانی اغلب استفاده از مدل های مرسومی مانند شبکه های عصبی پرسپترون چندلایه[2]، ماشین بردار پشتیبان[3] و... اغلب دارای عملکرد مناسبی هستند. در صورتی که سری زمانی وابستگی بیشتری به گام های قبلی خود داشته باشد، تعداد گام های قبل از خود که به آن ها وابسته است بیشتر باشد، مدل های ذکر شده ممکن است خروجی مطلوبی نداشته و بهتر است از مدل های حافظه داری که دارای حافظه کوتاه مدت هستند استفاده کنیم. در ادامه این فصل به بررسی این مدل های حافظه دار خواهیم پرداخت.

۲. **سری های زمانی با حافظه کوتاه مدت و بلند مدت[4]:** این نوع سری های زمانی مستلزم استفاده از مدل های حافظه دار پیچیده تری هستند. در صورت استفاده از مدل های مرسوم مانند شبکه های عصبی پرسپترون چند لایه برای نوع از سری های زمانی لندازه بردار ورودی، پنجره ورودی، بزرگ است. مقدار پارامتر متغیر در گام k در این سری ها به تعداد بالائی از گام های گذشته خود، $k-i, i=0,\dots,k-1$، وابسته است. مدل های پیشنهاد شده برای این نوع سری ها، شبکه های بازگشتی دروازه دار است که علاوه با برطرف کردن مشکل مربوط به بعد بالای بردار ورودی، یک مدل پیش بینی کننده با عملکرد مطلوبی را برای این سری های زمانی ارائه می کند که مشابه حالت ۱

[1] Short term memory
[2] MLP
[3] SVM
[4] Long term memory

تعداد کمتری از گام های گذشته را به عنوان ورودی دریافت می کند.

۳. **سری های زمانی با حافظه بلند مدت:** این سری های زمانی رفتاری مشابه سری های زمانی مطرح شده در حالت ۱ را داشته و در آن ها مقدار پارامتر متغیر وابسته به تعداد محدودی از گام های پیش از خود است، با این تفاوت که فاصله گام آینده از گام هایی که به آن وابسته است بیشتر از حالت ۱ است. به بیانی دیگر این حالت همان حالت اول است که پنجره خروجی در آن در افق دورتری، $k+n$، تعریف می شود. براساس الگوی رفتاری این سری های زمانی و درجه غیرخطی آن ها می توان گستره وسیعی از مدل ها را برای پیش بینی این نوع سری های زمانی استفاده کرد؛ هم ساختارهای مرسوم تری مانند شبکه های عصبی پرسپترون چند لایه و هم انوع مدل های بازگشتی و حافظه دار برای تعریف مدل پیش بینی کننده این سری های زمانی کاربرد دارد. با توجه به فاصله پنجره ورودی و خروجی در این سری های زمانی در صورت استفاده از مدل های مرسومی مانند شبکه های عصبی پرسپترون چند لایه، مدل مذکور معمولا نسبت به مدل های معادلی که برای سری های زمانی حالت ۱ استفاده می شوند دارای ساختار پیچیده تری هستند.

۱۱.۳ شبکه های هاپفیلد[1]

شبکه های هاپفیلد یکی از اولین ساختارهای بازگشتی، حافظه دار، معرفی شده است. حافظه این شبکه یک حافظه انجمنی[2] بوده قادر به ذخیره سازی تعدادی الگوی ورودی است؛ این ذخیره سازی با تنظیم پارامترهای صورت می گیرد. ساختار این شبکه مشابه ماشین بولتزمن که در فصل ۱۰ بررسی کردیم است. در روند ذخیره سازی الگوهای ورودی، پارامترهای شبکه به گونه ای تنظیم می شوند تا مقدار تابع انرژی به ازای آن ورودی به حداقل مقدار ممکن برسد که مشابه روند آموزش ماشین بولتزمن با نمونه های مجموعه دادگان ورودی است. پس از اتمام روند ذخیره سازی، می توانیم الگوهای ذخیره شده را با استفاده از یک الگوی دلخواه، یکی از الگوهای ذخیره شده که به آن نویز اضافه شده باشد یا قسمتی از آن از بین رفته باشد، را بازیابی کنیم. روند بازیابی یک الگو مشابه روند تولید دادگان جدید براساس توزیع آموزش داده شده در ماشین های بولتزمن است. در ادامه با ارائه مثالی روند ذخیره سازی و بازیابی در شبکه هاپفیلد را بررسی می کنیم.

۱۱.۳.۱ ذخیره یک الگو (تصویر) در ساختار شبکه هاپفیلد

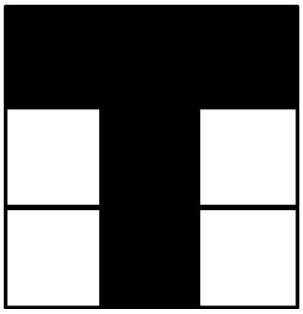
شکل ۱۴-۱۱ را در نظر بگیرید که تصویری با ابعاد، تعداد پیکسل[3]، کوچک از یک کاراکتر

[1] Hopfield networks

[2] Associative memory: حافظه ای که قادر به ذخیره سازی و بازیابی مواردی است که لزوما ارتباطی به هم ندارند

[3] Pixel

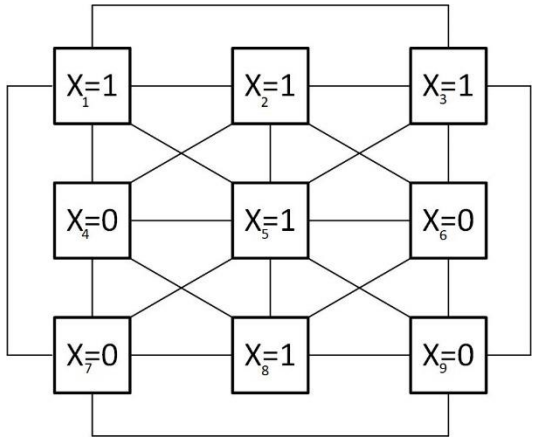
است. این تصویر یک الگوی باینری مشابه بردارهای ورودی ماشین های بولتزمن است؛ نقاط سیاه رنگ معادل عدد یک و نقاط سفید معادل عدد صفر در بردار ورودی هستند. بنابراین با کنار هم قرار دادن مقادیر پیکسل های این تصویر می توانیم آن را به یک بردار باینری مشابه رابطه ۶-۱۱ تبدیل کنیم.



شکل ۱۴-۱۱: تصویری از یک کاراکتر با ابعاد (تعداد پیکسل) کوچک

$$\mathbf{x} = \begin{bmatrix} x_1 = 1 \\ x_2 = 1 \\ x_3 = 1 \\ x_4 = 0 \\ x_5 = 1 \\ x_6 = 0 \\ x_7 = 0 \\ x_8 = 1 \\ x_9 = 0 \end{bmatrix} \qquad (۶-۱۱)$$

همان طور که بررسی ماشین های بولتزمن در فصل ۱۰ مطرح کردیم، هر بعد از بردار ورودی، معادل متغیر یک واحد از ماشین بولتزمن است. بنابراین مقادیر بردار ورودی معادل تصویر را مطابق شکل ۱۵-۱۱ در مقادیر متغیرهای یک ساختار شبکه هاپفیلد قرار می دهیم. ساختار شکل ۱۵-۱۱ ساختاری مشابه ساختار یک ماشین بولتزمن شکل ۱-۱۰ است که در آن هر یک از اتصالات بین واحد ها یک پارامتر وزن بوده و هر یک از واحدها دارای یک پارامتر بایاس است که به منظور جلوگیری از پیچیدگی پارامترهای بایاس و بعضی از پارامترهای وزن در شکل ۱۵-۱۱ نشان داده نشده است.



شکل ۱۵-۱۱: ساختار یک شبکه هاپفیلد

تابع انرژی شبکه هاپفیلد با در نظر گرفتن بردار رابطه ۶-۱۱ که همان تصویر نمایش داده شده در شکل ۱۴-۱۱ است، به عنوان ورودی مطابق رابطه ۷-۱۱ خواهد بود. در رابطه ۷-۱۱، x_i و x_j به ترتیب متغیر تصادفی واحد i و j، b_i بایاس واحد i و $w_{i\,j}$ وزن بین دو واحد i و j است. نماد θ زیروند تابع انرژی بیانگر مجموعه پارامتر های شبکه بوده و متشکل از کل بایاس ها و وزن های ماشین است.

$$E_{\theta(\boldsymbol{x})} = -\sum_{i=1}^{n} b_i x_i - \sum_{i=1}^{n-1} \sum_{j=i+1}^{n} w_{i,j} x_i x_j = -\boldsymbol{b}^T \boldsymbol{x} - \boldsymbol{x}^T W \boldsymbol{x} \qquad (۱۱\text{-}۷)$$

ذخیره سازی بردار ورودی در ساختار شبکه هاپفیلد با آموزش پارامترهای شبکه به منظور کاهش مقدار تابع انرژی رابطه، به روش هبیان[1] با استفاده از مشتقات زنجیره ای و گرادیان نزولی که در فصل ۱۰ برای ماشین های بولتزمن معرفی کردیم انجام می شود. بنابراین روند ذخیره سازی الگو در شبکه هاپفیلد، که معادل آموزش ماشین بولتزمن است، مطابق روابط ۸-۱۱ و ۹-۱۱ به ترتیب برای پارامترهای بایاس و وزن در هر گام آموزشی[2] است. به بیان دیگر ذخیره سازی یک الگوی ورودی در شبکه هاپفیلد معادل آموزش یک ماشین بولتزمن با مجموعه دادگانی متشکل از یک نمونه است، تفاوت آموزش ماشین بولتزمن و ذخیره سازی الگو در شبکه هاپفیلد در این است که از آنجایی که تعداد نمونه های مجموعه دادگان در شبکه هاپفیلد محدود است به جای تعریف توزیع برای مجموعه دادگان و استفاده از معیار واگرائی KL برای توزیع ها مطابق رابطه ۸-۱۰ که برای آموزش ماشین بولتزمن استفاده می‌کردیم، تابع هزینه شبکه هاپفیلد را همان تابع انرژی به ازای الگوی ورودی در نظر گرفته و با آموزش پارامترهای شبکه بر اساس گرادیان نزولی مقدار آن را

[1] Hebbian learning rule
[2] Epoch

کاهش می دهیم.

$$\boldsymbol{b}_{(k+1)} = \boldsymbol{b}_{(k)} + \eta \frac{\partial E_{(\theta)}}{\partial \boldsymbol{b}}(k) \qquad (۱۱-۸)$$

$$-\boldsymbol{x}_{(k)}$$

$$W_{(k+1)} = W_{(k)} + \eta \frac{\partial E_{(\theta)}}{\partial W}(k) \qquad (۱۱-۹)$$

$$-\boldsymbol{x}_{(k)}{}^{T}\boldsymbol{x}_{(k)}$$

در صورتی که تعداد الگوها برای ذخیره سازی بیشتر باشد در این صورت، اگر از روش گرادیان نزولی استفاده کنیم، در هرگام آموزشی مقادیر گرادیان برای تابع انرژی میانگین همه الگوها محاسبه شده و براساس آن پارامترهای شبکه هاپفیلد طبق روابط ۸-۱۱ و ۹-۱۱ تنظیم می شوند. این روند ذخیره سازی تا تعداد مشخصی گام آموزشی ادامه می یابد تا مقدار تابع انرژی میانگین به مقدار کوچک قابل قبولی همگرا شود.

همان طور که در ماشین های بولتزمن تعداد پارامتر های ماشین طبق رابطه ۲۷-۱۰ برای مدل کردن توزیع احتمالاتی دادگان ورودی محدود است، در شبکه های هاپفیلد نیز ظرفیت حافظه ساختار شبکه هاپفیلد برای ذخیره کردن الگوهای ورودی محدود است. بنابراین اگر تعداد الگوهایی که هدف ذخیره سازی آن ها در شبکه است افزایش باید، عملکرد شبکه برای ذخیره آن ها کاهش می باید، البته این کاهش با افزایش تعداد واحدهای شبکه هاپفیلد، واحد های پنهان، مشابه ماشین های بولتزمن می توان ظرفیت حافظه شبکه را تا حدی افزایش داد. حداکثر تعداد الگوهای قابل ذخیره، ظرفیت حافظه، در یک شبکه هاپفیلد متشکل از n واحد، شامل واحدهای پنهان و نمایان، در صورتی که بخواهیم این الگوها بدون خطا بازیابی شوند از رابطه ۱۰-۱۱ محاسبه می شود. در این رابطه a تعداد حالات ممکن برای مقادیر متغیرهای واحدهای شبکه است که از آنجایی که متغیرها در این قسمت باینری فرض شده است مقدار برابر دو است، همچنین در این رابطه ! نماد عملیات فاکتوریل[1] و C تعداد الگوهای قابل ذخیره است.

$$C \cong \frac{1}{2(2a-3)!!}\frac{n^{a-1}}{\log n} \qquad (۱۱-۱۰)$$

در صورتی که بخواهیم بخواهیم در یک ساختار شبکه هاپفیلد تعداد الگوهای بیشتری نسبت به آنچه از رابطه ۱۰-۱۱ حاصل می شود ذخیره کنیم در این صورت بازیابی الگوها با مقداری خطا همراه خواهد بود. در این صورت حداکثر تعداد الگوهای قابل ذخیره با پذیرفتن خطای بازیابی از رابطه ۱۱-۱۱ محاسبه می شود. در رابطه ۱۱-۱۱، α_a مقدار ثابت اسکالر، n تعداد واحدهای شبکه،

[1] Factorial

شـــامل واحدهای پنهان و نمایان، و a تعداد حالات ممکن برای مقادیر متغیرهای واحدهای شـــبکه است که از آنجایی که متغیرها در این قسمت باینری فرض شده است مقدار برابر ۲ است.

$$C \cong \alpha_a n^{a-1} \qquad (۱۱-۱۱)$$

رابطه ۱۱-۱۱ برای یک شـــبکه هاپفیلد با n واحد که متغیرهای آن به صـــورت باینری تعریف شده باشند مطابق رابطه ۱۲-۱۱ است.

$$C \cong 0.14n \qquad (۱۱-۱۲)$$

ظرفیت یک شبکه هاپفیلد ، بدون داشتن خطای بازیابی و با خطای بازسازی برای الگوی ورودی با ابعاد مطابق شکل ۱۵-۱۱ به ترتیب به صورت روابط ۱۳-۱۱ و ۱۴-۱۱ است.

$$C \cong \frac{9}{2\log 9} \sim 4 \qquad (۱۱-۱۳)$$

$$C \cong 0.14 \times 9 \sim 1 \qquad (۱۱-۱۴)$$

با توجه به وابستگی و شباهت بین الگوهای مختلف در عمل شبکه هاپفیلد بازیابی مناسبی برای تعداد نمونه هایی حتی کمتر از مقدار حاصل از روابط ۱۰-۱۱ و ۱۱-۱۱ ندارد. از آنجایی که در یک ساختار هاپفیلد ذخیره سازی الگوها براساس کاهش مقدار تابع انرژی تعریف می شود و از طرفی ممکن است الگوهای متفاوت دارای مقادیر انژری مشابهی باشند، بر این اساس الگوهای متفاوت با مقدار تابع انرژی یکسان می توانند در مرحله ذخیره سازی پارامترهای شبکه را به مقادیر مشابهی سوق دهند. در این صورت در مرحله بازیابی این مسئله باعث می شود الگوی بازیابی شده متناظر با الگوی مدنظر نبوده و بازیابی دارای مقادیر بزرگی از خطا باشـــد. با توجه به ماهیت مولد ســـاختار[1]، حتی ممکن است الگوی بازیابی شده در میان الگوهای ذخیره شده نباشد، بلکه یک الگوی متاثر از توزیع الگوهای ذخیره شـــده باشـــد. با این توضـــیحات می توانیم چنین اســـتنباط کنیم که شـــبکه هاپفیلد تضـمینی بر همگرائی الگوی وارد شـده در مرحله بازیابی به الگوی متناظر ذخیره شـده و یا حتی همگرائی به یکی از الگوهای ذخیره شده نداشته و فقط کاهش مقدار تابع انرژی آن در طی هر دو مرحله ذخیره سازی و بازیابی در این ساختار تضمین شده است. به عبارت دیگر مقدار تابع انرژی یک شبکه هاپفیلد چه با تنظیم پارامترها در مرحله ذخیره سازی و چه با به روزرسانی مقادیر متغیر هر واحد در مرحله بازیابی که در ادامه به آن اشـــاره خواهیم کرد، حتما کاهش پیدا می کند ولی الگویی که به ازای آن مقدار تابع انرژی کاهش یافته لزوما الگوی ذخیره شـــده و یا یکی از الگوهای ذخیره شده نیست.

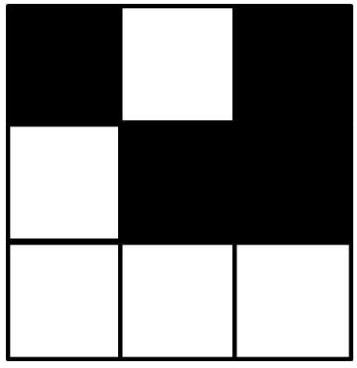
[1] مشابه ماشین های بولتزمن

۱۱.۳.۲ بازیابی الگو (تصویر) ذخیره شده در ساختار شبکه هاپفیلد

پس از اتمام مرحله ذخیره سازی الگوها در ساختار شبکه هاپفیلد، مقادیر پارامترهای تنظیم شده آن ذخیره شده و پس از آن تغییری نمی کنند. برای بازیابی الگوی ذخیره شده می توانیم از یک الگوی دلخواه مطابق رابطه ۱۵-۱۱ که بعد آن با بعد الگوهای ذخیره شده در ساختار مطابق رابطه ۶-۱۱ برابر باشد استفاده کنیم.

$$\mathbf{x}=\begin{bmatrix} x_1=1 \\ x_2=0 \\ x_3=1 \\ x_4=0 \\ x_5=1 \\ x_6=1 \\ x_7=0 \\ x_8=0 \\ x_9=0 \end{bmatrix} \tag{۱۱-۱۵}$$

در آغاز، مقادیر ابعاد مختلف این الگوی دلخواه رابطه ۱۵-۱۱ را در متغیرهای متناظر با هر بعد در ساختار شبکه قرار می دهیم. فرض کنیم الگوی رابطه ۱۵-۱۱ یک تصویر دلخواه مطابق شکل ۱۶-۱۱ بوده و هدف بازیابی الگوی شکل ۱۴-۱۱ باشد.



شکل ۱۶-۱۱: تصویر دلخواه ورودی

سپس مقادیر متغیر هر یک از واحدهای شبکه اعم از واحدهای پنهان و نمایان طبق رابطه ۱۶-۱۱ به روز رسانی می کنیم. در رابطه x_i و x_j مقادیر متغیر واحدهای i و j شبکه در هر گام

بازیابی، $w_{i\,j}$ وزن بین دو واحد i و j و b_i بایاس واحد i است. تابع $f_{(.)}$ نیز یک تابع علامت مطابق رابطه ۱۷-۱۱ است.

$$x_{i(k)} = f\left(\sum_{j=1}^{n-1} w_{i,j(k-1)} x_{j(k-1)} + b_{i(k-1)}\right) \qquad (۱۱-۱۶)$$

$$f(x) = \text{sign}(x) = \begin{cases} 1, & if \quad x > 0.5 \\ 0, & if \quad x \le 0.5 \end{cases} \qquad (۱۱-۱۷)$$

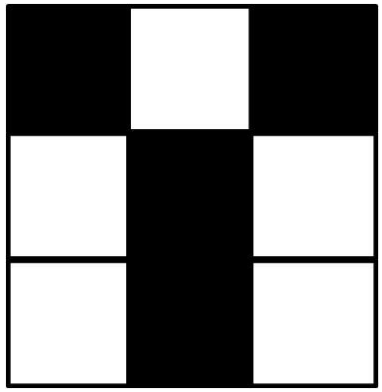
در صورتی که مقدار حاصل از رابطه ۱۶-۱۱ برای یک واحد از شبکه بزرگتر از ۰/۵ باشد در این صورت مقدار متغیر آن واحد جدا از اینکه قبلا چه مقداری داشت به مقدار یک و در صورتی که مقدار حاصل رابطه کوچکتر از ۰/۵ باشد مقدار متغیر آن واحد به مقدار صفر به روزرسانی می شود. در هر گام بازیابی باید یک بار متغیر همه واحدهای ساختار را به روزرسانی کنیم. گام های بازیابی تا جایی ادامه پیدا می کنند که با به روزرسانی واحد ها مقدار متغیر آن ها تغییر نکرده و با مقدار پیش از به روزرسانی، مقدار گام پیشین، برابر باشد. در این صورت مقادیر متغیرهای واحدهای نمایان، همان بعدهای الگو، بردار، بازیابی شده هستند که ممکن است با بردار دلخواهی که ابتدا در نظر گرفته و بازیابی ساختار را با آن آغاز کردیم متفاوت باشد. به این روند بازیابی که هر واحد به صورت جداگانه در یک زمان به روزرسانی می شود، به روزرسانی ناهمگام[1] گفته می شود. برای به روزرسانی همه ی واحد ها به صورت همزمان از روش به روز رسانی همگام مطابق رابطه ۱۸-۱۱ استفاده می کنیم، در این رابطه W و $\boldsymbol{b}$ به ترتیب ماتریس وزن و بردار بایاس شبکه مشابه روابط فصل ۱۰ هستند.

$$\mathbf{x}_{(k)} = f\left(W\mathbf{x}_{(k-1)} + \mathbf{b}^T\mathbf{x}_{(k-1)}\right) \qquad (۱۱-۱۸)$$

چنانچه برای روند بازسازی از یک الگوی کاملا تصادفی مطابق رابطه ۱۵-۱۱ استفاده کنیم که مقادیر ابعاد مختلف آن مشابه شکل ۱۶-۱۱ تفاوت محسوسی با الگوهای ذخیره شده داشته باشد، الگوی بازیابی شده ممکن است به صورت تصادفی یکی از الگوهای ذخیره شده باشد که لزوما بازیابی آن مدنظر نیست، حتی این احتمال وجود دارد که الگوی بازیابی هیچ کدام از الگوهای ذخیره شده نباشد. ولی در صورتی که بردار دلخواه مورد استفاده در روند بازیابی یک الگویی استفاده می‌کنیم یک نمونه تخریب شده[2]، الگوی ذخیره شده‌ای که به آن نویز اضافه شده باشد، الگوی ذخیره شده مشابه شکل ۱۷-۱۱ باشد، در این صورت احتمال اینکه ساختار شبکه هاپفیلد بتواند الگوی اصلی را براساس الگوی تخریب شده بازیابی کند بیشتر خواهد بود. با این توضیحات

[1] Asynchronous
[2] Corrupted pattern

می توانیم چنین برداشت کنیم که تنها کاربرد مناسب قابل تعریف برای ساختار شبکه هاپفیلد، بازیابی الگوهای تخریب شده است. مرحله بازیابی در ساختار شبکه هاپفیلد مشابه تولید دادگان در ماشین های بولتزمن است. این ساختار تنها یک حافظه با قابلیت ذخیره سازی تعداد محدود الگو بوده که با استفاده از نمونه تخریب شده، الگوهای ذخیره شده قابل بازیابی هستند. با این تفاسیر اگرچه این ساختار برای مدل سازی رفتار دادگانی مانند سری های زمانی مناسب نیست اما یکی از اولین مدل های حافظه دار معرفی شده است که زمینه‌ای برای توسعه ساختارهای پیچیده تر فراهم کرده است.



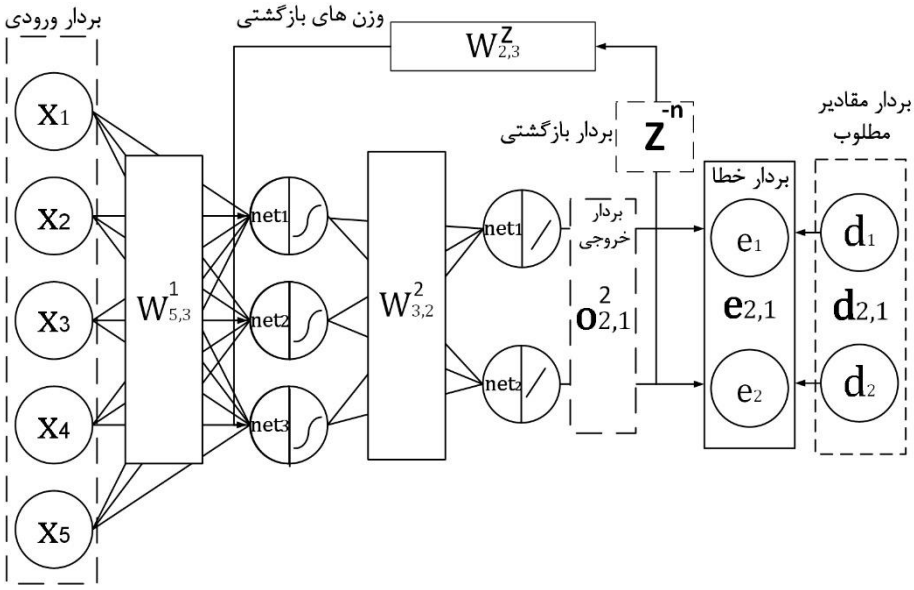
شکل ۱۷-۸: الگوی تصویری تخریب شده

۱۱.۴ شبکه های بازگشتی جردن[1]

شبکه عصبی جردن یکی از مدل های بازگشتی است که به صورت یک شبکه عصبی دو لایه تعریف شده و دارای یک حافظه کوتاه مدت است. فرض کنیم که برای مجموعه دادگانی مشابه شکل ۱-۱۱ بخواهیم یک مدل شبکه عصبی پرسپترون چند لایه پیش بینی کننده تعریف کنیم، در صورتی که طول پنجره ورودی را برای آموزش چنین مدلی افزایش دهیم، بعد ورودی بالاتر، عملکرد مدل در پیش بینی مقادیر پنجره گام آینده، پنجره خروجی، به ویژه در مواردی که سری زمانی رفتاری با درجه غیرخطی بالاتری داشته باشد، بهبود پیدا خواهد کرد. بدیهی است که این افزایش بعد ورودی مستلزم استفاده از ساختار شبکه پرسپترون پیچیده تری است که تعداد نورون‌های، و در مواردی تعداد لایه، بیشتری نسبت به حالتی که بعد ورودی، اندازه پنجره ورودی، کوچک تر است دارد. این گسترش ساختار شبکه پرسپترون بار محاسباتی را افزایش می دهد، علاوه بر این ممکن است این گسترش ساختار لزوما باعث بهبود عملکرد نشود. برای رفع این معضل مدل های بازگشتی متنوعی ارائه شده اند که شبکه جردن یکی از این ساختارها است. این شبکه قادر است با استفاده از بردار خروجی نمونه های قبل بدون نیاز به بالا بردن اندازه پنجره ورودی، بعد

[1] Jordan neural network

ورودی، پیش بینی مناسبی برای مقادیر گام آینده ارائه کند. ساختار این شبکه مطابق شکل ۱۸-۱۱ است که در آن سه نورون برای لایه پنهان و دو نورون برای لایه خروجی با فرض بردار ورودی با بعد پنج در نظر گرفته شده است.



شکل ۱۸-۱۱: ساختار شبکه جردن

با توجه به شکل ۱۸-۱۱ در ساختار شبکه جردن بردار خروجی مربوط به نمونه‌ای که قبلا به عنوان ورودی شبکه استفاده شده است، $\boldsymbol{Z^{-n}}$، با اعمال وزن های مناسب، $\boldsymbol{W^z}$، به عنوان قسمتی از ورودی به همراه بردار ورودی نمونه فعلی استفاده می شود که به آن بردار بازگشتی اطلاق می شود این بردار بازگشتی همان حافظه شبکه است. این بردار بازگشتی می تواند مربوط به یک، $\boldsymbol{Z^{-n}, n = 1}$، یا چند نمونه قبل تر از نمونه فعلی باشد. در ابتدای آغاز هر گام آموزشی ممکن است این بردار بازگشتی قابل تعریف نباشد، به عنوان مثال اگر بردار خروجی یک نمونه قبل تر را به عنوان بردار بازگشتی در نظر بگیریم، در زمان وارد شدن اولین نمونه از آنجایی که نمونه ای قبل از آن وارد نشده است، بردار بازگشتی، بردار خروجی نمونه قبل، قابل تعریف نیست در این موارد می توانیم مقدار بردار بازگشتی را صفر در نظر بگیریم. با توجه به این که بردار بازگشتی در این ساختار فقط از یک نمونه مربوط به گام های قبلی نشئت گرفته است لذا این ساختار دارای حافظه‌ای با ظرفیت محدود، کوتاه مدت، است که با وارد شدن هر بردار ورودی جدید مقدار بردار بازگشتی، حافظه، نیز تغییر می کند.

با توجه به ساختار شکل ۱۸-۱۱ رابطه پیشرو لایه اول برای یک شبکه جردن به صورت رابطه ۱۹-۱۱ تعریف می شود:

$$o^1_{i(k)} = f(\sum_{j=1}^{n=5} w^1_{i,j(k)} x_{j(k)} + \sum_{j=1}^{n=2} w^z_{i,j(k)} o^2_{j(k-1)} + b_{i(k)}), i = 1,...,3 \quad (۱۱-۱۹)$$

رابطه ۱۹-۱۱ را می توان به صورت روابط ۲۰-۱۱ و ۲۱-۱۱ به فرم برداری نیز نمایش دهیم:

$$\mathbf{net}^1_{1\times 3(k)} = \mathbf{x}^T_{5\times 1(k)} \times W^1_{5\times 3(k)} + \mathbf{o}^{2^T}_{2\times 1(k-1)} \times W^z_{2\times 3(k)} \quad (۱۱-۲۰)$$

$$\mathbf{o}^1_{1\times 3(k)} = f(\mathbf{net}^1_{1\times 3(k)}) \quad (۱۱-۲۱)$$

در رابطه ۱۸-۱۱ و ۲۰-۱۱، $f_{(.)}$ یک تابع فعال ساز غیرخطی است. رابطه پیشرو لایه دوم مطابق رابطه ۲۲-۱۱ است:

$$o^2_{i(k)} = f(\sum_{j=1}^{n=3} w^2_{i,j(k)} o^1_{j(k)} + b_{i(k)}), i = 1,2 \quad (۱۱-۲۲)$$

رابطه ۲۲-۱۱ را می توان به صورت روابط ۲۳-۱۱ و ۲۴-۱۱ به فرم برداری نیز نمایش دهیم:

$$\mathbf{net}^2_{1\times 2(k)} = \mathbf{o}^{1^T}_{3\times 1(k)} \times W^2_{3\times 2(k)} \quad (۱۱-۲۳)$$

$$\mathbf{o}^2_{1\times 2(k)} = f(\mathbf{net}^2_{1\times 2(k)}) \quad (۱۱-۲۴)$$

در رابطه ۲۲-۱۱ و ۲۴-۱۱، $f_{(.)}$ یک تابع فعال ساز خطی مطابق ساختار شکل ۱۸-۱۱ فرض شده است.

برای آموزش یک شبکه جردن به روش گرادیان نزولی، روابط مشتقات زنجیره ای را برای وزن های لایه خروجی، وزن های لایه پنهان و وزن های بازگشتی به ترتیب به صورت روابط ۲۵-۱۱، ۲۶-۱۱ و ۲۷-۱۱ با فرض رابطه مجموع مربعات خطا به عنوان تابع هزینه ساختار، E، تعریف می کنیم:

$$\frac{\partial E}{\partial W^2} = \underbrace{\frac{\partial E}{\partial \mathbf{e}}}_{e} \underbrace{\frac{\partial \mathbf{e}}{\partial \mathbf{o}^2}}_{-1} \underbrace{\frac{\partial \mathbf{o}^2}{\partial \mathbf{net}^2}}_{f'_{(\mathbf{net})}} \underbrace{\frac{\partial \mathbf{net}^2}{\partial W^2}}_{\mathbf{o}^1} = -e f'_{(\mathbf{net}_{(k)})} \mathbf{o}^1 \quad (۱۱-۲۵)$$

$$\frac{\partial E}{\partial W^1} = \frac{\partial E}{\partial \mathbf{e}} \frac{\partial \mathbf{e}}{\partial \mathbf{o}^2} \frac{\partial \mathbf{o}^2}{\partial \mathbf{net}^2} \frac{\partial \mathbf{net}^2}{\partial \mathbf{o}^1} \frac{\partial \mathbf{o}^1}{\partial \mathbf{net}^1} \frac{\partial \mathbf{net}^1}{\partial W^1} \quad (۱۱-۲۶)$$

$\mathbf{x}$

$$\frac{\partial E}{\partial W^z}=\frac{\partial E}{\partial \mathbf{e}}\frac{\partial \mathbf{e}}{\partial \mathbf{o}^2}\frac{\partial \mathbf{o}^2}{\partial \mathbf{net}^2}\frac{\partial \mathbf{net}^2}{\partial \mathbf{o}^1}\frac{\partial \mathbf{o}^1}{\partial \mathbf{net}^1}\frac{\partial \mathbf{net}^1}{\partial W^z} \quad (۱۱-۲۷)$$

$$\frac{\partial \mathbf{net}^1}{\partial W^z}=\mathbf{o}^2_{(k-1)}+W^z\frac{\partial \mathbf{o}^2_{(k-1)}}{\partial W^z} \quad (۱۱-۲۸)$$

مقدار پارامترهای مربوط به وزن های لایه خروجی، لایه ورودی و وزن های بازگشتی در هرگام با استفاده از روابط ۱۱-۲۵ تا ۱۱-۲۷ به ترتیب به صورت روابط ۱۱-۲۹ تا ۱۱-۳۱ است:

$$W^2_{(k+1)}=W^2_{(k)}-\eta\frac{\partial E}{\partial W^2}(k) \quad (۱۱-۲۹)$$

$$W^1_{(k+1)}=W^1_{(k)}-\eta\frac{\partial E}{\partial W^1}(k) \quad (۱۱-۳۰)$$

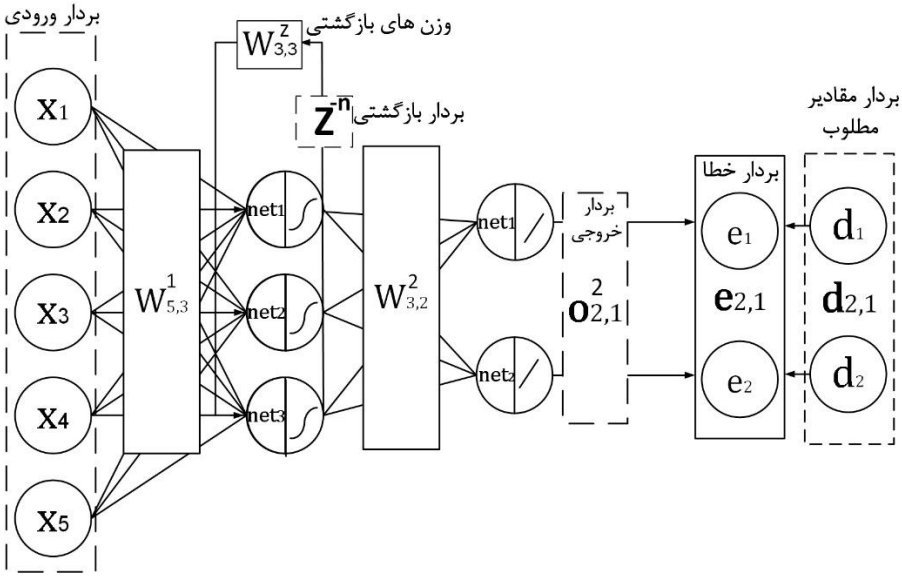
$$W^z_{(k+1)}=W^z_{(k)}-\eta\frac{\partial E}{\partial W^z}(k) \quad (۱۱-۳۱)$$

در روابط ۱۱-۲۹ تا ۱۱-۳۱، η نرخ یادگیری است که به صورت یک اسکالر در بازه صفر تا یک تعریف می شود.

۱۱.۵ شبکه های بازگشتی المن[1]

شبکه المن نیز یک شبکه بازگشتی بوده و ساختار آن به ساختار شبکه جردن شباهت دارد، تفاوت بین این شبکه با شبکه جردن در این است که در ساختار جردن بردار بازگشتی مربوط به حافظه به صورت بردار خروجی نهایی شبکه به ازای یکی از نمونه های قبلی تعریف می شد در حالی که در ساختار المن این بردار بازگشتی بردار پنهان، خروجی لایه پنهان، متناظر با یکی از نمونه های قبل است. ساختار شبکه المن نیز مشابه شبکه جردن دارای یک حافظه کوتاه مدت است. شکل ۱۱-۱۹ ساختار یک شبکه بازگشتی المن را نشان می دهد. بردار بازگشتی در این ساختار نیز مشابه شبکه جردن، در صورتی که قابل تعریف نباشد مقدار صفر خواهد داشت. در ساختار شکل ۱۱-۱۹، سه نورون برای لایه پنهان و دو نورون برای لایه خروجی با فرض بردار ورودی با بعد پنج در نظر گرفته شده است.

[1] Elman neural network



شکل ۱۹-۱۱: ساختار شبکه المن

روابط پیشرو لایه پنهان و لایه خروجی برای یک شبکه جردن مشابه ساختار شکل ۱۹-۱۱ به ترتیب مطابق روابط ۳۲-۱۱ تا ۳۷-۱۱ است، طبق این روابط بردار بازگشتی که همان بردار پنهان نمونه قبلی (نمونه گام قبل) است که با اعمال وزن های مناسب به عنوان قسمتی از ورودی گام فعلی در نظر گرفته می شود:

$$o^1_{i(k)} = f\left(\sum_{j=1}^{n=5} w^1_{i,j(k)} x_{j(k)} + \sum_{j=1}^{n=3} w^z_{i,j(k)} o^1_{j(k-1)} + b_{i(k)}\right), i = 1,\ldots,3 \quad (۱۱-۳۲)$$

$$o^2_{i(k)} = f\left(\sum_{j=1}^{n=3} w^2_{i,j(k)} o^1_{j(k)} + b_{i(k)}\right), i = 1,2 \quad (۱۱-۳۳)$$

روابط پیشرو ۳۲-۱۱ و ۳۳-۱۱ را می توانیم به فرم برداری به ترتیب به صورت روابط ۳۴-۱۱، ۳۵-۱۱ و ۳۶-۱۱ و ۳۷-۱۱ نشان دهیم:

$$\mathbf{net}^1_{1\times3(k)} = \mathbf{x}^T_{5\times1(k)} \times W^1_{5\times3(k)} + \mathbf{o}^{1^T}_{3\times1(k-1)} \times W^z_{3\times3(k)} \quad (۱۱-۳۴)$$

$$\mathbf{o}^1_{1\times3(k)} = f\left(\mathbf{net}^1_{1\times3(k)}\right) \quad (۱۱-۳۵)$$

$$\mathbf{net}^2_{1\times2(k)} = \mathbf{o}^{1^T}_{3\times1(k)} \times W^2_{3\times2(k)} \quad (۱۱-۳۶)$$

$$\mathbf{o}^2_{1\times2(k)} = f\left(\mathbf{net}^2_{1\times2(k)}\right) \quad (۱۱-۳۷)$$

برای آموزش وزن های لایه خروجی، لایه ورودی و بازگشتی یک شبکه المن مشتقات زنجیره ای، با فرض مجموع مربعات خطا به عنوان تابع هزینه به ترتیب طبق روابط ۳۸-۱۱ تا ۴۱-۱۱ تعریف می شوند:

$$\frac{\partial E}{\partial W^2}=\frac{\partial E}{\partial \mathbf{e}}\frac{\partial \mathbf{e}}{\partial \mathbf{o}^2}\frac{\partial \mathbf{o}^2}{\partial \mathbf{net}^2}\frac{\partial \mathbf{net}^2}{\partial W^2} \tag{۱۱-۳۸}$$

$$\frac{\partial E}{\partial W^1}=\frac{\partial E}{\partial \mathbf{e}}\frac{\partial \mathbf{e}}{\partial \mathbf{o}^2}\frac{\partial \mathbf{o}^2}{\partial \mathbf{net}^2}\frac{\partial \mathbf{net}^2}{\partial \mathbf{o}^1}\frac{\partial \mathbf{o}^1}{\partial \mathbf{net}^1}\frac{\partial \mathbf{net}^1}{\partial W^1} \tag{۱۱-۳۹}$$

$$\frac{\partial E}{\partial W^z}=\frac{\partial E}{\partial \mathbf{e}}\frac{\partial \mathbf{e}}{\partial \mathbf{o}^2}\frac{\partial \mathbf{o}^2}{\partial \mathbf{net}^2}\frac{\partial \mathbf{net}^2}{\partial \mathbf{o}^1}\frac{\partial \mathbf{o}^1}{\partial \mathbf{net}^1}\frac{\partial \mathbf{net}^1}{\partial W^z} \tag{۱۱-۴۰}$$

$$\frac{\partial \mathbf{net}^1}{\partial W^z}=\mathbf{o}^1_{(k-1)}+W^z\frac{\partial \mathbf{o}^1_{(k-1)}}{\partial W^z} \tag{۱۱-۴۱}$$

مقدار پارامترهای مربوط به وزن های لایه خروجی، لایه ورودی و وزن های بازگشتی در هرگام با استفاده از روابط ۳۸-۱۱ تا ۴۱-۱۱ به ترتیب به صورت روابط ۴۲-۱۱ تا ۴۴-۱۱ است:

$$W^2_{(k+1)}=W^2_{(k)}-\eta\frac{\partial E}{\partial W^2}(k) \tag{۱۱-۴۲}$$

$$W^1_{(k+1)}=W^1_{(k)}-\eta\frac{\partial E}{\partial W^1}(k) \tag{۱۱-۴۳}$$

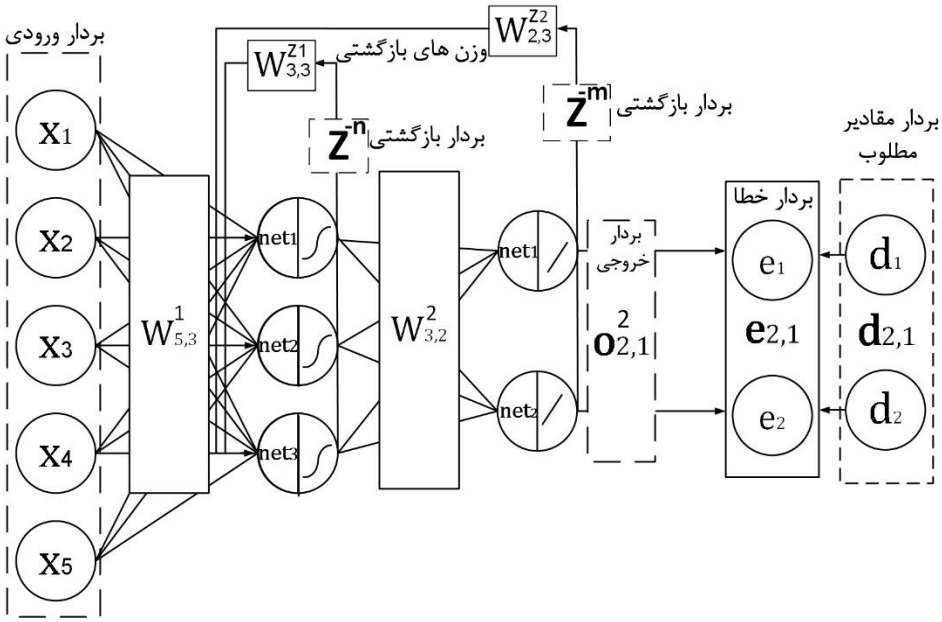
$$W^z_{(k+1)}=W^z_{(k)}-\eta\frac{\partial E}{\partial W^z}(k) \tag{۱۱-۴۴}$$

۱۱.۶ شبکه های بازگشتی المن-جردن[1]

ساختار شبکه المن جردن از ترکیب دو شبکه المن و جردن حاصل می شود. در این شبکه هم بردار خروجی و هم بردار پنهان متناظر با یکی از نمونه های قبلی به عنوان بخشی از ورودی شبکه در نظر گرفته می شود. نمونه قبلی که براساس آن بردار های بازگشتی تعریف می شوند ممکن است برای هر دو بردار بازگشتی از خروجی و بردار بازگشتی از خروجی لایه پنهان یکسان باشد و یا

[1] Elman-Jordan neural network

از نمونه های متفاوتی برای تشکیل این بردارهای بازگشتی استفاده شود. شبکه المن-جردن در مقایسه با ساختار های المن یا جردن ظرفیت حافظه بیشتری، دو برابر، داشته و متعاقباً دارای بار محاسباتی بیشتری نیز است. شکل ۲۰-۱۱ ساختار یک شبکه المن-جردن را نشان می دهد که در آن برای لایه پنهان سه و لایه خروجی دو نورون با فرض ورودی با بعد پنج در نظر گرفته شده است.



شکل ۲۰-۱۱: ساختار شبکه المن-جردن

برای یک ساختار المن-جردن مشابه شکل ۲۰-۱۱ روابط ۴۵-۱۱ و ۴۶-۱۱ را به ترتیب برای محاسبه رابطه پیشرو لایه پنهان و لایه خروجی داریم:

$$o^1_{i(k)} = f\left(\sum_{j=1}^{n=5} w^1_{i,j(k)} x_{j(k)} + \sum_{j=1}^{n=3} w^{z1}_{i,j(k)} o^1_{j(k-1)} + \sum_{j=1}^{n=2} w^{z2}_{i,j(k)} o^2_{j(k-1)} + b_{i(k)}\right)$$

$$i = 1,...,3 \qquad (۱۱-۴۵)$$

$$o^2_{i(k)} = f\left(\sum_{j=1}^{n=3} w^2_{i,j(k)} o^1_{j(k)} + b_{i(k)}\right), i = 1,2 \qquad (۱۱-۴۶)$$

روابط ۴۵-۱۱ و ۴۶-۱۱ را می توانیم به فرم ماتریسی به ترتیب به صورت روابط ۴۷-۱۱ تا ۵۰-۱۱ نیز نشان دهیم:

$$\mathbf{net}^{1}_{1\times 3(k)} = \mathbf{x}^{T}_{5\times 1(k)} \times W^{1}_{5\times 3} + \mathbf{o}^{1^{T}}_{3\times 1(k-1)} \times W^{z1}_{3\times 3} + \mathbf{o}^{2^{T}}_{2\times 1(k-1)} \times W^{z2}_{2\times 3} \quad (۱۱-۴۷)$$

$$\mathbf{o}^{1}_{1\times 3(k)} = f(\mathbf{net}^{1}_{1\times 3(k)}) \quad (۱۱-۴۸)$$

$$\mathbf{net}^{2}_{1\times 2(k)} = \mathbf{o}^{1^{T}}_{3\times 1(k)} \times W^{2}_{3\times 2} \quad (۱۱-۴۹)$$

$$\mathbf{o}^{2}_{1\times 2(k)} = f(\mathbf{net}^{2}_{1\times 2(k)}) \quad (۱۱-۵۰)$$

مشتقات زنجیره ای برای وزن های لایه خروجی، لایه ورودی، بازگشتی از لایه پنهان و بازگشتی از لایه خروجی به ترتیب طبق روابط ۱۱-۵۱ تا ۱۱-۵۴ است:

$$\frac{\partial E}{\partial W^{2}} = \frac{\partial E}{\partial \mathbf{e}} \frac{\partial \mathbf{e}}{\partial \mathbf{o}^{2}} \frac{\partial \mathbf{o}^{2}}{\partial \mathbf{net}^{2}} \frac{\partial \mathbf{net}^{2}}{\partial W^{2}} \quad (۱۱-۵۱)$$

$$\frac{\partial E}{\partial W^{1}} = \frac{\partial E}{\partial \mathbf{e}} \frac{\partial \mathbf{e}}{\partial \mathbf{o}^{2}} \frac{\partial \mathbf{o}^{2}}{\partial \mathbf{net}^{2}} \frac{\partial \mathbf{net}^{2}}{\partial \mathbf{o}^{1}} \frac{\partial \mathbf{o}^{1}}{\partial \mathbf{net}^{1}} \frac{\partial \mathbf{net}^{1}}{\partial W^{1}} \quad (۱۱-۵۲)$$

$$\frac{\partial E}{\partial W^{z1}} = \frac{\partial E}{\partial \mathbf{e}} \frac{\partial \mathbf{e}}{\partial \mathbf{o}^{2}} \frac{\partial \mathbf{o}^{2}}{\partial \mathbf{net}^{2}} \frac{\partial \mathbf{net}^{2}}{\partial \mathbf{o}^{1}} \frac{\partial \mathbf{o}^{1}}{\partial \mathbf{net}^{1}} \frac{\partial \mathbf{net}^{1}}{\partial W^{z1}} \quad (۱۱-۵۳)$$

$$\frac{\partial E}{\partial W^{z2}} = \frac{\partial E}{\partial \mathbf{e}} \frac{\partial \mathbf{e}}{\partial \mathbf{o}^{2}} \frac{\partial \mathbf{o}^{2}}{\partial \mathbf{net}^{2}} \frac{\partial \mathbf{net}^{2}}{\partial \mathbf{o}^{1}} \frac{\partial \mathbf{o}^{1}}{\partial \mathbf{net}^{1}} \frac{\partial \mathbf{net}^{1}}{\partial W^{z2}} \quad (۱۱-۵۴)$$

$$\frac{\partial \mathbf{net}^{1}}{\partial W^{z1}} = \mathbf{o}^{1}_{(k-1)} + W^{z1} \frac{\partial \mathbf{o}^{1}_{(k-1)}}{\partial W^{z1}} \quad (۱۱-۵۵)$$

$$\frac{\partial \mathbf{net}^{1}}{\partial W^{z2}} = \mathbf{o}^{1}_{(k-1)} + W^{z2} \frac{\partial \mathbf{o}^{1}_{(k-1)}}{\partial W^{z2}} \quad (۱۱-۵۶)$$

با توجه به روابط مشـــتقات زنجیره ای ۱۱-۵۱ تا ۱۱-۵۴ مقدار این پارامترها در هر گام طبق روابط ۱۱-۵۷ تا ۱۱-۶۰ محاسبه می شود:

$$W^{2}_{(k+1)} = W^{2}_{(k)} - \eta \frac{\partial E}{\partial W^{2}}(k) \quad (۱۱-۵۷)$$

$$W^{1}_{(k+1)} = W^{1}_{(k)} - \eta \frac{\partial E}{\partial W^{1}}(k) \quad (۱۱-۵۸)$$

$$W_{(k+1)}^{z1} = W_{(k)}^{z1} - \eta \frac{\partial E}{\partial W^{z1}}(k) \quad (۱۱-۵۹)$$

$$W_{(k+1)}^{z2} = W_{(k)}^{z2} - \eta \frac{\partial E}{\partial W^{z2}}(k) \quad (۱۱-۶۰)$$

هر یک از ساختارهای المن، جردن و المن-جردن مدل هایی دارای حافظه کوتاه مدت هستند که می توانند به عنوان یک مدل خود هم بسته استفاده شوند. این مدل ها می توانند برای شبیه سازی رفتار دادگانی که نمونه های آن ها به هم وابستگی زمانی دارند مناسب باشند، البته با توجه به ظرفیت حافظه مدل عملکرد ساختار در مواردی که وابستگی زمانی محدودتر است، بهتر است. نسبت به رفتار دادگان و درجه غیرخطی بودن آن ها، هر یک از این مدل ها با انتخاب نمونه مناسب برای بردار حافظه و تعیین تعداد نورون در لایه پنهان و تابع فعال ساز مناسب می تواند عملکرد مطلوبی داشته باشد. بدیهی است که بعد بردار بازگشتی حاصل از خروجی، برابر با بعد خروجی شبکه و بعد بردار بازگشتی حاصل از بردار پنهان برابر تعداد نورون های لایه پنهان، بعد بردار پنهان، است.

ساختارهای بازگشتی المن، جردن و المن-جردن همواره به صورت یک ساختار دو لایه تعریف می شوند، بنابراین تغییر ظرفیت ساختار فقط با تغییر تعداد نورون های ساختار امکان پذیر بوده و لایه دیگری به ساختار اضافه نمی شود، البته همواره می توان بسته به نیاز و رفتار مجموعه دادگان ساختارهای بازگشتی با تعداد لایه دلخواه تعریف کنیم که مقدار خروجی هر لایه در گام های گذشته به عنوان قسمتی از ورودی آن لایه و یا هر لایه دیگری در ساختار باشد اما این ساختار تعریف شده به عنوان یک ساختار المن، جردن و یا المن-جردن شناخته نمی شود. با وجود محدودیت در تعداد لایه های ساختارهای المن، جردن و المن-جردن همچنان می توان در صورت وجود وابستگی زمانی در دادگان این ساختارها را در قالب ساختارهای دو لایه مرسوم مانند خودرمزگذارها که در فصل ۱۰ بررسی کردیم استفاده کرد.

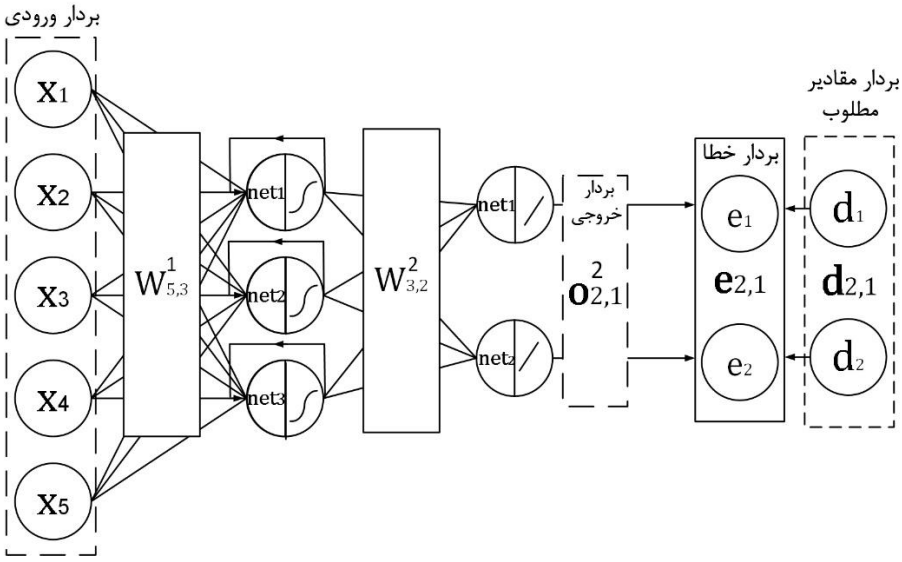
در هریک از ساختارهای المن، جردن و المن-جردن که بررسی کردیم و همچنین سایر ساختارهایی که در ادامه به بررسی آن ها خواهیم پرداخت همواره برای تشکیل بردار بازگشتی از یک نمونه که پیشتر به عنوان بردار ورودی وارد ساختار شده است استفاده می کنیم، به بیانی دیگر وابستگی زمانی بین نمونه های مختلف به صورت تعریف بردار بازگشتی و قرار گرفتن آن در کنار بردار ورودی مدل سازی می شود. بنابراین برای آموزش این ساختار ها، نمونه ها در مجموعه دادگان آموزشی همواره باید به ترتیب زمانی مرتب شده باشند؛ استفاده از ترتیب تصادفی[1] در نمونه های مجموعه دادگان که یک ترفند مرسوم برای بهبود روند آموزش مدل های دارای آموزش با

[1] Shuffling

سرپرست مبتنی بر گرادیان، محسوب می شود در این مدل های بازگشتی اشتباه است زیرا در این صورت نمونه قبلی که بردار بازگشتی براساس آن تعریف می شود لزوما مربوط به گام گذشته مدنظر نبوده و بدین ترتیب بردار بازگشتی متناظر آن نیز به درستی تعریف نشده و در نهایت روند شبیه سازی رفتار وابسته دادگان مطلوب نخواهد بود.

۱۱.۷ ساختار نورون ها با پسخور محلی[1]

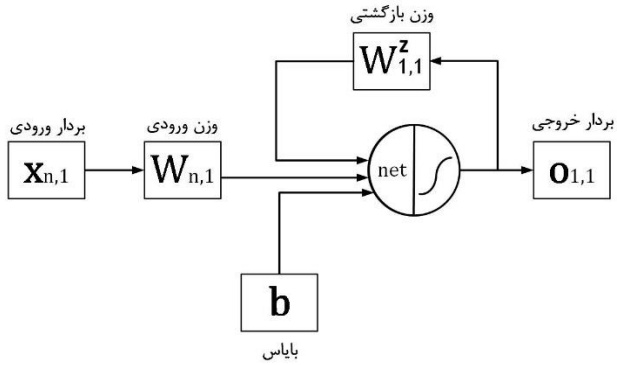
تا کنون مدل های بازگشتی بررسی شده، به صورت ساختارهای ثابتی بودند که انعطاف پذیری کمی داشته و امکان تغییر ساختار در آن ها محدود بود. اما با توجه به گسترش شبکه های عصبی ژرف[2] و تعریف ساختارهای متنوع، توسعه ساختارهای بازگشتی که انعطاف پذیری مناسبی داشته باشند و بتوان آن ها را در سایر ساختارهای مرسوم عمیق ادغام کرد مورد توجه قرار گرفته است. یکی از ساختارهای بازگشتی، ساختارهای بازگشتی ساده[3] هستند که در لایه های آن ها از نورون های بازگشتی با پسخور محلی استفاده شده است. شکل ۲۱-۱۱ یک ساختار بازگشتی ساده را نشان می دهد که در لایه اول (پنهان) آن از نورون هایی با پسخور محلی استفاده شده است. برخلاف ساختارهای المن، جردن و المن-جردن که در آن ها بردار بازگشتی به عنوان قسمتی از ورودی همه نورون های لایه پنهان در نظر گرفته می شد، در ساختار بازگشتی ساده پسخور هر نورون فقط به همان نورون وارد می شود.



شکل ۲۱-۱۱: ساختار بازگشتی ساده

[1] Local feedback
[2] Deep neural network
[3] Vanilla recurrent networks

شـکل ۲۲-۱۱ یک نورون با پسـخور محلی را نشـان می دهد که دارای یک پسـخور از خروجی متناظر با ورودی ان مرحله قبل اسـت. این پسـخور همانند بردار ورودی دارای وزن های متناسـب خود است.



شکل ۲۲-۱۱: ساختار یک نورون با پسخور محلی

روابط پیشرو یک نورون با پسخور محلی مطابق رابطه ۶۱-۱۱ است:

$$o_{(k)} = f\left(\sum_{j=1}^{n} w^1_{i,j(k)} x_{j,1(k)} + w^z_{1,1(k)} o_{1,1(k-1)} + b_{(k)}\right) \quad (۱۱-۶۱)$$

فرم ماتریسی رابطه ۶۱-۱۱ را می توانیم به صورت رابطه ۶۲-۱۱ و ۶۳-۱۱ نمایش دهیم:

$$\mathbf{net}^1_{1\times1(k)} = \mathbf{x}^T_{n\times1(k)} \times W^1_{n\times1(k)} + \mathbf{o}^T_{1\times1(k-1)} \times W^z_{1\times1(k)} + \mathbf{b}_{1\times1(k)} \quad (۱۱-۶۲)$$

$$\mathbf{o}_{1\times1(k)} = f\left(\mathbf{net}_{1\times1(k)}\right) \quad (۱۱-۶۳)$$

برای آموزش وزن های ورودی و بازگشـتی و بایاس یک نورون با پسـخور محلی به ترتیب روابط مشـتقات زنجیره ای ۶۴-۱۱ تا ۶۷-۱۱ را داریم (در این روابط مشـتقات مربوط به سایر قسـمت های ساختار نمایش داده نشده اند):

$$\frac{\partial E}{\partial W} = \cdots \frac{\partial \mathbf{o}}{\partial \mathbf{net}} \frac{\partial \mathbf{net}}{\partial \mathbf{o}} \frac{\partial \mathbf{o}}{\partial \mathbf{net}} \frac{\partial \mathbf{net}}{\partial W} \quad (۱۱-۶۴)$$

$$\frac{\partial E}{\partial W^z} = \cdots \frac{\partial \mathbf{o}}{\partial \mathbf{net}} \frac{\partial \mathbf{net}}{\partial \mathbf{o}} \frac{\partial \mathbf{o}}{\partial \mathbf{net}} \frac{\partial \mathbf{net}}{\partial W^z} \quad (۱۱-۶۵)$$

$$\frac{\partial \mathbf{net}}{\partial W^z} = \mathbf{o}_{(k-1)} + W^z \frac{\partial \mathbf{o}_{(k-1)}}{\partial W^z} \tag{۱۱-۶۶}$$

$$\frac{\partial E}{\partial \mathbf{b}} = \cdots \frac{\partial \mathbf{o}}{\partial \mathbf{net}} \frac{\partial \mathbf{net}}{\partial \mathbf{o}} \frac{\partial \mathbf{o}}{\partial \mathbf{net}} \frac{\partial \mathbf{net}}{\partial \mathbf{b}} \tag{۱۱-۶۷}$$

1

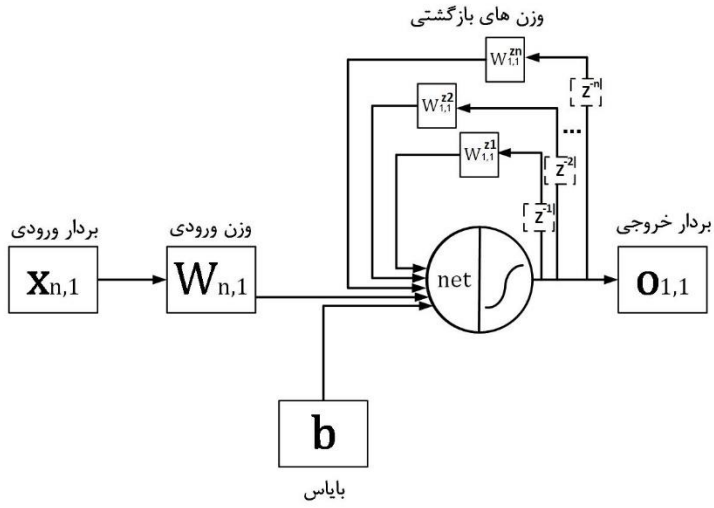
بر اساس روابط ۱۱-۶۴ تا ۱۱-۶۷ مقدار پارامترهای وزن ورودی و بازگشتی و بایاس یک نورون با پسخور محلی به صورت روابط ۱۱-۶۸ تا ۱۱-۷۰ است:

$$W_{(k+1)} = W_{(k)} - \eta \frac{\partial E}{\partial W}(k) \tag{۱۱-۶۸}$$

$$W^z_{(k+1)} = W^z_{(k)} - \eta \frac{\partial E}{\partial W^z}(k) \tag{۱۱-۶۹}$$

$$\mathbf{b}_{(k+1)} = \mathbf{b}_{(k)} - \eta \frac{\partial E}{\partial \mathbf{b}}(k) \tag{۱۱-۷۰}$$

یک نورون بازگشتی می تواند دارای چند پسخور از گام های گذشته مشابه شکل ۱۱-۲۳ باشد که در این صورت هر پسخور دارای وزن مختص به خود است:



شکل ۱۱-۲۳: ساختار یک نورون با چند پسخور محلی

۱۱.۸ پس انتشار در طول زمان[1]

با بررسی هر یک از روابط مشتقات زنجیره ای که برای آموزش پارامترهای ساختارهای بازگشتی تعریف کردیم، مشاهده می کنیم که این روابط دارای یک جمله مشابه رابطه ۱۱-۷۱ هستند:

$$\frac{\partial \mathbf{net}_{(k)}}{\partial W^z_{(k)}} = \mathbf{o}_{(k-1)} + W^z_{(k)} \frac{\partial \mathbf{o}_{(k-1)}}{\partial W^z_{(k)}} \tag{۱۱-۷۱}$$

برای محاسبه جمله دوم رابطه ۱۱-۷۱، رابطه ۱۱-۷۲ را داریم:

$$W^z_{(k)} \frac{\partial \mathbf{o}_{(k-1)}}{\partial W^z_{(k)}} = W^z_{(k)} \underbrace{\frac{\partial \mathbf{o}_{(k-1)}}{\partial \mathbf{net}_{(k-1)}}}_{f'(\mathbf{net}_{(k-1)})} \frac{\partial \mathbf{net}_{(k-1)}}{\partial W^z_{(k-1)}} \tag{۱۱-۷۲}$$

برای محاسبه آخرین مشتق از زنجیره در رابطه ۱۱-۷۲، رابطه ۱۱-۷۳ را داریم که همان رابطه ۱۱-۷۱ در یک گام قبل تر است:

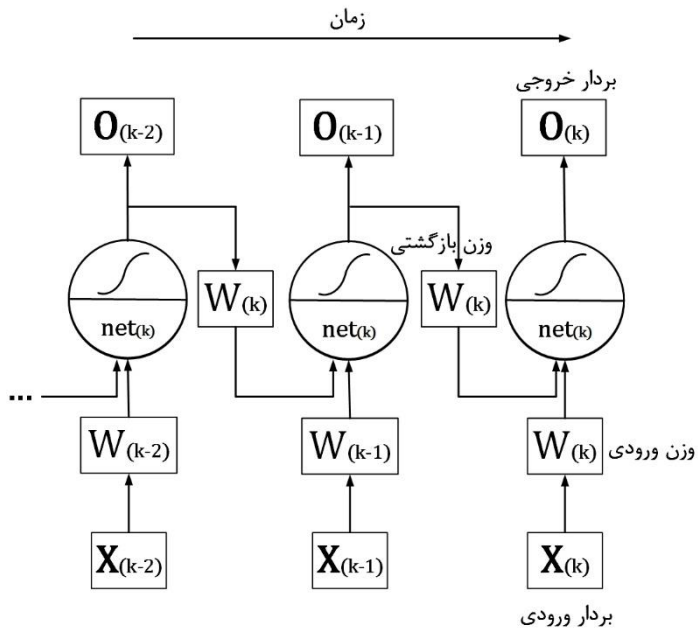
$$\frac{\partial \mathbf{net}_{(k-1)}}{\partial W^z_{(k-1)}} = \mathbf{o}_{(k-2)} + W^z_{(k-1)} \frac{\partial \mathbf{o}_{(k-2)}}{\partial W^z_{(k-1)}} \tag{۱۱-۷۳}$$

این وابستگی زمانی در مشتقات زنجیره ای تا اولین نمونه ادامه داشته و شامل جملات بسیار زیادی است، بنابراین برای جلوگیری از این پیچیدگی مقدار جمله در رابطه از اول صفر فرض کرده و از آن صرف نظر می کنیم. در این صورت رابطه ۱۱-۷۱ به صورت رابطه ۱۱-۷۴ خواهد بود:

$$\frac{\partial \mathbf{net}}{\partial W^z}(k) = \mathbf{o}_{(k-1)} \tag{۱۱-۷۴}$$

شکل ۱۱-۲۴ نمایش گسترش[2] یک نورون با پسخور محلی در طول زمان است، وابستگی زمانی بین جملات از این شکل نیز استنباط می شود؛ همان طور که در شکل ۱۱-۲۴ مشاهده می شود بردارهای در هر گام زمانی مانند بردار ورودی مستقل نبوده و مقدار آن از گام های گذشته محاسبه می شود.

[1] Back propagation through time
[2] Unfold



شکل ۲۴-۱۱: گسترش یک ساختار بازگشتی در طول زمان

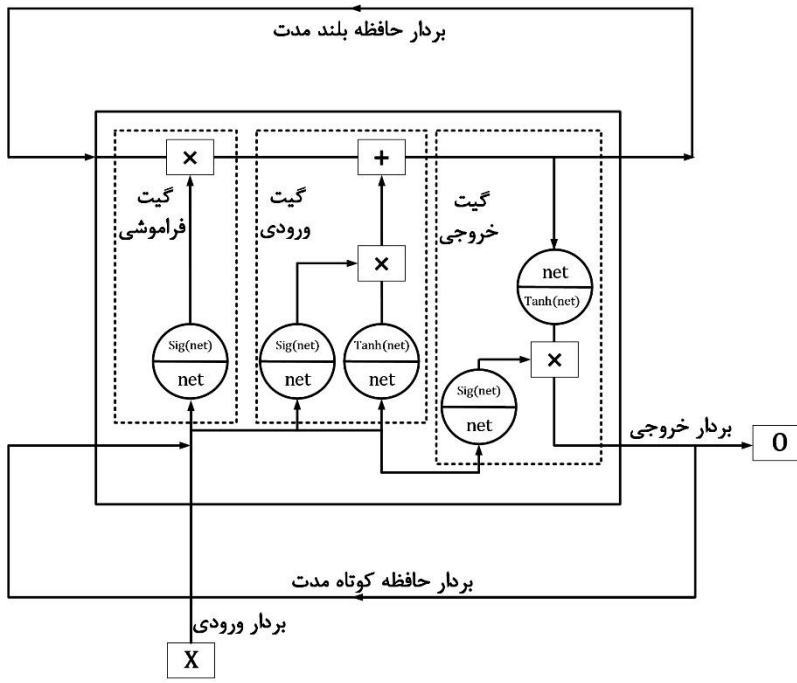
در صورتی که رابطه ۷۱-۱۱ را به صورت رابطه ۷۴-۱۱ فرض کنیم در این صورت از وابستگی زمانی که در محاسبه مشتقات زنجیره ای وجود داشت صرف نظر کرده ایم. البته با توجه به رابطه ۷۲-۱۱ حتی در صورت محاسبه جمله بازگشتی به دلیل ظاهر شدن مشتق تابع فعال ساز در روابط محاسبه آن و پدیده محوشدگی گرادیان[1]، پس از چند نمونه، مقدار این جمله بازگشتی تغییر چندانی نداشته و عملا ادامه این محاسبات سودمند نخواهد بود. این مشکل در ساختارهایی که دارای واحدهای دروازه ای[2] هستند و در ادامه به بررسی آن ها خواهیم پرداخت، کمتر نمود پیدا می کند ولی حتی در این ساختارهای دروازه ای نیز معمولا از جمله وابستگی زمانی صرف نظر می کنند.

۱۱.۹ واحد دروازه ای LSTM[3]

واحدهای دروازه ای برای رفع مشکل محوشدگی گرادیان و پیچیدگی محاسباتی که در ساختارهای بازگشتی ساده وجود داشت معرفی شده اند. این واحدها به عنوان یک نورون در یک ساختار استفاده می شوند، به بیان دیگر این واحدهای دروازه ای یک نورون پیچیده هستند. یکی از مرسوم ترین این واحدها LSTM است که دارای یک حافظه بلند مدت و یک حافظه کوتاه مدت

[1] Gradient vanishing
[2] Gated units
[3] Long short-term memory

بوده و با گیت[1] های متنوعی که دارد می تواند میزان تاثیرگذاری هر یک از بردارهای ورودی، حافظه کوتاه مدت و بلند مدت را در خروجی واحد تنظیم کند. شکل ۲۵-۱۱ ساختار یک واحد LSTM را به صورت ساده شده نشان می دهد که در ادامه هر یک از قسمت های آن را بررسی می کنیم. ساختار های متشکل از واحد های LSTM به سبب داشتن حافظه بلند مدت می توانند در شبیه سازی سری های زمانی با حافظه بلند و کوتاه مدت عملکرد مناسبی داشته باشند.

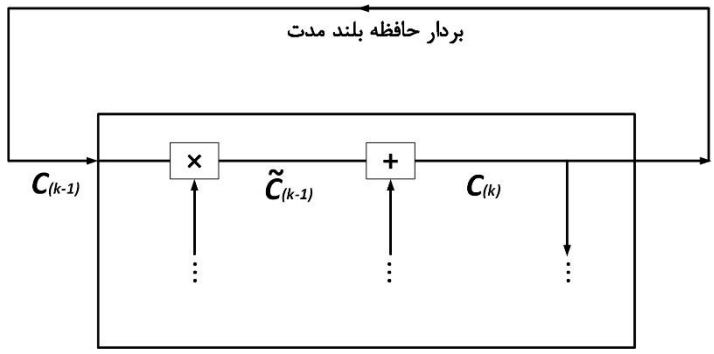


شکل ۲۵-۱۱: ساختار واحد LSTM

۱۱.۹.۱ بردار حافظه بلند مدت[2]

در ساختار واحد LSTM یک حافظه بلند مدت تعریف شده که با نماد C نشان داده می شود. عملکرد این حافظه مشابه پسخور محلی در ساختارهای بازگشتی ساده است با این تفاوت که علمیات و نگاشت پیچیده خاصی، نگاشت غیرخطی، بر آن اعمال نمی شود به همین دلیل در محاسبه مشتقات زنجیره ای از بروز پدیده محو شدگی گرادیان تا حد زیادی جلوگیری می شود. مقدار این حافظه در هر گام زمانی به وسیله گیت هایی که در ادامه به آن ها می پردازیم به روز رسانی می شود، همچنین خروجی نهائی واحد نیز براساس این مقدار به روزرسانی شده حافظه بلند مدت محاسبه می شود. شکل ۲۶-۱۱ حافظه بلند مدت یک واحد LSTM را نشان می دهد.

[1] Gate
[2] Cell state

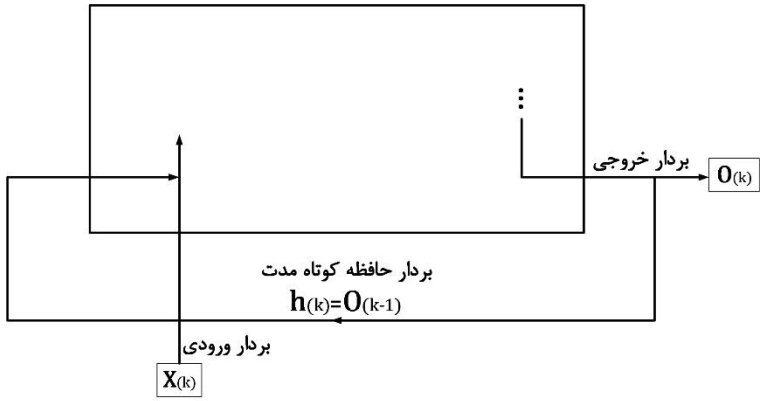


شکل ۲۶-۱۱: حافظه بلند مدت

۱۱.۹.۲ بردار ورودی و خروجی و حافظه کوتاه مدت

هر واحد LSTM مانند نورون های مرسوم دیگری که در ساختارها استفاده می شود دارای یک بردار ورودی است، این بردار ورودی در صورتی که واحد LSTM در لایه اول ساختار تعریف شده باشد همان نمونه های موجود در مجموعه دادگان آموزشی، $\boldsymbol{x}$، است که به ترتیب زمانی مرتب شده و به عنوان ورودی وارد ساختار می شود. در صورتی که واحد LSTM در لایه خروجی یا میانی یک ساختار باشد بردار ورودی آن همان بردار خروجی لایه قبل، $\boldsymbol{o^{n-1}}$، است. هر واحد LSTM دارای یک بردار خروجی، $\boldsymbol{o^n}$، نیز است که معادل خروجی نورون های مرسوم است. همان طور که در ساختار نورون با پسخور محلی خروجی نورون در گام قبل به عنوان قسمتی از ورودی گام فعلی وارد نورون می شد، در واحد های LSTM نیز خروجی گام قبل، $k-1$، به عنوان قسمتی از ورودی به صورت پسخور در گام فعلی وارد واحد می شود از این رو به خروجی واحد LSTM حافظه کوتاه مدت نیز اطلاق می شود. رابطه خروجی و حافظه کوتاه مدت ساختار در رابطه ۷۵-۱۱ تعریف شده است. در واحد های LSTM فقط خروجی یک گام قبل به عنوان پسخور، حافظه کوتاه مدت، تعریف شده و نمی توان برای آن پسخورهایی از گام هایی بیش از یک گام قبل تر از گام فعلی مشابه شکل ۲۳-۱۱ که برای ساختارهای بازگشتی ساده تعریف کردیم، در نظر بگیریم، این موضوع در مورد حافظه بلند مدت نیز صدق کرده و در هر گام فقط حافظه بلند مدت یک گام قبل تر وارد واحد می شود. در شکل بردار ۲۷-۱۱ ورودی، $\boldsymbol{x}$، و بردار حافظه کوتاه مدت $\boldsymbol{h}$، بردار خروجی $\boldsymbol{o}$، یک واحد LSTM نشان داده شده است.

$$\mathbf{h}_{(k)} = \mathbf{o}_{(k-1)} \qquad (۱۱-۷۵)$$

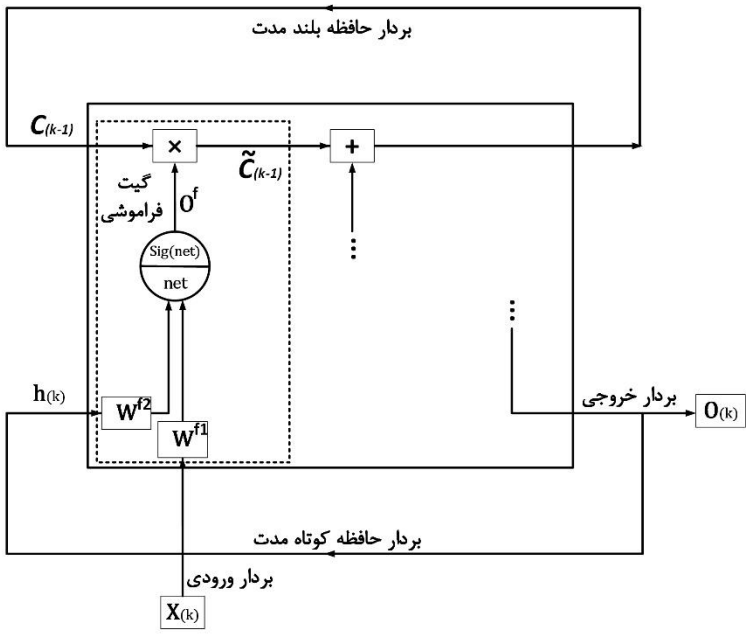


شکل ۲۷-۱۱: بردار ورودی، خروجی و حافظه کوتاه مدت

۱۱.۹.۳ گیت فراموشی[1]

گیت فراموشی مانند یک شیر کنترلی عمل می کند که در هر گام زمانی تعیین می کند چه درصدی از مقدار حافظه بلند مدت در گام قبل، $\boldsymbol{C}_{(k-1)}$، وارد گام فعلی، k، شود. ساختار این گیت یک نورون است که ورودی آن بردار ورودی واحد و بردار حافظه کوتاه مدت واحد است که به هریک از آن ها وزن هایی با ابعاد مناسب اعمال می شود. خروجی این گیت که خروجی یک تابع فعال ساز است بین صفر تا یک تعریف می شود، بدیهی است که مقادیر نزدیک به یک در خروجی درصد بیشتری از حافظه بلند مدت گام قبل را وارد واحد در گام فعلی می کند. شکل ۲۸-۱۱ گیت فراموشی و نحوه اتصال آن به حافظه بلند مدت با فرض تابع سیگموئید برای آن را نشان می دهد. مقدار خروجی گیت فراموشی با تابع فعال ساز سیگموئید مطابق شکل در حافظه بلند مدت ضرب شده و حاصل آن با نماد $\widetilde{\boldsymbol{C}}_{(k-1)}$ نشان داده می شود، ضرب اسکالر در بردار.

[1] Forget gate



شکل ۲۸-۱۱: گیت فراموشی

فرم برداری رابطه پیشرو گیت فراموشی مطابق روابط ۷۶-۱۱ تا ۷۸-۱۱ است:

$$\mathbf{net}^{f}_{(k)} = \mathbf{x}^{T}_{n\times 1(k)} \times W^{f1}_{n\times 1(k)} + \mathbf{h}^{T}_{m\times 1(k)} \times W^{f2}_{m\times 1(k)} + \mathbf{b}_{(k)} \quad (۱۱-۷۶)$$

$$\mathbf{o}^{f}_{(k)} = f(\mathbf{net}^{f}_{(k)}) = Sig(\mathbf{net}^{f}_{(k)}) \quad (۱۱-۷۷)$$

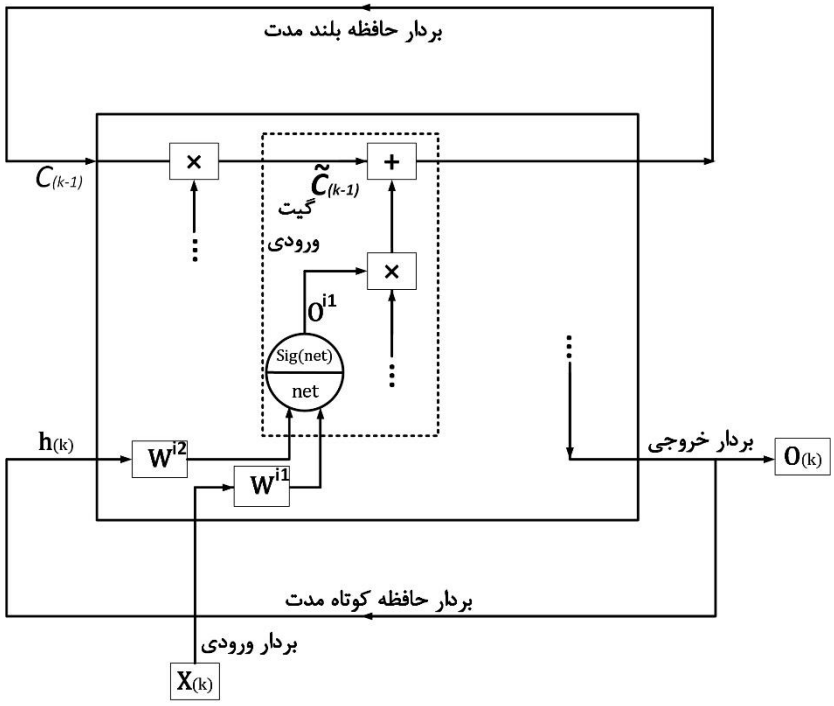
$$\tilde{\mathbf{C}}_{(k-1)} = \mathbf{o}^{f}_{(k)} \times \mathbf{C}_{(k-1)} \quad (۱۱-۷۸)$$

۱۱.۹.۴ گیت ورودی[1]

وظیفه گیت ورودی این است که قسمتی از بردار ورودی و حافظه بلند مدت را وارد حافظه بلند مدت کرده و بدین ترتیب حافظه بلند مدت را در هر گام زمانی به روزرسانی کند. گیت ورودی از دو قسمت تشکیل شده است، یک قسمت از آن مانند گیت فراموشی به عنوان یک کنترل کننده عمل می کند. این قسمت از گیت تعیین می کند که چه درصدی از خروجی قسمت دوم، فعال ساز، گیت ورودی وارد حافظه بلند مدت شود. ساختار این قسمت کنترلی مشابه گیت فراموشی است و ورودی آن نیز همان بردار ورودی و بردار حافظه کوتاه مدت هستند. این قسمت یک خروجی، اسکالر، در بازه صفر تا یک تولید می کند، خروجی تابع فعال ساز، که در خروجی بخش

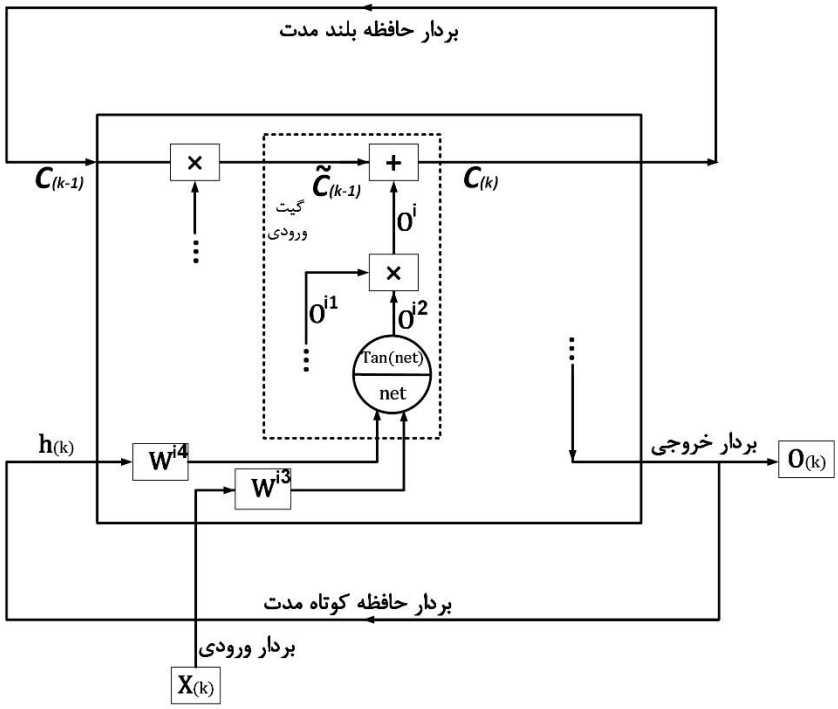
[1] Input gate

دوم گیت ورودی ضرب می شود، ضرب اسکالر در بردار. شکل ۲۹-۱۱ قسمت کنترلی گیت ورودی با تابع فعال ساز سیگموئید را نشان می دهد.



شکل ۲۹-۱۱: بخش کنترلی گیت ورودی

عملکرد قسمت دوم گیت ورودی مشابه نورون های مرسوم است، این قسمت با اعمال وزن های مناسب به بردار ورودی و بردار حافظه بلند مدت مقادیر حاصل را به عنوان ورودی یک تابع فعال ساز، تانژانت هایپربولیک، در نظر می گیرد. مقدار خروجی تابع فعال ساز در مقدار خروجی قسمت کنترلی گیت که پیش تر محاسبه شده است ضرب شده، ضرب اسکالر در بردار، و مقدار نهایی با مقدار حافظه بلند مدت که خروجی گیت فراموشی به آن اعمال شده است جمع شده و بدین ترتیب حافظه بلند مدت به روزرسانی می شود. بنابراین توضیحات می توانیم چنین برداشت کنیم که ابعاد خروجی گیت ورودی و حافظه بلند مدت با هم برابر است. شکل ۳۰-۱۱ گیت ورودی و نحوه به روزرسانی حافظه بلند مدت را نشان می دهد.



شکل ۳۰-۱۱: گیت ورودی

فرم برداری رابطه پیشرو برای قسمت کنترلی و فعال ساز گیت ورودی و به روزرسانی حافظه بلند مدت طبق شکل ۳۰-۱۱ به ترتیب طبق روابط ۷۹-۱۱ تا ۸۴-۱۱ است:

$$\mathbf{net}^{i1}_{(k)} = \mathbf{x}^{T}_{n\times 1(k)} \times W^{i1}_{n\times 1(k)} + \mathbf{h}^{T}_{m\times 1(k)} \times W^{i2}_{m\times 1(k)} + \mathbf{b}_{(k)} \qquad (۱۱-۷۹)$$

$$\mathbf{o}^{i1}_{(k)} = f(\mathbf{net}^{i1}_{(k)}) = Sig(\mathbf{net}^{i1}_{(k)}) \qquad (۱۱-۸۰)$$

$$\mathbf{net}^{i2}_{(k)} = \mathbf{x}^{T}_{n\times 1(k)} \times W^{i3}_{n\times 1(k)} + \mathbf{h}^{T}_{m\times 1(k)} \times W^{i4}_{m\times 1(k)} + \mathbf{b}_{(k)} \qquad (۱۱-۸۱)$$

$$\mathbf{o}^{i2}_{(k)} = f(\mathbf{net}^{i2}_{(k)}) = \mathrm{Tanh}(\mathbf{net}^{i2}_{(k)}) \qquad (۱۱-۸۲)$$

$$\mathbf{o}^{i}_{(k)} = \mathbf{o}^{i1}_{(k)} \times \mathbf{o}^{i2}_{(k)} \qquad (۱۱-۸۳)$$

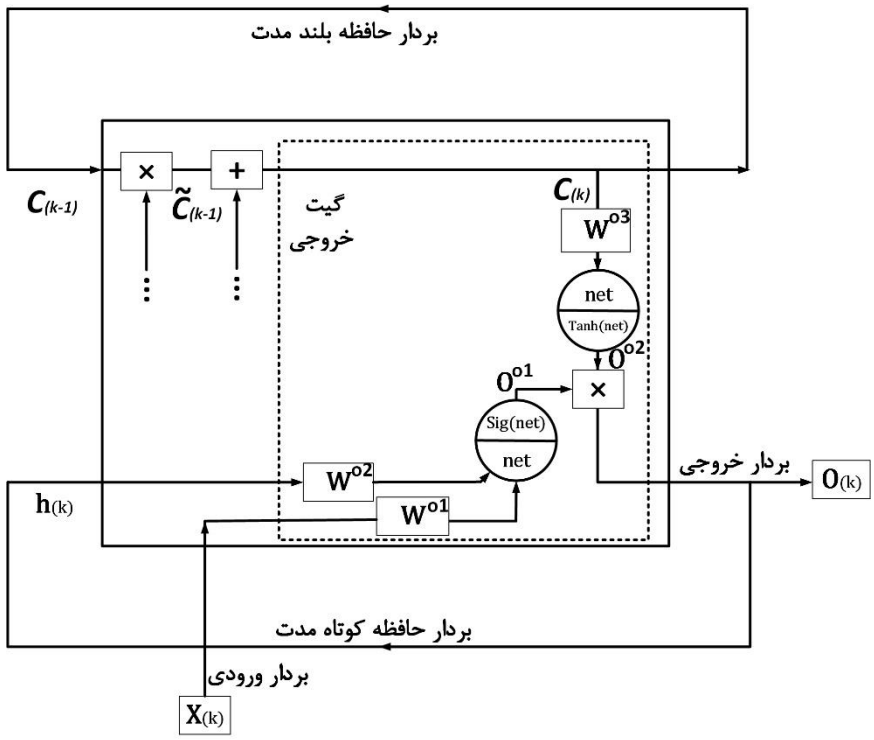
$$\mathbf{C}_{(k)} = \mathbf{o}^{i}_{(k)} \times \tilde{\mathbf{C}}_{(k-1)} \qquad (۱۱-۸۴)$$

۱۱.۹.۵ گیت خروجی[1]

وظیفه گیت خروجی ایجاد خروجی نهائی واحد، حافظه کوتاه مدت، است. ساختار این گیت نیز

[1] Output gate

مانند گیت ورودی دارای یک قسمت کنترلی و یک قسمت فعال ساز است که ورودی آن بردار ورودی و حافظه کوتاه مدت است، در حالی که ورودی قسمت فعال ساز حافظه بلند مدت به روزرسانی شده واحد است. شکل ۳۱-۱۱ نمایش گیت خروجی است.



شکل ۳۱-۱۱: گیت خروجی

فرم برداری روابط پیشرو برای قسمت کنترلی و فعال ساز گیت خروجی به ترتیب طبق روابط ۸۵-۱۱ تا ۸۹-۱۱ است:

$$\mathbf{net}^{o1}_{(k)} = \mathbf{x}^{T}_{n\times 1(k)} \times W^{o1}_{n\times 1(k)} + \mathbf{h}^{T}_{m\times 1(k)} \times W^{o2}_{m\times 1(k)} + \mathbf{b}_{(k)} \quad (۱۱-۸۵)$$

$$\mathbf{o}^{o1}_{(k)} = f(\mathbf{net}^{o1}_{(k)}) = Sig(\mathbf{net}^{o1}_{(k)}) \quad (۱۱-۸۶)$$

$$\mathbf{net}^{o2}_{(k)} = \mathbf{C}^{T}_{n\times 1(k)} \times W^{o3}_{n\times 1} \quad (۱۱-۸۷)$$

$$\mathbf{o}^{o2}_{(k)} = f(\mathbf{net}^{o2}_{(k)}) = \mathrm{Tanh}(\mathbf{net}^{o2}_{(k)}) \quad (۱۱-۸۸)$$

$$\mathbf{o}_{(k)} = \mathbf{o}^{o1}_{(k)} \times \mathbf{o}^{o2}_{(k)} \quad (۱۱-۸۹)$$

۱۱.۹.۶ آموزش یک واحد LSTM با الگوریتم پس انتشار خطا به روش گرادیان نزولی

به منظور سهولت در محاسبات برای تعریف مشتقات زنجیره برای یک واحد LSTM، فرض می کنیم ساختار مدنظر فقط از یک لایه که دارای یک واحد LSTM است تشکیل شده است. بنابراین طبق این فرض خروجی واحد LSTM همان خروجی نهائی ساختار است که با مقایسه آن با مقدار مطلوب خطای ساختار تعریف می شود. همچنین تابع هزینه ساختار را میانگین مربعات خطا، E، در نظر می گیریم. با این مفروضات مشتقات زنجیره ای هر یک از وزن های واحد را به صورت زیر تعریف می کنیم:

$$\frac{\partial E}{\partial W^{o3}}=\frac{\partial E}{\partial \mathbf{e}}\frac{\partial \mathbf{e}}{\partial \mathbf{o}}\frac{\partial \mathbf{o}}{\partial \mathbf{o}^{o2}}\underbrace{\frac{\partial \mathbf{o}^{o2}}{\partial \mathbf{net}^{o2}}}_{f'_{(\mathbf{net}^{o2})}}\underbrace{\frac{\partial \mathbf{net}^{o2}}{\partial W^{o3}}}_{\mathbf{C}_{(k)}}=-e\underbrace{f'_{(\mathbf{net}^{o2}_{(k)})}}_{e}\underbrace{\mathbf{o}^{o1}}_{-1}\underbrace{\mathbf{C}_{(k)}}_{\mathbf{o}^{o1}} \tag{۱۱-۹۰}$$

$$\frac{\partial E}{\partial W^{o2}}=\frac{\partial E}{\partial \mathbf{e}}\frac{\partial \mathbf{e}}{\partial \mathbf{o}}\frac{\partial \mathbf{o}}{\partial \mathbf{o}^{o1}}\frac{\partial \mathbf{o}^{o1}}{\partial \mathbf{net}^{o1}}\underbrace{\frac{\partial \mathbf{net}^{o1}}{\partial W^{o2}}}_{\mathbf{h}_{(k)}} \tag{۱۱-۹۱}$$

$$\frac{\partial E}{\partial W^{o1}}=\frac{\partial E}{\partial \mathbf{e}}\frac{\partial \mathbf{e}}{\partial \mathbf{o}}\frac{\partial \mathbf{o}}{\partial \mathbf{o}^{o1}}\frac{\partial \mathbf{o}^{o1}}{\partial \mathbf{net}^{o1}}\underbrace{\frac{\partial \mathbf{net}^{o1}}{\partial W^{o1}}}_{\mathbf{x}_{(k)}} \tag{۱۱-۹۲}$$

$$\frac{\partial E}{\partial W^{i4}}=\frac{\partial E}{\partial \mathbf{e}}\frac{\partial \mathbf{e}}{\partial \mathbf{o}}\frac{\partial \mathbf{o}}{\partial \mathbf{o}^{o2}}\frac{\partial \mathbf{o}^{o2}}{\partial \mathbf{net}^{o2}}\underbrace{\frac{\partial \mathbf{net}^{o2}}{\partial \mathbf{C}_{(k)}}}_{W^{o3}}\frac{\partial \mathbf{C}_{(k)}}{\partial \mathbf{o}^{i}}\frac{\partial \mathbf{o}^{i}}{\partial \mathbf{o}^{i2}}\frac{\partial \mathbf{o}^{i2}}{\partial \mathbf{net}^{i2}}\underbrace{\frac{\partial \mathbf{net}^{i2}}{\partial W^{i4}}}_{\mathbf{h}_{(k)}} \tag{۱۱-۹۳}$$

$$\frac{\partial E}{\partial W^{i3}}=\frac{\partial E}{\partial \mathbf{e}}\frac{\partial \mathbf{e}}{\partial \mathbf{o}}\frac{\partial \mathbf{o}}{\partial \mathbf{o}^{o2}}\frac{\partial \mathbf{o}^{o2}}{\partial \mathbf{net}^{o2}}\underbrace{\frac{\partial \mathbf{net}^{o2}}{\partial \mathbf{C}_{(k)}}}_{W^{o3}}\frac{\partial \mathbf{C}_{(k)}}{\partial \mathbf{o}^{i}}\frac{\partial \mathbf{o}^{i}}{\partial \mathbf{o}^{i2}}\frac{\partial \mathbf{o}^{i2}}{\partial \mathbf{net}^{i2}}\underbrace{\frac{\partial \mathbf{net}^{i2}}{\partial W^{i3}}}_{\mathbf{x}_{(k)}} \tag{۱۱-۹۴}$$

$$\frac{\partial E}{\partial W^{i2}}=\frac{\partial E}{\partial \mathbf{e}}\frac{\partial \mathbf{e}}{\partial \mathbf{o}}\frac{\partial \mathbf{o}}{\partial \mathbf{o}^{o2}}\frac{\partial \mathbf{o}^{o2}}{\partial \mathbf{net}^{o2}}\underbrace{\frac{\partial \mathbf{net}^{o2}}{\partial \mathbf{C}_{(k)}}}_{W^{o3}}\frac{\partial \mathbf{C}_{(k)}}{\partial \mathbf{o}^{i}}\frac{\partial \mathbf{o}^{i}}{\partial \mathbf{o}^{i1}}\frac{\partial \mathbf{o}^{i1}}{\partial \mathbf{net}^{i1}}\underbrace{\frac{\partial \mathbf{net}^{i1}}{\partial W^{i2}}}_{\mathbf{h}_{(k)}} \tag{۱۱-۹۵}$$

$$\frac{\partial E}{\partial W^{i1}}=\frac{\partial E}{\partial \mathbf{e}}\frac{\partial \mathbf{e}}{\partial \mathbf{o}}\underbrace{\frac{\partial \mathbf{o}}{\partial \mathbf{o}^{o2}}}_{\mathbf{o}^{o1}}\frac{\partial \mathbf{o}^{o2}}{\partial \mathbf{net}^{o2}}\underbrace{\frac{\partial \mathbf{net}^{o2}}{\partial \mathbf{C}_{(k)}}}_{W^{o3}}\underbrace{\frac{\partial \mathbf{C}_{(k)}}{\partial \mathbf{o}^{i}}}_{1}\underbrace{\frac{\partial \mathbf{o}^{i}}{\partial \mathbf{o}^{i1}}}_{\mathbf{o}^{i2}}\frac{\partial \mathbf{o}^{i1}}{\partial \mathbf{net}^{i1}}\underbrace{\frac{\partial \mathbf{net}^{i1}}{\partial W^{i1}}}_{\mathbf{x}_{(k)}} \tag{۱۱-۹۶}$$

$$\frac{\partial E}{\partial W^{f2}}=\frac{\partial E}{\partial \mathbf{e}}\frac{\partial \mathbf{e}}{\partial \mathbf{o}}\underbrace{\frac{\partial \mathbf{o}}{\partial \mathbf{o}^{o2}}}_{\mathbf{o}^{o1}}\frac{\partial \mathbf{o}^{o2}}{\partial \mathbf{net}^{o2}}\underbrace{\frac{\partial \mathbf{net}^{o2}}{\partial \mathbf{C}_{(k)}}}_{W^{o3}}\underbrace{\frac{\partial \mathbf{C}_{(k)}}{\partial \tilde{\mathbf{C}}_{(k)}}}_{1}\underbrace{\frac{\partial \tilde{\mathbf{C}}_{(k)}}{\partial \mathbf{o}^{f}}}_{\mathbf{C}_{(k-1)}}\frac{\partial \mathbf{o}^{f}}{\partial \mathbf{net}^{f}}\underbrace{\frac{\partial \mathbf{net}^{f}}{\partial W^{f2}}}_{\mathbf{h}_{(k)}} \tag{۱۱-۹۷}$$

$$\frac{\partial E}{\partial W^{f1}}=\frac{\partial E}{\partial \mathbf{e}}\frac{\partial \mathbf{e}}{\partial \mathbf{o}}\underbrace{\frac{\partial \mathbf{o}}{\partial \mathbf{o}^{o2}}}_{\mathbf{o}^{o1}}\frac{\partial \mathbf{o}^{o2}}{\partial \mathbf{net}^{o2}}\underbrace{\frac{\partial \mathbf{net}^{o2}}{\partial \mathbf{C}_{(k)}}}_{W^{o3}}\underbrace{\frac{\partial \mathbf{C}_{(k)}}{\partial \tilde{\mathbf{C}}_{(k)}}}_{1}\underbrace{\frac{\partial \tilde{\mathbf{C}}_{(k)}}{\partial \mathbf{o}^{f}}}_{\mathbf{C}_{(k-1)}}\frac{\partial \mathbf{o}^{f}}{\partial \mathbf{net}^{f}}\underbrace{\frac{\partial \mathbf{net}^{f}}{\partial W^{f1}}}_{\mathbf{x}_{(k)}} \tag{۱۱-۹۸}$$

طبق مشتقات زنجیره ای تعریف شده در روابط ۹۰-۱۱ تا ۹۸-۱۱، مقدار هر یک از پارامترهای واحد LSTM در هر گام به صورت روابط زیر محاسبه می شود:

$$W^{o3}_{(k+1)}=W^{o3}_{(k)}-\eta\frac{\partial E}{\partial W^{o3}}(k) \tag{۱۱-۹۹}$$

$$W^{o2}_{(k+1)}=W^{o2}_{(k)}-\eta\frac{\partial E}{\partial W^{o2}}(k) \tag{۱۱-۱۰۰}$$

$$W^{o1}_{(k+1)}=W^{o1}_{(k)}-\eta\frac{\partial E}{\partial W^{o1}}(k) \tag{۱۱-۱۰۱}$$

$$W^{i4}_{(k+1)}=W^{i4}_{(k)}-\eta\frac{\partial E}{\partial W^{i4}}(k) \tag{۱۱-۱۰۲}$$

$$W^{i3}_{(k+1)}=W^{i3}_{(k)}-\eta\frac{\partial E}{\partial W^{i3}}(k) \tag{۱۱-۱۰۳}$$

$$W^{i2}_{(k+1)}=W^{i2}_{(k)}-\eta\frac{\partial E}{\partial W^{i2}}(k) \tag{۱۱-۱۰۴}$$

$$W^{i1}_{(k+1)}=W^{i1}_{(k)}-\eta\frac{\partial E}{\partial W^{i1}}(k) \tag{۱۱-۱۰۵}$$

$$W^{f2}_{(k+1)}=W^{f2}_{(k)}-\eta\frac{\partial E}{\partial W^{f2}}(k) \tag{۱۱-۱۰۶}$$

$$W_{(k+1)}^{f1} = W_{(k)}^{f1} - \eta \frac{\partial E}{\partial W^{f1}}(k)$$

(۱۱-۱۰۷)

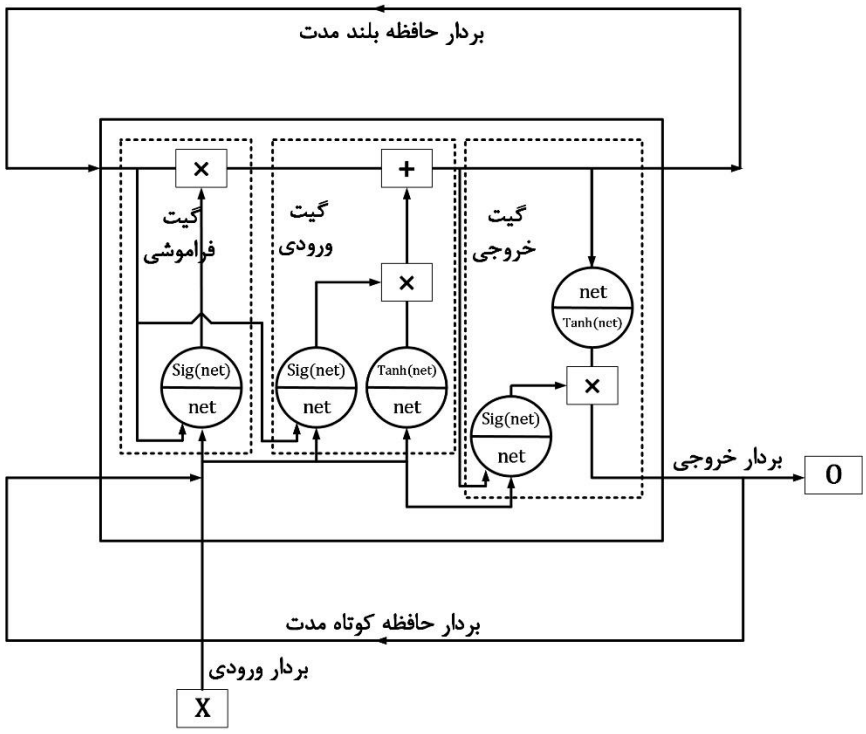
۱۱.۹.۷ انواع مختلف واحدهای LSTM

ساختاری که بررسی کردیم، ساختار کلی برای یک واحد LSTM بود. با توجه به ملزومات هر مسئله و دادگان آموزشی مورد استفاده گونه های مختلفی برای واحد LSTM تعریف شده است که با اعمال تغییراتی به ساختار واحد LSTM اصلی تعریف شده اند. در ادامه به بررسی چند نمونه از این ساختارها پرداخته و ویژگی های هر کدام را بررسی می کنیم.

۱۱.۹.۷.۱ واحد LSTM روزنه ای[1]

ساختار این نوع از واحدهای LSTM مطابق شکل ۱۱-۳۲ است. همان طور که در شکل ۱۱-۳۲ مشخص شده است تفاوت این ساختار با ساختار اصلی واحد LSTM در این است که در واحد LSTM اصلی، ورودی بخش‌های کنترلی هر یک از گیت ها بردار ورودی واحد و بردار حافظه کوتاه مدت واحد بود در حالی که در ساختار روزنه ای، ورودی بخش های کنترلی علاوه بر بردار ورودی و بردار حافظه کوتاه مدت شامل حافظه بلند مدت نیز می شود، همان طور که دو بردار ورودی و حافظه کوتاه مدت کنار هم قرار گرفته[2] و به عنوان بردار ورودی بخش کنترلی گیت ها استفاده می شوند در ساختار روزنه ای این بردار ورودی بخش های کنترلی از کنار هم قرار گرفتن سه بردار ورودی، حافظه کوتاه مدت و حافظه بلند مدت تشکیل می شود. در ساختار روزنه ای بر روی بردار حافظه بلند مدت در هر بخش کنترلی همانند حافظه کوتاه مدت و بردار ورودی وزن های مناسبی اعمال شده می شود. برای قسمت کنترلی دو گیت فراموشی و ورودی، از حافظه بلند مدت گام قبل $\boldsymbol{C}_{(k-1)}$، پیش از به روز رسانی، و برای بخش کنترلی گیت خروجی از حافظه بلند مدت گام فعلی $\boldsymbol{C}_{(k)}$، به روزرسانی شده، استفاده می کنیم. بقیه قسمت های ساختار مشابه با ساختار واحد LSTM اصلی است. با این تفاسیر در ساختار LSTM روزنه ای روابط پیشرو قسمت های کنترلی گیت های فراموشی، ورودی و خروجی که دارای تابع فعال ساز بین صفر و یک، سیگموئید، هستند را به ترتیب به صورت روابط ۱۱-۱۰۸ تا ۱۱-۱۱۳ تعریف می کنیم:

[1] Peephole LSTM
[2] Concatenate



شکل ۳۲-۱۱: ساختار LSTM روزنه ای

$$\mathbf{net}^{f}_{(k)} = \mathbf{x}^{T}_{n\times 1(k)} \times W^{f1}_{n\times 1(k)} + \mathbf{h}^{T}_{m\times 1(k)} \times W^{f2}_{m\times 1(k)} + \mathbf{C}^{T}_{l\times 1(k-1)} \times W^{fc}_{l\times 1(k)} + \mathbf{b}_{(k)} \quad (۱۱-۱۰۸)$$

$$\mathbf{o}^{f}_{(k)} = f(\mathbf{net}^{f}_{(k)}) = Sig(\mathbf{net}^{f}_{(k)}) \quad (۱۱-۱۰۹)$$

$$\mathbf{net}^{i1}_{(k)} = \mathbf{x}^{T}_{n\times 1(k)} \times W^{i1}_{n\times 1(k)} + \mathbf{h}^{T}_{m\times 1(k)} \times W^{i2}_{m\times 1(k)} + \mathbf{C}^{T}_{l\times 1(k-1)} \times W^{ic}_{l\times 1(k)} + \mathbf{b}_{(k)} \quad (۱۱-۱۱۰)$$

$$\mathbf{o}^{i1}_{(k)} = f(\mathbf{net}^{i1}_{(k)}) = Sig(\mathbf{net}^{i1}_{(k)}) \quad (۱۱-۱۱۱)$$

$$\mathbf{net}^{o1}_{(k)} = \mathbf{x}^{T}_{n\times 1(k)} \times W^{o1}_{n\times 1(k)} + \mathbf{h}^{T}_{m\times 1(k)} \times W^{o2}_{m\times 1(k)} + \mathbf{C}^{T}_{l\times 1(k)} \times W^{oc}_{l\times 1(k)} + \mathbf{b}_{(k)} \quad (۱۱-۱۱۲)$$

$$\mathbf{o}^{o1}_{(k)} = f(\mathbf{net}^{o1}_{(k)}) = Sig(\mathbf{net}^{o1}_{(k)}) \quad (۱۱-۱۱۳)$$

۱۱.۹.۷.۲ واحد LSTM چندگانه[1]

واحد LSTM چندگانه برای افزایش ظرفیت واحد بدون افزودن پارامترهای جدید معرفی شده اند. تفاوت این ساختار با ساختار استاندارد LSTM در این است که در ساختار استاندارد برای محاسبه ***net*** قسمت هایی که ورودی آن ها بردار ورودی و بردار حافظه کوتاه مدت بود از رابطه ای ۱۱-۱۱۴ مشابه رابطه استفاده می کردیم در حالی که این رابطه برای ساختار چندگانه به صورت

[1] Multiplicative LSTM

رابطه ۱۱۵-۱۱ تعریف می شود که در آن نماد $\odot$ بیانگر ضرب درایه به درایه (ضرب هادامارد) و g یک مقدار اسکالر دلخواه است که ابعاد وزن های مشخص شده را تعریف می کند.

$$\mathbf{net}_{(k)} = \mathbf{x}^T_{n\times 1(k)} \times W^1_{n\times 1(k)} + \mathbf{h}^T_{m\times 1(k)} \times W^2_{m\times 1(k)} + \mathbf{b}_{(k)} \quad (۱۱-۱۱۴)$$

$$\mathbf{net}_{(k)} = \mathbf{x}^T_{n\times 1(k)} \times W^1_{n\times g(k)} \odot \mathbf{h}^T_{m\times 1(k)} \times W^2_{m\times g(k)} + \mathbf{b}_{(k)} \quad (۱۱-۱۱۵)$$

بدین ترتیب روند پیشرو در یک ساختار واحد LSTM چندگانه به صورت روابط ۱۱۶-۱۱ تا ۱۲۸-۱۱ خواهد بود:

$$\mathbf{net}^f_{(k)} = \mathbf{x}^T_{n\times 1(k)} \times W^{f1}_{n\times g(k)} \odot \mathbf{h}^T_{m\times 1(k)} \times W^{f2}_{m\times g(k)} + \mathbf{b}_{(k)} \quad (۱۱-۱۱۶)$$

$$\mathbf{o}^f_{(k)} = f(\mathbf{net}^f_{(k)}) = Sig(\mathbf{net}^f_{(k)}) \quad (۱۱-۱۱۷)$$

$$\mathbf{net}^{i1}_{(k)} = \mathbf{x}^T_{n\times 1(k)} \times W^{i1}_{n\times g(k)} \odot \mathbf{h}^T_{m\times 1(k)} \times W^{i2}_{m\times g(k)} + \mathbf{b}_{(k)} \quad (۱۱-۱۱۸)$$

$$\mathbf{o}^{i1}_{(k)} = f(\mathbf{net}^{i1}_{(k)}) = Sig(\mathbf{net}^{i1}_{(k)}) \quad (۱۱-۱۱۹)$$

$$\mathbf{net}^{i2}_{(k)} = \mathbf{x}^T_{n\times 1(k)} \times W^{i3}_{n\times g} \odot \mathbf{h}^T_{m\times 1(k)} \times W^{i4}_{m\times g} + \mathbf{b} \quad (۱۱-۱۲۰)$$

$$\mathbf{o}^{i2}_{(k)} = f(\mathbf{net}^{i2}_{(k)}) = \mathrm{Tanh}(\mathbf{net}^{i2}_{(k)}) \quad (۱۱-۱۲۱)$$

$$\mathbf{o}^{i}_{(k)} = \mathbf{o}^{i1}_{(k)} \times \mathbf{o}^{i2}_{(k)} \quad (۱۱-۱۲۲)$$

$$\mathbf{C}_{(k)} = \mathbf{o}^{i}_{(k)} \times \tilde{\mathbf{C}}_{(k-1)} \quad (۱۱-۱۲۳)$$

$$\mathbf{net}^{o1}_{(k)} = \mathbf{x}^T_{n\times 1(k)} \times W^{o1}_{n\times g} \odot \mathbf{h}^T_{m\times 1(k)} \times W^{o2}_{m\times g} + \mathbf{b} \quad (۱۱-۱۲۴)$$

$$\mathbf{o}^{o1}_{(k)} = f(\mathbf{net}^{o1}_{(k)}) = Sig(\mathbf{net}^{o1}_{(k)}) \quad (۱۱-۱۲۵)$$

$$\mathbf{net}^{o2}_{(k)} = \mathbf{C}^T_{n\times 1(k)} \times W^{o3}_{n\times 1} \quad (۱۱-۱۲۶)$$

$$\mathbf{o}^{o2}_{(k)} = f(\mathbf{net}^{o2}_{(k)}) = \mathrm{Tanh}(\mathbf{net}^{o2}_{(k)}) \quad (۱۱-۱۲۷)$$

$$\mathbf{o}_{(k)} = \mathbf{o}^{o1}_{(k)} \times \mathbf{o}^{o2}_{(k)} \quad (۱۱-۱۲۸)$$

در ساختار استاندارد، انتهای روابط مشتقات زنجیره ای وزن های متناظر با هر یک از بردارهای ورودی و حافظه کوتاه مدت، مانند رابطه ۹۷-۱۱ و ۹۸-۱۱، جملاتی مشابه رابطه ۱۲۹-۱۱ و ۱۳۰-۱۱ داشتند در حالی که این جملات برای ساختار چندگانه به صورت روابط ۱۳۱-۱۱ و

۱۱-۱۳۲ است. با مقایسه مشتقات زنجیره ای دو ساختار چنین برداشت می کنیم که با ضرب شدن وزن و بردار در جمله انتهایی مشـــتقات زنجیره ای واحدهای چندگانه، این مشـــتقات مقادیر بزرگ تری داشته و بر این اساس واحدهای چندگانه نسبت به واحد استاندارد در برابر پدیده محو شدگی گرادیان ها مقاوم تر هستند.

$$\cdots \frac{\partial \mathbf{o}}{\partial \mathbf{net}} \frac{\partial \mathbf{net}}{\partial W^2} \mathbf{h}_{(k)} \tag{۱۱-۱۲۹}$$

$$\cdots \frac{\partial \mathbf{o}}{\partial \mathbf{net}} \frac{\partial \mathbf{net}}{\partial W^1} \mathbf{x}_{(k)} \tag{۱۱-۱۳۰}$$

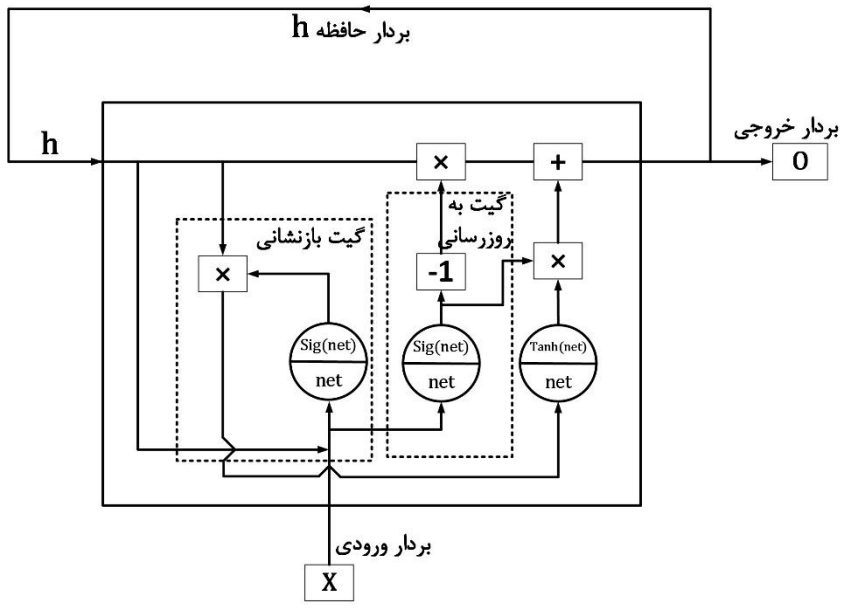
$$\cdots \frac{\partial \mathbf{o}}{\partial \mathbf{net}} \frac{\partial \mathbf{net}}{\partial W^2} \mathbf{x}_{(k)} W^1 \mathbf{h}_{(k)} \tag{۱۱-۱۳۱}$$

$$\cdots \frac{\partial \mathbf{o}}{\partial \mathbf{net}} \frac{\partial \mathbf{net}}{\partial W^1} \mathbf{h}_{(k)} W^2 \mathbf{x}_{(k)} \tag{۱۱-۱۳۲}$$

۱۱.۱۰ واحد دروازه ای GRU[1]

یکی دیگر از ساختارهای مرسوم دروازه ای که به طور گسترده در ساختار شبکه های ژرف به کار برده می شـود، واحد های GRU هســتند. همانند واحد های LSTM، ســاختار این واحد نیز از چند گیت تشکیل شده که هر یک از آن ها با کنترل مقادیر بردار ورودی و حافظه، در هر گام زمانی خروجی مناســب را تولید می کنند. واحد های GRU در مقایســه با واحدهای LSTM پیچیدگی کمتری داشته و نیازمند هزنیه محاسباتی کمتری هستند. بسته به رفتار دادگان ساختار متشکل از هر یک از گونه های واحد های LSTM و یا واحد های GRU می تولند عملکرد مناســبی داشــته باشد؛ لذا سادگی واحدهای GRU در مقایسه با LSTM لزوما به معنی افت عملکرد ساختار نیست. در ادامه هر یک از بخش های یک واحد GRU را بررسی می کنیم. شکل ۱۱-۳۳ ساختار یک واحد GRU را نشان می دهد.

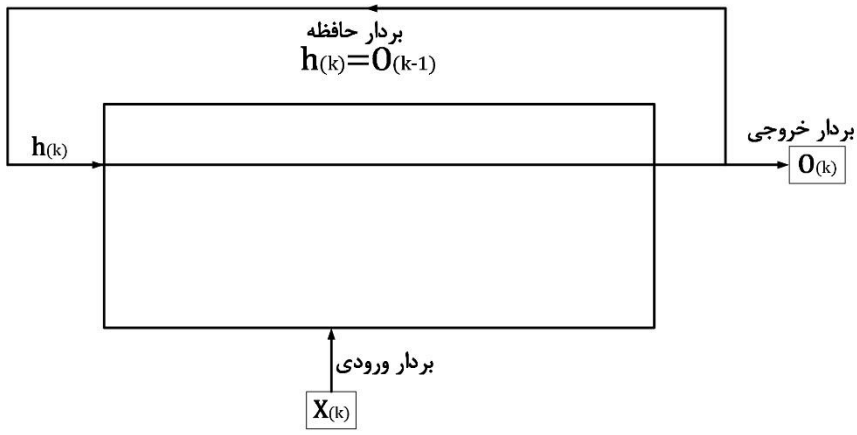
[1] Gated recurrent unit



شکل ۳۳-۱۱: ساختار واحد GRU

۱۱.۱۰.۱ بردار ورودی، خروجی و حافظه

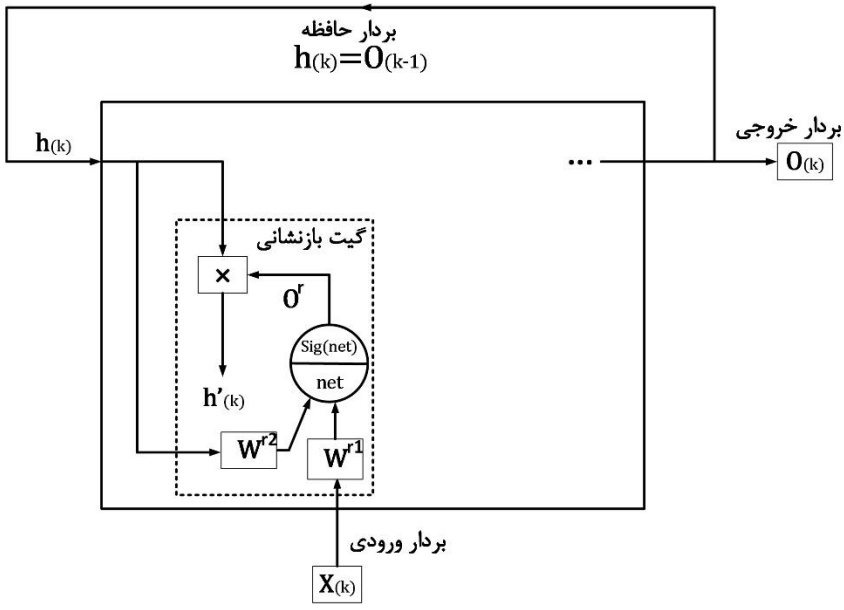
در ساختار واحدهای GRU برخلاف ساختار LSTM، تنها از یک حافظه استفاده شده است. حافظه واحد در هر گام زمانی به روزرسانی شده و مقادیر جدیدی را به خود می گیرد. بردار حافظه در یک واحد GRU همان بردار خروجی واحد است. همانند ساختار LSTM، برای این ساختار نیز در هر گام زمانی یک بردار ورودی تعریف می شود. شکل ۳۴-۱۱ ساختار حافظه و بردار ورودی را برای یک واحد GRU نشان می دهد.



شکل ۳۴-۱۱: بردار ورودی، خروجی و حافظه

۱۱.۱۰.۲ گیت بازنشانی[1]

این گیت در ساختار واحد GRU عملکردی مشابه گیت فراموشی در ساختار واحدهای LSTM را دارد. گیت بازنشانی از یک بخش کنترلی تشکیل شده است که ورودی آن بردار ورودی و بردار حافظه گام پیشین هستند. در گیت بازنشانی به بردار ورودی و حافظه یک تابع فعال سازی که بازه خروجی آن بین ۰ تا ۱ باشد، سیگموئید، اعمال شده و در نهایت خروجی آن به عنوان مقداری که میزان ورودی بردار حافظه گام قبل به گام فعلی را تعیین می کند در بردار حافظه گام پیشین ضرب می شود. بدین ترتیب برخلاف گیت فراموشی ساختار واحدهای LSTM که در آن بردار ورودی و حافظه کوتاه مدت میزان ورود بردار حافظه بلند مدت به گام فعلی را تعیین می کردند در این ساختار خود بردار حافظه نیز در تعیین این میزان نقش داشته و از این نظر مشابه ساختار واحد LSTM روزنه ای است. شکل ۳۵-۱۱ نحوه عملکرد گیت بازنشانی در ساختار یک واحد GRU را نشان می دهد.



شکل ۳۵-۱۱: گیت بازنشانی

روابط پیشرو برای گیت بازنشانی یک واحد GRU به صورت روابط ۱۳۳-۱۱ تا ۱۳۵-۱۱ است:

$$\mathbf{net}^{r}_{(k)} = \mathbf{x}^{T}_{n\times 1(k)} \times W^{r1}_{n\times 1(k)} + \mathbf{h}^{T}_{m\times 1(k)} \times W^{r2}_{m\times 1(k)} + \mathbf{b}_{(k)} \qquad (۱۱-۱۳۳)$$

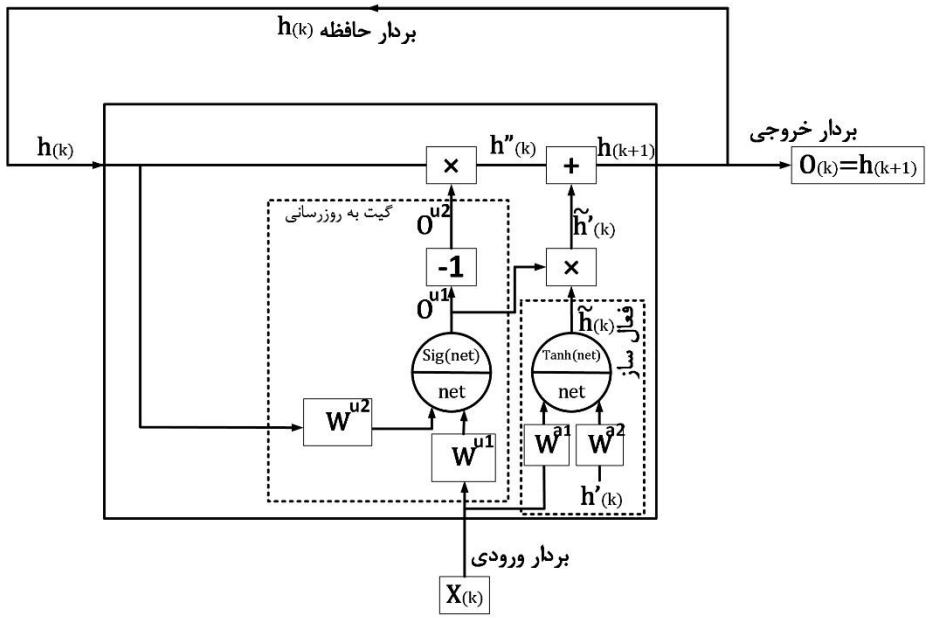
$$\mathbf{o}^{r}_{(k)} = f(\mathbf{net}^{r}_{(k)}) = Sig(\mathbf{net}^{r}_{(k)}) \qquad (۱۱-۱۳۴)$$

[1] Reset gate

$$\mathbf{h}'_{(k)} = \mathbf{o}^{r}_{(k)} \times \mathbf{h}_{(k)} \quad (۱۱-۱۳۵)$$

۱۱.۱۰.۳ گیت به روزرسانی[1]

در ساختار واحدهای GRU، وظایف کنترلی دو گیت ورودی و خروجی که در ساختار واحد LSTM تعریف می شوند با هم ادغام شده و به صورت گیت به روزرسانی تعریف شده است. گیت به روزرسانی دارای یک بخش کنترلی است که همزمان میزان ورود حافظه گام قبل و خروجی قسمت فعال ساز واحد را کنترل می کند. ورودی این قسمت کنترلی بردار ورودی و حافظه گام قبل است. قسمت فعال ساز واحد یک تابع فعال ساز، تانژانت هایپربولیک، است که ورودی آن بردار ورودی و بردار حافظه گام قبل که به آن خروجی گیت بازنشانی اعمال شده است. خروجی بخش فعال ساز واحد پس از اعمال خروجی بخش کنترلی گیت به روزرسانی با بردار حافظه گام قبل که قرینه خروجی بخش کنترلی گیت به روزرسانی به آن اعمال شده جمع شده و بدین ترتیب بردار حافظه گام فعلی که همان خروجی واحد است تشکیل می شود. شکل ۳۶-۱۱ ساختار گیت به روزرسانی و قسمت فعال ساز یک واحد GRU را نشان می دهد.



شکل ۳۶-۱۱: گیت به روز رسانی و قسمت فعال ساز

روابط پیشرو برای گیت به روزرسانی و قسمت فعال ساز یک واحد GRU به صورت روابط ۱۳۶-۱۱ تا ۱۴۳-۱۱ است:

[1] Update gate

$$\mathbf{net}^{u}_{(k)} = \mathbf{x}^{T}_{n\times 1(k)} \times W^{u1}_{n\times 1(k)} + \mathbf{h}^{T}_{m\times 1(k)} \times W^{u2}_{m\times 1(k)} + \mathbf{b}_{(k)} \quad (۱۱-۱۳۶)$$

$$\mathbf{o}^{u1}_{(k)} = f(\mathbf{net}^{u}_{(k)}) = Sig(\mathbf{net}^{u}_{(k)}) \quad (۱۱-۱۳۷)$$

$$\mathbf{o}^{u2}_{(k)} = -1 \times \mathbf{o}^{u1}_{(k)} \quad (۱۱-۱۳۸)$$

$$\mathbf{h}''_{(k)} = \mathbf{o}^{u2}_{(k)} \times \mathbf{h}_{(k)} \quad (۱۱-۱۳۹)$$

$$\mathbf{net}^{a}_{(k)} = \mathbf{x}^{T}_{n\times 1(k)} \times W^{a1}_{n\times 1(k)} + \mathbf{h}'^{T}_{m\times 1(k)} \times W^{a2}_{m\times 1(k)} + \mathbf{b}_{(k)} \quad (۱۱-۱۴۰)$$

$$\tilde{\mathbf{h}}_{(k)} = f(\mathbf{net}^{a}_{(k)}) = \mathrm{Tanh}(\mathbf{net}^{a}_{(k)}) \quad (۱۱-۱۴۱)$$

$$\tilde{\mathbf{h}}'_{(k)} = \mathbf{o}^{u1}_{(k)} \times \tilde{\mathbf{h}}_{(k)} \quad (۱۱-۱۴۲)$$

$$\mathbf{h}_{(k+1)} = \mathbf{o}_{(k)} = \mathbf{h}''_{(k)} + \tilde{\mathbf{h}}'_{(k)} \quad (۱۱-۱۴۳)$$

۱۱.۱۰.۴ آموزش یک واحد GRU با الگوریتم پس انتشار خطا به روش گرادیان نزولی

به منظور سهولت در محاسبات برای تعریف مشتقات زنجیره برای یک واحد GRU، فرض می کنیم ساختار مدنظر فقط از یک لایه که دارای یک واحد GRU است تشکیل شده است. بنابراین طبق این فرض خروجی واحد GRU همان خروجی نهائی ساختار است که با مقایسه آن با مقدار مطلوب خطای ساختار تعریف می شود. همچنین تابع هزینه ساختار را میانگین مربعات خطا، E، در نظر می گیریم. با این مفروضات مشتقات زنجیره ای هر یک از وزن های واحد را به صورت روابط زیر تعریف می کنیم:

$$\frac{\partial E}{\partial W^{a1}} = \underset{e}{\frac{\partial E}{\partial \mathbf{e}}} \underset{-1}{\frac{\partial \mathbf{e}}{\partial \mathbf{o}}} \underset{1}{\frac{\partial \mathbf{o}}{\partial \tilde{\mathbf{h}}'_{(k)}}} \underset{\mathbf{o}^{u3}}{\frac{\partial \tilde{\mathbf{h}}'_{(k)}}{\partial \tilde{\mathbf{h}}_{(k)}}} \underset{f'_{(\mathbf{net}^{a})}}{\frac{\partial \tilde{\mathbf{h}}_{(k)}}{\partial \mathbf{net}^{a}}} \underset{\mathbf{x}_{(k)}}{\frac{\partial \mathbf{net}^{a}}{\partial W^{a1}}} = -e f'_{(\mathbf{net}^{a}_{(k)})} \mathbf{o}^{u3} \mathbf{x}_{(k)} \quad (۱۱-۱۴۴)$$

$$\frac{\partial E}{\partial W^{a2}} = \underset{e}{\frac{\partial E}{\partial \mathbf{e}}} \underset{-1}{\frac{\partial \mathbf{e}}{\partial \mathbf{o}}} \underset{1}{\frac{\partial \mathbf{o}}{\partial \tilde{\mathbf{h}}'_{(k)}}} \underset{\mathbf{o}^{u3}}{\frac{\partial \tilde{\mathbf{h}}'_{(k)}}{\partial \tilde{\mathbf{h}}_{(k)}}} \underset{f'_{(\mathbf{net}^{a})}}{\frac{\partial \tilde{\mathbf{h}}_{(k)}}{\partial \mathbf{net}^{a}}} \underset{\mathbf{h}'_{(k)}}{\frac{\partial \mathbf{net}^{a}}{\partial W^{a2}}} \quad (۱۱-۱۴۵)$$

$$\frac{\partial E}{\partial W^{u1}} = \underset{e}{\frac{\partial E}{\partial \mathbf{e}}} \underset{-1}{\frac{\partial \mathbf{e}}{\partial \mathbf{o}}} \Big(\underset{1}{\frac{\partial \mathbf{o}}{\partial \tilde{\mathbf{h}}'_{(k)}}} \underset{\mathbf{o}^{a}}{\frac{\partial \tilde{\mathbf{h}}'_{(k)}}{\partial \mathbf{o}^{u1}}} + \underset{1}{\frac{\partial \mathbf{o}}{\partial \mathbf{h}''_{(k)}}} \underset{\mathbf{h}_{(k)}}{\frac{\partial \mathbf{h}''_{(k)}}{\partial \mathbf{o}^{u2}}} \underset{-1}{\frac{\partial \mathbf{o}^{u2}}{\partial \mathbf{o}^{u1}}} \Big) \underset{f'_{(\mathbf{net}^{u})}}{\frac{\partial \mathbf{o}^{u1}}{\partial \mathbf{net}^{u}}} \underset{\mathbf{x}_{(k)}}{\frac{\partial \mathbf{net}^{u}}{\partial W^{u1}}} \quad (۱۱-۱۴۶)$$

$$\frac{\partial E}{\partial W^{u2}}=\frac{\partial E}{\partial \mathbf{e}}\frac{\partial \mathbf{e}}{\partial \mathbf{o}}\Big(\underbrace{\frac{\partial \mathbf{o}}{\partial \tilde{\mathbf{h}}'_{(k)}}}_{1}\underbrace{\frac{\partial \tilde{\mathbf{h}}'_{(k)}}{\partial \mathbf{o}^{u1}}}_{\mathbf{o}^{a}}+\underbrace{\frac{\partial \mathbf{o}}{\partial \mathbf{h}''_{(k)}}}_{1}\underbrace{\frac{\partial \mathbf{h}''_{(k)}}{\partial \mathbf{o}^{u2}}}_{\mathbf{h}_{(k)}}\underbrace{\frac{\partial \mathbf{o}^{u2}}{\partial \mathbf{o}^{u1}}}_{-1}\Big)\underbrace{\frac{\partial \mathbf{o}^{u1}}{\partial \mathbf{net}^{u}}}_{f'_{(\mathbf{net}^{u})}}\underbrace{\frac{\partial \mathbf{net}^{u}}{\partial W^{u2}}}_{\mathbf{h}'_{(k)}} \tag{۱۱-۱۴۷}$$

$$\frac{\partial E}{\partial W^{r1}}=\frac{\partial E}{\partial \mathbf{e}}\frac{\partial \mathbf{e}}{\partial \mathbf{o}}\Big(\underbrace{\frac{\partial \mathbf{o}}{\partial \tilde{\mathbf{h}}'_{(k)}}}_{1}\underbrace{\frac{\partial \tilde{\mathbf{h}}'_{(k)}}{\partial \mathbf{o}^{u1}}}_{\mathbf{o}^{a}}+\underbrace{\frac{\partial \mathbf{o}}{\partial \mathbf{h}''_{(k)}}}_{1}\underbrace{\frac{\partial \mathbf{h}''_{(k)}}{\partial \mathbf{o}^{u2}}}_{\mathbf{h}_{(k)}}\underbrace{\frac{\partial \mathbf{o}^{u2}}{\partial \mathbf{o}^{u1}}}_{-1}\Big)\cdots \tag{۱۱-۱۴۸}$$

$$\cdots\underbrace{\frac{\partial \mathbf{o}^{u1}}{\partial \mathbf{net}^{u}}}_{f'_{(\mathbf{net}^{u})}}\underbrace{\frac{\partial \mathbf{net}^{u}}{\partial \mathbf{h}'_{(k)}}}_{W^{u2}}\underbrace{\frac{\partial \mathbf{h}'_{(k)}}{\partial \mathbf{o}^{r}}}_{\mathbf{h}_{(k)}}\underbrace{\frac{\partial \mathbf{o}^{r}}{\partial \mathbf{net}^{r}}}_{f'_{(\mathbf{net}^{r})}}\underbrace{\frac{\partial \mathbf{net}^{r}}{\partial W^{r1}}}_{\mathbf{x}_{(k)}}$$

$$\frac{\partial E}{\partial W^{r2}}=\frac{\partial E}{\partial \mathbf{e}}\frac{\partial \mathbf{e}}{\partial \mathbf{o}}\Big(\underbrace{\frac{\partial \mathbf{o}}{\partial \tilde{\mathbf{h}}'_{(k)}}}_{1}\underbrace{\frac{\partial \tilde{\mathbf{h}}'_{(k)}}{\partial \mathbf{o}^{u1}}}_{\mathbf{o}^{a}}+\underbrace{\frac{\partial \mathbf{o}}{\partial \mathbf{h}''_{(k)}}}_{1}\underbrace{\frac{\partial \mathbf{h}''_{(k)}}{\partial \mathbf{o}^{u2}}}_{\mathbf{h}_{(k)}}\underbrace{\frac{\partial \mathbf{o}^{u2}}{\partial \mathbf{o}^{u1}}}_{-1}\Big)\cdots \tag{۱۱-۱۴۹}$$

$$\cdots\underbrace{\frac{\partial \mathbf{o}^{u1}}{\partial \mathbf{net}^{u}}}_{f'_{(\mathbf{net}^{u})}}\underbrace{\frac{\partial \mathbf{net}^{u}}{\partial \mathbf{h}'_{(k)}}}_{W^{u2}}\underbrace{\frac{\partial \mathbf{h}'_{(k)}}{\partial \mathbf{o}^{r}}}_{\mathbf{h}_{(k)}}\underbrace{\frac{\partial \mathbf{o}^{r}}{\partial \mathbf{net}^{r}}}_{f'_{(\mathbf{net}^{r})}}\underbrace{\frac{\partial \mathbf{net}^{r}}{\partial W^{r2}}}_{\mathbf{h}_{(k)}}$$

طبق مشـــتقات زنجیره ای تعریف شـــده در روابط ۱۴۴-۱۱ تا ۱۴۹-۱۱، مقدار هر یک از پارامترهای واحد GRU در به صورت روابط زیر محاسبه می شود:

$$W^{a1}_{(k+1)}=W^{a1}_{(k)}-\eta\frac{\partial E}{\partial W^{a1}}(k) \tag{۱۱-۱۵۰}$$

$$W^{a2}_{(k+1)}=W^{a2}_{(k)}-\eta\frac{\partial E}{\partial W^{a2}}(k) \tag{۱۱-۱۵۱}$$

$$W^{u1}_{(k+1)}=W^{u1}_{(k)}-\eta\frac{\partial E}{\partial W^{u1}}(k) \tag{۱۱-۱۵۲}$$

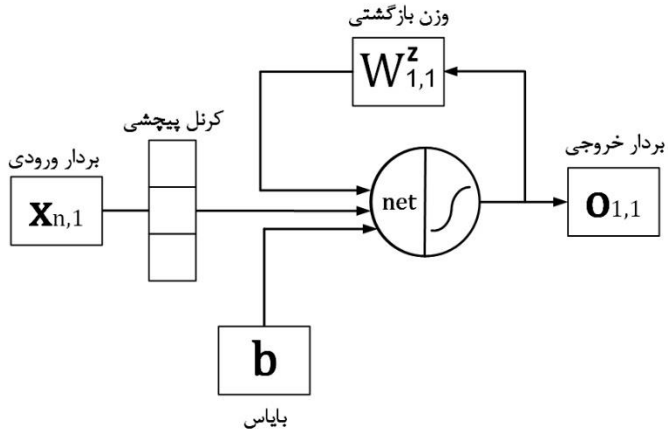
$$W^{u2}_{(k+1)}=W^{u2}_{(k)}-\eta\frac{\partial E}{\partial W^{u2}}(k) \tag{۱۱-۱۵۳}$$

$$W^{r1}_{(k+1)}=W^{r1}_{(k)}-\eta\frac{\partial E}{\partial W^{r1}}(k) \tag{۱۱-۱۵۴}$$

$$W^{r2}_{(k+1)}=W^{r2}_{(k)}-\eta\frac{\partial E}{\partial W^{r2}}(k) \tag{۱۱-۱۵۵}$$

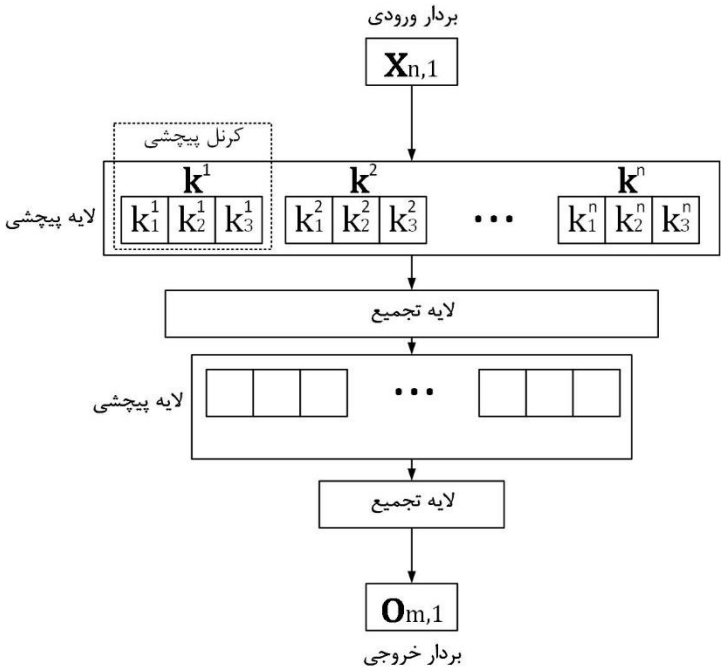
۱۱.۱۱ لایه های پیچشی[1] در دادگان زمانی

در فصل ۵ بررسی کردیم که اعمال یک فیلتر بر روی یک سیگنال معادل تعریف یک ضرب پیچشی بین تابع فیلتر، کرنل[2]، و سری زمانی است. می توانیم اعمال این کرنل ها بر روی دادگان سیگنال، سری زمانی، را به صورت اعمال آن به هر یک از نمونه مجموعه دادگان آموزشی که آن را از دادگان سری زمانی اصلی ایجاد کرده ایم، مطابق مطالب بخش ۱-۲-۱۱، اعمال کنیم. همانند خروجی فیلتر اعمال شده به کل یک سری زمانی، اعمال آن به هر نمونه نیز باعث ایجاد نگاشت پیچشی بر روی مقادیر هر نمونه از دادگان آموزشی خواهد بود. همچنین با اعمال لایه تجمیع[3] می توانیم ابعاد نمونه ها را کاهش دهیم. کرنل های پیچشی را هم می توانیم قبل از ورود یک بردار به نورون بازگشتی و یا یک واحد دروازه ای اعمال کنیم که در این صورت همانند افزودن پارامترهای پیچشی به ساختار نورون است و هم می توانیم از توالی کرنل های پیچشی نگاشت ها و کاهش بعد های مناسبی بر روی نمونه ها اعمال کرده و در نهایت خروجی نهایی این توالی را یا مستقیما به عنوان خروجی ساختار در نظر بگیریم. شکل ۳۷-۱۱ و شکل ۳۸-۱۱ به ترتیب نمایش استفاده از یک کرنل پیچشی در ورودی یک نورون با پسخور محلی و ساختار متشکل از کرنل های پیچشی که همانند لایه های یک شبکه عصبی در کنار هم قرار گرفته اند.



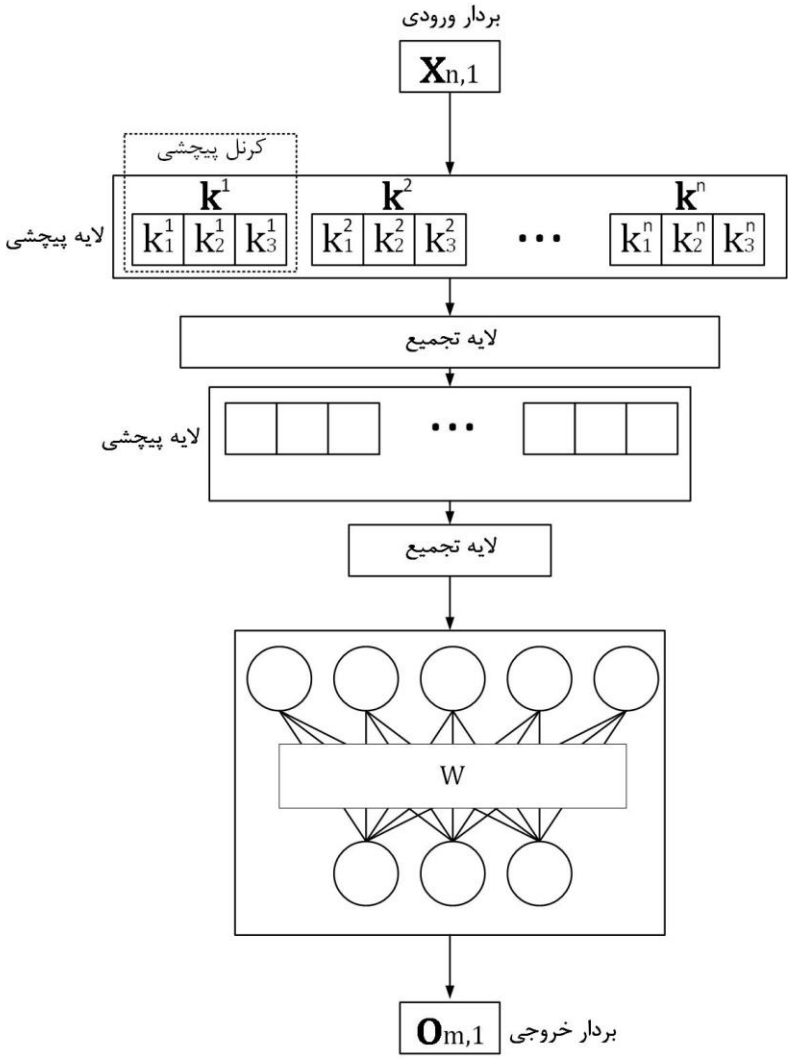
شکل ۳۷-۱۱: استفاده از کرنل پیچشی در ساختار نورون بازگشتی ساده

[1] Convolutional layers
[2] Kernel
[3] Pooling layer



شکل ۳۸-۱۱: ساختار پیچشی برای دادگان زمانی

در صورت استفاده از ساختاری مشابه شکل ۳۸-۱۱ در انتهای ساختار می توانیم از یک یا دو لایه پرسپترون استفاده کرده و خروجی قسمت پیچشی را ابتدا وارد این لایه های پرسپترون و در نهایت خروجی لایه انتهایی پرسپترون را به عنوان خروجی نهایی ساختار در نظر بگیریم. شکل ۳۹-۱۱ استفاده از ساختار ترکیبی پیچشی و پرسپترون را نشان می دهد. اندازه ابعاد بردار ورودی در ساختار پیچشی مشابه شکل ۳۸-۱۱ محدودیتی ندارد و فقط باید بزرگ تر یا مساوی کرنل اولین لایه باشد، ولی همواره باید اندازه کرنل ها و اندازه بردار لایه تجمیع را به گونه ای تنظیم کنیم که در نهایت خروجی ساختار ابعادی برابر با ابعاد بردار مقدار مطلوب داشته و یا در صورت استفاده از شبکه پرسپترون در انتهای ساختار مشابه شکل ۳۹-۱۱ پیچشی خروجی قسمت پیچشی دارای ابعادی مناسب به عنوان ورودی لایه پرسپترون باشد.



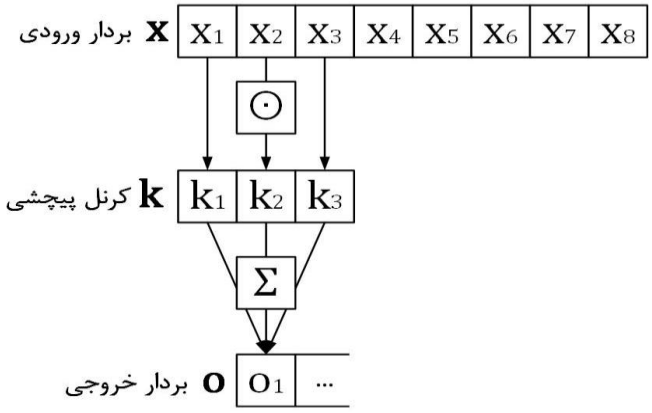
شکل ۳۹-۱۱: ساختار پیچشی با شبکه پرسپترون دو لایه

۱۱.۱۱.۱ نحوه حرکت یک کرنل به روی نمونه ورودی و گام حرکتی[1]

یک بردار مطابق رابطه ۱۵۶-۱۱ با اندازه $n = 8$ را به عنوان ورودی و یک بردار به اندازه $m = 3$ را به عنوان کرنل مطابق رابطه ۱۵۷-۱۱ فرض می کنیم. برای ضرب پیچشی کرنل در ورودی ابتدا دو بردار مطابق شکل ۴۰-۱۱ طوری در کنار هم قرار می گیرند که اولین بعد هر کدام رو به روی هم قرا بگیرد. سپس ابعادی که رو به روی هم قرار گرفته اند با هم به صورت درایه به

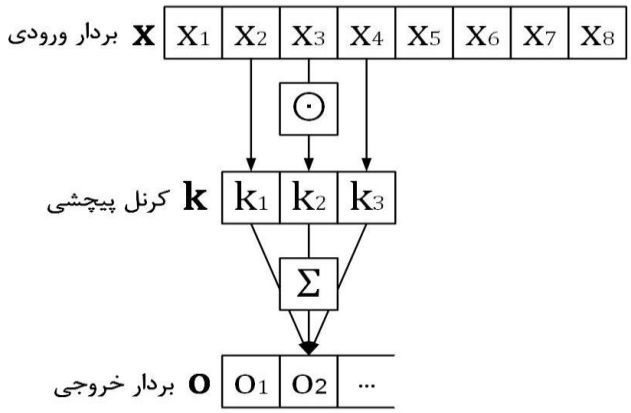
[1] Stride

درایه، ضرب هادامارد[1] که در فصل ۲ بررسی شده است، ضرب شده و خروجی آن گام حرکتی را می سازند. پس از اتمام این مرحله بردار کرنل روی بردار ورودی می لغزد تا درایه های دیگری رو به روی هم قرار گرفته و ضرب هادامارد دوباره اجرا شود، میران لغزش کرنل بر روی ورودی همان گام حرکتی است. بدین ترتیب اگر اندازه گام حرکتی را یک در نظر بگیریم در این صورت در گام دوم با لغزش کرنل بر روی ورودی بعد اول کرنل رو به روی بعد دوم بردار ورودی قرار گرفته و عمل ضرب هادامارد مطابق شکل ۴۱-۱۱ اجرا می شود. در هر گام حرکتی یک بعد از بردار خروجی با مجموع ضرب‌های هر دو بعد کرنل در ورودی مطابق شکل۴۱-۱۱ تشکیل می شود، همچنین در صورتی که اندازه گام حرکتی با اندازه کرنل برابر باشد در این صورت ابعادی از ورودی که در کرنل ضرب می شوند با گام حرکتی قبل و بعد هم پوشانی نخواهند داشت؛ هیچ کدام از بعد های ورودی در دو گام حرکتی برابر نخواهد بود. البته در حالتی که برای گام حرکتی اندازه بزرگی تعریف شود بار محاسباتی کاهش می یابد ولی در مقابل بعد بردار خروجی کوچکتر شده و اطلاعات زیادی از ورودی نسبت به حالتی که مقدار گام حرکتی کوچکتر است با اعمال ضرب پیچشی از بین خواهند رفت، بنابراین در حالت کلی بهتر است همواره اندازه گام حرکتی را مقادیر کوچک، یک، تعریف کنیم.



شکل ۴۰-۱۱: اولین گام ضرب پیچشی

[1] Hadamard product



شکل ۴۱-۱۱: ادامه روند ضرب پیچشی

$$\mathbf{x}=\begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \\ x_6 \\ x_7 \\ x_8 \end{bmatrix} \tag{۱۱-۱۵۶}$$

$$\mathbf{k}=\begin{bmatrix} k_1 \\ k_2 \\ k_3 \end{bmatrix} \tag{۱۱-۱۵۷}$$

اندازه بردار خروجی یک ضرب پیچشی مطابق رابطه ۱۱-۱۵۸ محاسبه می شود، در رابطه ۱۱-۱۵۸، l,m,n به ترتیب اندازه بردار ورودی، کرنل و خروجی و p اندازه بعد گذاری که در ادامه معرفی می کنیم، این مقدار برای بردار ورودی مطابق شکل ۴۰-۱۱ و ۴۱-۱۱ برابر صفر است، و s نیز اندازه گام حرکتی است که در عملیاتی مطابق شکل ۴۰-۱۱ و ۴۱-۱۱ یک فرض شده است. در رابطه ۱۱-۱۵۸ اندازه کرنل، بعد گذاری و گام حرکتی به گونه ای انتخاب می شود که حاصل کسر در این رابطه همواره عدد صحیح باشد، اما در مواردی که این کسر حاصلی غیرصحیح داشته باشد

مقدار آن را اولین عدد صحیح بزرگ تر[1] از آن در نظر می گیریم.

$$l = \frac{n-m+p}{s} + 1 \qquad (۱۱-۱۵۸)$$

باید در نظر داشته باشیم که شکل ۱۱-۴۰ و ۱۱-۴۱ و روابط ۱۱-۱۵۶ و ۱۱-۱۵۷ برای حالتی است که مجموعه دادگان آموزشی فقط از یک سری زمانی مشابه شکل ۱۱-۱ باشند. در صورتی که دادگان موجود برای چند سری زمانی مشابه شکل ۱۱-۵ باشند در این صورت، بردارهای ورودی همه سری های زمانی باید دارای اندازه برابر بوده و این بردارها برخلاف روش های پیشین که یک بردار ورودی با طولی برابر مجموع بردارهای استخراج شده از هر سری را تشکیل می دادند، یک ماتریس با ابعاد تشکیل $m \times n$ دهند که m تعداد سری های زمانی است، همچنین کرنل ضرب پیچشی به نیز در این حالت به صورت یک ماتریس با ابعاد $m \times l$ که l اندازه کرنل پیچشی است. به بیانی دیگر در این روش برای هر سری یک کرنل پیچشی مجزا تعریف می شود ولی اندازه کرنل های، و همچنین بردارهای ورودی، همه سری ها با هم برابر است. رابطه ۱۱-۱۵۹ و ۱۱-۱۶۰ به ترتیب ماتریس ورودی، A، و کرنل پیچشی برای دادگان مربوط به چند سری را نشان می دهند.

$$A = \begin{bmatrix} x_1 & x_2 & \cdots & x_n \\ y_1 & y_2 & \cdots & y_n \\ \vdots & \vdots & \vdots & \vdots \\ z_1 & z_2 & \cdots & z_n \end{bmatrix}_{m \times n} \qquad (۱۱-۱۵۹)$$

$$K = \begin{bmatrix} k_{x1} & k_{x2} & \cdots & k_{xl} \\ k_{y1} & k_{y2} & \cdots & k_{yl} \\ \vdots & \vdots & \vdots & \vdots \\ k_{z1} & k_{z2} & \cdots & k_{zl} \end{bmatrix}_{m \times l} \qquad (۱۱-۱۶۰)$$

۱۱.۱۱.۲ روابط پیشرو در ضرب پیچشی یک بعدی

با توجه به تعاریف گام حرکتی و نحوه انجام ضرب هادامارد می توانیم رابطه پیشروی ضرب پیچشی و ایجاد بردار خروجی متناظر از یک بردار ورودی و کرنل را مطابق رابطه ۱۱-۱۶۱ تعریف کنیم، در رابطه ۱۱-۱۶۱، مقدار l از رابطه ۱۱-۱۵۸ محاسبه می شود، همچنین نماد : به معنی جدا کردن بازه ای از کل بردار، m اندازه بردار کرنل است.

[1] Ceil

$$o_i = \sum_{p=0}^{m-1} x_{i+p} k_{p+1} = sum(\mathbf{x}_{(i:i+m-1)} \odot \mathbf{k}), i = 1, ..., l \qquad (۱۱-۱۶۱)$$

می توانیم پس از محاسبه بردار خروجی رابطه ۱۶۱-۱۱ یک تابع فعال ساز نیز به آن اعمال کنیم، در این صورت تابع فعال ساز به هر یک از درایه های بردار خروجی به صورت مستقل طبق رابطه ۱۶۲-۱۱ اعمال می شود:

$$o_i' = f(o_i) \qquad (۱۱-۱۶۲)$$

همچنین در صورت استفاده از بایاس[1]، این بایاس قبل از اعمال رابطه ۱۶۲-۱۱ به صورت یک بردار هم اندازه با بردار $\boldsymbol{o}$ در نظر گرفته شده و با آن، بردار $\boldsymbol{o}$، جمع می شود.

۱۱.۱۱.۳ بعد گذاری[2]

در یک عملیات پیچشی مطابق شکل ۴۰-۱۱، با فرض گام حرکتی یک برای ضرب پیچشی، در طول روند اجرای ضرب پیچشی بر روی یک بردار ورودی، کرنل پیچشی تنها یک با رو به روی بعد اول و آخر بردار ورودی قرار گرفته و بدین ترتیب بعد ابتدایی و انتهایی بردار ورودی تنها یک بار در یکی از ابعاد کرنل ضرب می شوند در حالی که همه ابعاد بین این دو بعد چند بار، دو یا سه برای حالت شکل ۴۰-۱۱، در یکی از ابعاد کرنل ضرب شده و نقش بیشتری نسبت به ابعاد ابتدایی و انتهایی در ایجاد اطلاعات موجود در هر یک از ابعاد بردار خروجی دارند، برای رفع این مشکل به ابتدا و انتهای بردار ورودی تعداد مناسبی بعد اضافه می کنیم تا در روند ضرب پیچشی کرنل با لغزش به روی این ابعاد افزوده، ابعاد ابتدا و انتهایی بردار اصلی نیز چند بار، بعد های نزدیک به دو انتها نیز بیشتر از قبل، در روند ضرب پیچشی دخیل شوند. معمولا این ابعاد افزوده به طور مساوی در ابتدا و انتهای بردار ورودی قرار گرفته و مجموع تعداد آن ها در دو طرف از ۱۵۶-۱۱ رابطه محاسبه می شود. همچنین مقدار این ابعاد صفر در نظر گرفته می شود که به این بعدگذاری صفر[3] اطلاق می شود، البته در مواردی نسبت به رفتار دادگان می توان برای آن ها مقادیری غیر صفر در نظر گرفت. در رابطه ۱۶۳-۱۱، p, m به ترتیب اندازه کرنل پیچشی و مجموع ابعاد لازم برای بعد گذاری است.

$$p = m - 1 \qquad (۱۱-۱۶۳)$$

طبق رابطه ۱۶۳-۱۱ تعداد بعد هر طرف، ابتدا و انتهای بردار، از رابطه ۱۶۴-۱۱ محاسبه می

[1] Bias
[2] Padding
[3] Zero padding

شود.

$$p' = \frac{p}{2} \tag{۱۱-۱۶۴}$$

۱۱.۱۱.۴ آموزش کرنل یک بعدی پیچشی به روش گرادیان نزولی

رفتار کرنل های پیچشی در ساختارها تا حدودی مشابه رفتار پارامترهای وزن است. همان طور که پارامترهای وزن یک ساختار را با تعریف روابط مشتقات زنجیره ای برای آن ها آموزش می دادیم، می توانیم برای پارامترهای یک کرنل پیچشی که همان مقادیر ابعاد مختلف آن کرنل است نیز با توسعه مشتقات زنجیره ای مناسب، روابط آموزشی تعریف کنیم. بنابراین برای یک ساختار فرضی متشکل از یک لایه ای که فقط شامل یک کرنل پیچشی است روابط مشتقات زنجیره ای مطابق روابط ۱۱-۱۶۵ تا ۱۱-۱۶۷ را با فرض مجموع مربعات خطا، E، به عنوان تابع هزینه را تعریف می کنیم.

$$\frac{\partial E}{\partial k_i} = \frac{\partial E}{\partial \mathbf{e}} \sum_{j=1}^{l} \frac{\partial \mathbf{e}}{\partial o_j} \frac{\partial o_j}{\partial k_i} \tag{۱۱-۱۶۵}$$

$$\frac{\partial o_j}{\partial k_i} = \frac{\partial}{\partial k_i} (sum(\mathbf{x}_{(j:j+m-1)} \odot \mathbf{k})) \tag{۱۱-۱۶۶}$$

$$\frac{\partial}{\partial k_i} (sum(\mathbf{x}_{(j:j+m-1)} \odot \mathbf{k})) = \frac{\partial}{\partial k_i} (k_1 x_j + k_2 x_{j+1} + ... + k_i x_{j+i-1}) = x_{j+i-1} \tag{۱۱-۱۶۷}$$

با توجه به روابط ۱۱-۱۶۵ تا ۱۱-۱۶۷ مقادیر بردار کرنل در هر گام آموزشی[1] از رابطه ۱۱-۱۶۸ محاسبه می شود.

$$k_{i(k+1)} = k_{i(k)} - \eta \frac{\partial E}{\partial k_i}(k) \tag{۱۱-۱۶۸}$$

در صورتی که مشتقات زنجیره ای از لایه پیچشی عبور کرده و وارد لایه قبل شوند مشتقات زنجیره ای مطابق روابط ۱۱-۱۶۹ تا ۱۱-۱۷۱ خواهد بود.

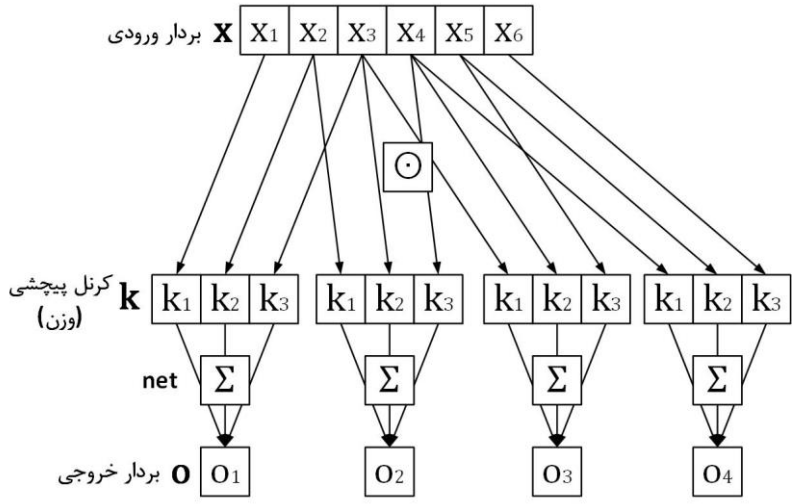
[1] Epoch

$$\frac{\partial E}{\partial \mathbf{e}} \sum_{j=1}^{l} \frac{\partial \mathbf{e}}{\partial o_j} \frac{\partial o_j}{\partial x_i} \frac{\partial x_i}{\cdots} \cdots \qquad (۱۱-۱۶۹)$$

$$\frac{\partial o_j}{\partial x_i} = \frac{\partial}{\partial x_i} (sum(\mathbf{x}_{(j:j+m-1)} \odot \mathbf{k})) \qquad (۱۱-۱۷۰)$$

$$\frac{\partial}{\partial x_i} (k_1 x_j + k_2 x_{j+1} + k_i x_{j+i-1} + ...) = \begin{cases} k_v, x_i \in \mathbf{x}_{(j:j+m-1)} \\ 0, x_i \notin \mathbf{x}_{(j:j+m-1)} \end{cases} \qquad (۱۱-۱۷۱)$$

در رابطه ۱۱-۱۷۱، k_v بعدی از کرنل اســـت که در آن گام رو به روی بعد x_i بردار ورودی قرار گرفته اسـت. اگر یک عملیات ضـرب پیچشـی را در طول گام های حرکتی گسـترش دهیم، مطابق شـکل ۴۲-۱۱ خواهد بود، با توجه به شـکل ۴۲-۱۱ می توانیم چنین برداشـت کنیم که ضـرب پیچشـی همانند یک لایه پرسـپترون با اتصـالات تنک، همه ورودی ها به همه نورون ها متصـل نیسـتند، با وزن های مشـترک بین همه نورون های لایه اسـت. در صـورت اسـتفاده از تابع فعال سـاز غیرخطی هر یک از ابعاد بردار خروجی که معادل یک net هسـتند وارد تابع شـده و بردار خروجی نهائی را تشکیل می دهند.



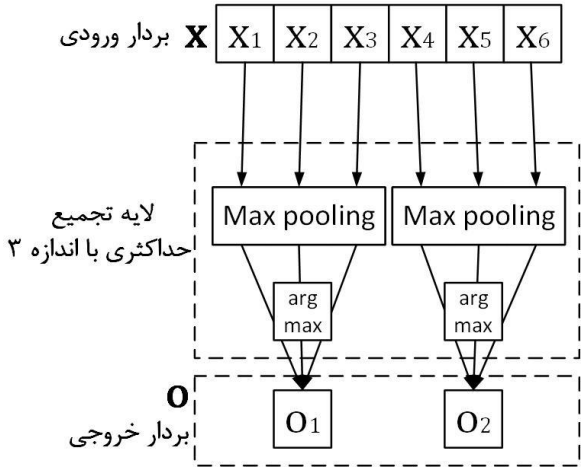
شکل ۴۲-۱۱: گسترش عملیات ضرب پیچشی

۱۱.۱۱.۵ لایه تجمیع[1] در ساختارهای پیچشی

از لایه های تجمیع که عملیاتی مشابه با ضرب کرنل پیچشی را بر روی یک بردار ورودی اعمال می کنند برای کاهش بعد ورودی استفاده می شود. با توجه به این که در عملیات ضرب پیچشی بر روی یک بردار مطابق رابطه، اندازه بردار خروجی متاثر از اندازه بردار کرنل اســـت، با تنظیم اندازه کرنل می توان ابعاد بردار خروجی را کنترل کرده و بدین ترتیب عمل کاهش بعد را با اســـتفاده از ضـرب پیچشـــی اعمال کرد. تفاوت لایه تجمیع با کرنل پیچشـــی در این اســـت که در لایه تجمیع پارامترهای آموزشی دخیل نبوده و برای آن روابط آموزشی تعریف نمی شود. یک لایه تجمیع مانند یک ضـرب پیچشـی اسـت که در آن همواره گام حرکتی هم اندازه کرنل اسـت، $m = s$، بنابراین لایه تجمیع یک ضـرب پیچشـــی بدون هم پوشـــانی اســـت. دو روش مرســـوم برای لایه تجمیع، تجمیع حداکثری و تجمیع میانگین هستند که در ادامه آن ها را بررسی خواهیم کرد.

۱۱.۱۱.۵.۱ تجمیع حداکثری[2]

در روش تجمیع حداکثری در هر گام حرکتی بعدی که حداکثر مقدار را در میان ابعادی از ورودی که رو به روی پنجره لایه تجمیع قرار دارند به عنوان خروجی آن گام حرکتی انتخاب شـــده و بعد متناظر در بردار خروجی را ایجاد می کند. شکل ۴۳-۱۱ نمایش روش تجمیع حداکثری است.



شکل ۴۳-۱۱: لایه تجمیع حداکثری

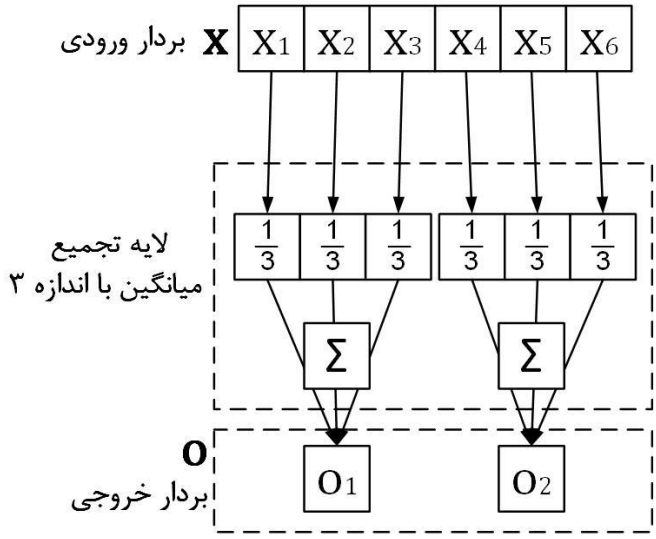
بدین ترتیب رابطه پیشـــرو یک لایه تجمیع حداکثری برای هر نمونه بردار ورودی به صـــورت رابطه ۱۷۲-۱۱ تعریف می شود، در این رابطه m اندازه لایه تجمیع است.

[1] Pooling layers
[2] Max pooling

$$o_i = \arg\max \mathbf{x}_{(m(i-1)+1:mi)}$$ (۱۱-۱۷۲)

۱۱.۱۱.۵.۲ تجمیع میانگین[1]

تفاوت روش تجمیع میانگین با روش تجمیع حداکثری در این اســـت که در تجمیع حداکثری فقط یک بعد از میان بعدهای هر گام حرکتی، بعد با مقدار حداکثری، به عنوان خروجی آن گام تعریف می شــود در حالی که در تجمیع میانگین خروجی هر گام حرکتی میانگین همه بعدهای آن گام است. با این توضیحات تجمیع حداکثری همانند یک کرنل پیچشی به اندازه کرنل و گام حرکتی m است که مقادیر هر بعد کرنل برابر $\frac{1}{m}$ است. شکل ۴۴-۱۱ نمایش روش تجمیع میانگین است:



شکل ۴۴-۱۱: لایه تجمیع میانگین

بدین ترتیب رابطه پیشـــرو یک لایه تجمیع میانگین برای هر نمونه ورودی به صـــورت رابطه ۱۱-۱۷۳ تعریف می شود:

$$o_i = \frac{1}{m} sum(\mathbf{x}_{(m(i-1)+1:mi)})$$ (۱۱-۱۷۳)

اندازه بردار خروجی لایه تجمیع از رابطه ۱۱-۱۵۸، با فرض $m = s$ محاسبه می شود.

[1] Average pooling

۱۱.۱۱.۵.۳ مشتقات زنجیره ای لایه تجمیع

با توجه به ماهیت لایه تجمیع، روابط آموزشی برای این لایه تعریف نمی شود اما در صورتی که این لایه به صورت سری بین سایر لایه های یک ساختار قرار گرفته باشد رابطه مشتق زنجیره ای که از این لایه برای آموزش پارامترهای لایه قبل از آن استفاده می شود می گذرد برای تجمیع حداکثری مطابق رابطه ۱۱-۱۷۴ و ۱۱-۱۷۵ و برای تجمیع میانگین مطابق رابطه ۱۱-۱۷۶ است.

$$\frac{\partial E}{\partial \mathbf{e}}\frac{\partial \mathbf{e}}{\partial o_j}\frac{\partial o_j}{\partial x_i}\cdots \qquad e \quad -1$$

(۱۱-۱۷۴)

$$\frac{\partial o_j}{\partial x_i}=\begin{cases}1, o_i = \arg\max \mathbf{x}_{(m(j-1)+1:mj)}\\ 0, o_i \neq \arg\max \mathbf{x}_{(m(j-1)+1:mj)}\end{cases}$$

(۱۱-۱۷۵)

$$\frac{\partial E}{\partial \mathbf{e}}\frac{\partial \mathbf{e}}{\partial \mathbf{o}}\frac{\partial \mathbf{o}}{\partial \mathbf{x}}\cdots \qquad e \quad -1 \quad \frac{1}{m}$$

(۱۱-۱۷۶)

۱۱.۱۲ شبکه های شبه بازگشتی[1]

با مقایسه دو روش ساختار واحد دروازه ای مانند LSTM و ساختار پیچشی مشابه شکل ۱۱-۳۸ و یا ۱۱-۳۹ استنباط های ذیل حاصل می شود:

۱. ساختار های دروازه ای علیرغم عملکرد مناسب در پردازش دادگان مربوط به سری های زمانی حجم محاسبات نسبت به همه ساختارهای معرفی شده دیگر بیشتر بوده و این موضوع سبب افزایش زمان مورد نیاز برای آموزش ساختار خواهد شد. افزودن کرنل های پیچشی به این ساختارها مشابه ساختار شکل ۱۱-۳۷ باعث تشدید این موضوع می شود.

۲. مسئله پس انتشار در زمان همچنان در ساختارهای دروازه ای مطرح بوده و برای اجتناب از آن همواره فرض استقلال برای بردارهای حافظه ساختار در نظر می گیریم.

۳. همه ی اطلاعات موجود در یک واحد دروازه‌ای اعم از حافظه، ورودی و... با استفاده از پارامترهایی در نهایت روی یک بردار خروجی نگاشت شده و از این نظر این بردار خروجی ممکن است نگاشت مناسبی از اطلاعات ارائه نکند، این موضوع به ویژه در

[1] Quasi recurrent neural networks (QRNN)

بخش های کنترلی ساختار که یک مقدار ثابت، اسکالر، در کل یک بردار ضرب می شود چالش برانگیز است.

۴. از طرفی ساختارهای پیچشی مشابه شکل ۳۸-۱۱ نسبت به ساختارهای دروازه ای پیچیدگی محاسباتی کمتری دارند. همچنین با توجه به عدم وابستگی خروجی در هر گام زمانی نسبت به خروجی گام های قبل در ساختارهای پیچشی می توان در روند اجرای آن ها از روش های پردازش موازی[1] که قابلیت مرسومی در بسیاری از رایانه های امروزی، به ویژه رایانه های مجهز به سخت افزار واحد پردازش گرافیکی[2] و واحد پردازش تانسور[3]، بیشتر استفاده کرد در حالی که این امکان در ساختارهای دروازه ای محدود به عملیات های داخل ساختار در هر گام زمانی است. بدین ترتیب ساختارهای پیچشی زمان کمتری برای محاسبه مقدار خروجی لازم خواهند داشت.

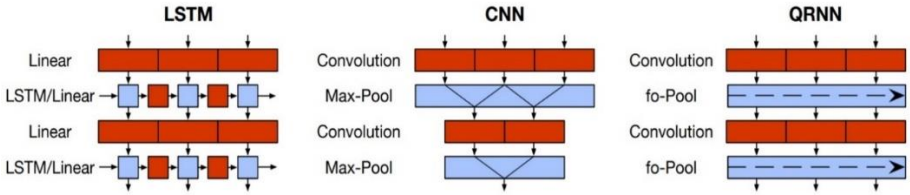
۵. در ساختارهای پیچشی پارامترها با در نظر گرفتن وابستگی بین ابعاد یک ورودی در چند گام حرکتی به بردار اعمال شده و از این نظر نسبت به ساختار های دروازه ای با توجه به چالش ذکر شده در مورد ۳ ارجحیت دارند.

۶. در ساختارهای پیچشی پارامترهای ساختار برای هر نمونه از مجموعه دادگان آموزشی به طور مستقل اعمال شده و از این لحاظ همانند ساختارهای دروازه ای مسئله پس انتشار در زمان در آن ها مطرح نمی شود، اما این موضوع باعث می شود وابستگی بین نمونه ها که ساختارهای دروازه ای به واسطه داشتن حافظه آن را شبیه سازی می کردند به خوبی در ساختار های پیچشی مدل سازی نشود.

با توجه به توضیحات فوق الذکر لزوم توسعه ساختاری بین ساختارهای دروازه ای و پیچشی که بتوانند از مزایای هر دو ساختار استفاده کند قابل توجیه است. بدین منظور ساختار های شبه بازگشتی معرفی شده اند که به بررسی آن ها می پردازیم. ساختارهای شبه بازگشتی با بهره گیری از مزایای ساختار واحدهای دروازه ای قابلیت پردازش موازی بیشتری را ارائه می کنند، علاوه بر این همانند ساختارهای پیچشی وابستگی بین ابعاد را نیز حفظ کرده و در نهایت بردار خروجی حاوی اطلاعات مناسب تری نسبت به ساختار دروازه ای ایجاد می کنند. شکل ۴۵-۱۱ الف، ب و ج به ترتیب مقایسه شماتیک سه ساختار (توالی ساختارها) واحد دروازه ای LSTM، پیچشی و شبه بازگشتی است هر یک از بلوک ها در شکل ۴۵-۱۱ نشان دهنده یک عملیات با پارامترهای آموزشی، مانند اعمال وزن به ورودی، و یا عملیات بدون پارامتر (مانند اعمال تابع فعال ساز) است،

[1] این قابلیت برای رایانه هایی که واحد پردازشی آن ها دارای چند هسته پردازشی است، قابل تعریف است.
[2] Graphics processing unit (GPU)
[3] Tensor processing unit (TPU)

بلوک هایی که به صورت به هم چسبیده در شکل ۴۵-۱۱ نشان داده شده اند، قابلیت پیاده سازی به صورت پردازش موازی را دارند.



شکل ۴۵-۱۱: مقایسه ساختارهای دروازه ای، پیچشی و شبه بازگشتی[۷۹]

روند اجرا و ساختار کلی یک ساختار شبه بازگشتی تا حدودی مشابه با ساختار واحد های دروازه ای است و تفاوت فقط در عملیات تعریف شده و اتصالات بخش های مختلف ساختار است. با توجه به شکل ۴۵-۱۱ برای یک ساختار شبه بازگشتی قسمت فعال ساز و سه گیت فراموشی، خروجی و ورودی در نظر گرفته و آن ها را به ترتیب مطابق روابط ۱۷۷-۱۱ تا ۱۸۰-۱۱ تعریف می کنیم:

$$\mathbf{z} = \mathrm{Tanh}(\mathbf{k}_z * \mathbf{x}) \qquad (۱۱-۱۷۷)$$

$$\mathbf{f} = Sig(\mathbf{k}_f * \mathbf{x}) \qquad (۱۱-۱۷۸)$$

$$\mathbf{o} = Sig(\mathbf{k}_o * \mathbf{x}) \qquad (۱۱-۱۷۹)$$

$$\mathbf{i} = Sig(\mathbf{k}_i * \mathbf{x}) \qquad (۱۱-۱۸۰)$$

در روابط فوق $\boldsymbol{k}$ یک کرنل پیچشی و نماد * بیانگر ضرب پیچشی است، بنابراین برخلاف ساختارهای دروازه ای که در روابط پیشرو آن ها یک ماتریس وزن در بردار ورودی ضرب می شد در این ساختار از ضرب های پیچشی برای این روابط خروجی گیت ها استفاده شده است. همچنین با توجه به روابط فوق خروجی قسمت های کنترلی مانند رابطه ۱۷۸-۱۱ در این ساختار برخلاف ساختارهای دروازه ای که یک اسکالر بود، یک بردار است که متعاقبا به عنوان یک کرنل پیچشی برای ایجاد بردار خروجی نهائی ساختار استفاده می شود. این تفاوت بین دو ساختار باعث می شود بردار حاصل از ضرب پیچشی محتویات اطلاعاتی بهتری در مقایسه با بردار حاصل از ضرب خروجی کنترلی اسکالر در ساختار دروازه ای داشته باشد.

بردار خروجی نهایی، بردار حافظه کوتاه مدت، ساختار شبه بازگشتی را مطابق حالت های ذیل تعریف می کنیم:

۱. برای ساختار حافظه بلند مدت در نظر نگرفته و فقط یک بردار حافظه مشابه واحد

GRU تعریف کنیم. با توجه به این که در این حالت فقط گیت فراموشی در محاسبات دخیل است به آن روش تجمیع f[1] گفته می شود:

$$\mathbf{h}_{(k)} = \mathbf{f}_{(k)} \odot \mathbf{h}_{(k-1)} + (1-\mathbf{f}_{(k)}) \odot \mathbf{z}_{(k)} \quad (۱۱-۱۸۱)$$

۲. ساختار مشابه واحد LSTM دارای دو حافظه کوتاه مدت (خروجی نهائی) و بلند مدت باشد، در این حالت دو گیت فراموشی و خروجی دخیل بوده به آن روش تجمیع fo[2] گفته می شود، در شکل ۴۵-۱۱ این حالت نمایش داده شده است:

$$\mathbf{c}_{(k)} = \mathbf{f}_{(k)} \odot \mathbf{c}_{(k-1)} + (1-\mathbf{f}_{(k)}) \odot \mathbf{z}_{(k)} \quad (۱۱-۱۸۲)$$

$$\mathbf{h}_{(k)} = \mathbf{o}_{(k)} \odot \mathbf{c}_{(k)} \quad (۱۱-۱۸۳)$$

۳. گیت ورودی نیز در محاسبه خروجی نهائی دخیل شود، در این صورت به این روش، تجمیع ifo[3] گفته می شود:

$$\mathbf{c}_{(k)} = \mathbf{f}_{(k)} \odot \mathbf{c}_{(k-1)} + \mathbf{i}_{(k)} \odot \mathbf{z}_{(k)} \quad (۱۱-۱۸۴)$$

$$\mathbf{h}_{k} = \mathbf{o}_{k} \odot \mathbf{c}_{k} \quad (۱۱-۱۸۵)$$

در حالت های فوق منظور از تجمیع همان ضرب درایه به درایه، هادامارد، دو بردار برای تشکیل بردار خروجی بوده و با روش های تجمیعی که در بخش ۵-۱۱-۱۱ برای کاهش بعد معرفی شدند متفاوت است.

در رابطه ۱۶۱-۱۱ عملیات ضرب پیچشی که پیش تر بررسی کردیم در هر گام حرکتی کرنل پیچشی سه نمونه، سه بعد، از ورودی در عملیات دخیل هستند، این موضوع در صورتی که هر یک از ابعاد بردار ورودی یک نمونه ثبت شده از یک سری زمانی باشد باعث نقض تئوریک ساختار در اولین گام حرکتی از ضرب پیچشی که هم زمان سه نمونه، نمونه مربوط به سه زمان متفاوت، وارد ساختار می شوند، می شود بدین معنی که هر یک از این ابعاد بردار ورودی در واقع در یک زمان خاصی که با زمان نمونه دیگر متفاوت است ثبت شده اند و لذا باید در طول بعد زمان همواره یک نمونه، بعد، از متغیر وارد ساختار شود در حالی که در گام حرکتی اول ضرب پیچشی رابطه چنین برداشت می شود که در یک زمان سه نمونه، سه بعد، ثبت شده اند، البته در صورت تعریف گام حرکتی بزرگ تر از یک این موضوع فقط به گام اول محدود نمی شود. در ساختار شبه بازگشتی برای اینکه همواره یک نمونه ثبت شده، مقدار متغیر در یک گام زمانی، در هر گام حرکتی وارد

[1] f-pooling: forget pooling
[2] fo-pooling: forget-out put pooling
[3] ifo-pooling: input-forget-output pooling

ساختار شود و از نظر تئوریک ساختار شبیه سازی درستی در بعد زمان داشته باشد علاوه بر اینکه همواره مقدار گام حرکتی یک در نظر گرفته می شـــود، بعدگذاری مربوط به بردار ورودی بر خلاف رابطه ۱۶۴-۱۱، بعد گذاری فقط ســمت اولین بعد بردار اعمال شــده و تعداد آن از رابطه ۱۶۳-۱۱ محاسبه می شود. بدین ترتیب در هر گام حرکتی به فقط یک نمونه، بعد، جدید وارد عملیات ضرب پیچشـی می شـود. به این روش ضـرب پیچشـی مسـتتر[1] اطلاق می شـود. اسـتفاده از روش ضـرب پیچشـی مسـتتر با این که مشـکل مربوط به بعد زمان را حل می کند اما در عوض باعث می شـود آخرین بعد ورودی همچنان کمتر از سایر ابعاد در ضرب پیچشی دخیل باشد.

۱۱.۱۳ شبکه های هاپفیلد مدرن[2]

در بخش ۲-۱۱ در مورد ساختار شبکه های هاپفیلد بحث کرده و بررسی کردیم که این ساختار دارای یک حافظه محدود است که فقط قابلیت ذخیره چند الگوی محدود را دارد. همچنین با توجه به روش به روزرسانی واحدهای شبکه مقادیر متغیرها و ورودی این ساختار را به صورت باینری در نظر می گرفتیم. ساختار شـبکه یک شـبکه هاپفیلد مطابق شـکل ۱۵-۱۱ انعطاف پذیری مناسـبی نداشـــته و با ســـاختار شـــبکه های مرســـومی عمیق تفاوت های عمده دارد، به همین دلیل امکان استفاده از این ساختار به عنوان یک ساختار عمیق و یا ادغام آن در داخل یک ساختار ژرف، مانند ساختار خودرمزگذار بازگشتی که در فصل ۱۰ معرفی کردیم، دیگر وجود ندارد. شبکه های هاپفیلد مدرن برای رفع این محدودیت های شـبکه های هاپفیلد معرفی شـده اند. برای سـاختار شـبکه های هاپفیلد مدرن امکان تعریف متغیرهای تصـــادفی حقیقی و یا حتی موهومی وجود داشـــته و ورودی این سـاختار محدود به متغیرهای باینری نیسـت. همچنین با توجه به تعریف تابع انرژی جدید برای ساختار امکان ادغام شبکه هاپفیلد مدرن در ساختارهای عمیق به صورت یک لایه یا ساختار فراهم شده است. در ادامه به بررسی ساختار شبکه های هاپفیلد مدرن می پردازیم.

با بررسـی مجدد رابطه ۷-۱۱ که تابع انرژی یک شـبکه هاپفیلد اسـت می توانیم چنین برداشـت کنیم که این تابع با توجه به اینکه به صورت یک نگاشت خطی تعریف شـده اسـت به کندی همگرا می شـــود، برای رفع این مشـــکل می توانیم به تابع انرژی رابطه پس از اعمال نگاشـــت خطی یک نگاشت غیرخطی نیز مشابه توابع فعال ساز نورون ها اعمال کنیم بدین ترتیب تابع انرژی یک شبکه هاپفیلد به صورت رابطه ۱۸۶-۱۱ تعریف می شود که در آن f یک تابع غیرخطی مانند $f_{(x)} = x^a$ است. به این تابع غیرخطی ، تابع تعامل[3] گفته می شود.

$$E_{\theta(x)} = -f\left(\sum_{i=1}^{n-1}\sum_{j=i+1}^{n} x_i x_j\right) = -f\left(\boldsymbol{x}^T\boldsymbol{x}\right) \tag{۱۱-۱۸۶}$$

[1] Masked convolution
[2] Modern Hopfield networks
[3] Interaction function

روابط ۱۰-۱۱ و ۱۱-۱۱ که مربوط به ظرفیت یک شبکه هاپفیلد است را نیز مجددا با توجه به تابع انرژی غیرخطی رابطه بررسی می کنیم، می توان از روابط چنین برداشت کرد که با افزایش بعد مربوط به دادگان ظرفیت حافظه افزایش می یابد اما این موضوع باعث اختلال بیشتر در همگرائی تابع انرژی که در بخش ۱-۲-۱۱ به آن اشاره کردیم می شود برای مرتفع کردن این مسئله راهکار مناسب استفاده از تابع نمائی به عنوان تابع تعامل است، در صورت استفاده از تابع نمائی با توجه به علامت منفی در رابطه تابع انرژی، مقدار تابع انرژی بسیار سریع تر از حالت خطی به مقدار کمتری رسیده و همگرا می شود. بدین ترتیب تابع انرژی شبکه هاپفیلد مدرن را می توانیم به صورت رابطه ۱۱-۱۸۷ تعریف کنیم تا علاوه بر افزایش ظرفیت حافظه و بعد ورودی، همگرائی تابع انرژی ساختار را افزایش دهیم.

$$E_{\theta(\boldsymbol{x})} = -e^{(\sum_{i=1}^{n-1}\sum_{j=i+1}^{n} x_i x_j)} = -e^{(\boldsymbol{x}^T \boldsymbol{x})} \tag{۱۱-۱۸۷}$$

رابطه ۱۱-۱۸۷ برای یک الگوی ذخیره شده در شبکه تعریف شده است آن را برای m الگوی ذخیره شده به صورت رابطه ۱۱-۱۸۸ تعمیم می دهیم:

$$E_{\theta(\boldsymbol{x})} = \sum_{k=1}^{m} -e^{(\sum_{i=1}^{n-1}\sum_{j=i+1}^{n} x_i x_j)} = \sum_{k=1}^{m} -e^{(\boldsymbol{x}^T \boldsymbol{x})} \tag{۱۱-۱۸۸}$$

رابطه ۱۱-۱۸۸ را می توانیم به صورت رابطه ۱۱-۱۹۰ با استفاده از رابطه ۱۱-۱۸۹ بازنویسی کنیم. در روابط ذیل β یک پارامتر اسکالر کنترلی است که مقدار آن بین صفر تا یک انتخاب می شود.

$$lse(\beta, \mathbf{x}) = \beta^{-1} \log(\sum_{i} e^{\beta x_i}) \tag{۱۱-۱۸۹}$$

$$E_{\theta(\boldsymbol{x})} = -e^{lse(1, \boldsymbol{x}^T \boldsymbol{x})} \tag{۱۱-۱۹۰}$$

از آنجائی که مقادیر متغیرهای این ساختار متغیرهای حقیقی هستند ممکن است رابطه ۱۱-۱۹۰ دارای مقادیر بزرگی باشد که حافظه رایانه را اشغال می کند، برای رفع مشکل حافظه رایانه و نیز نزدیک تر کردن رابطه ۱۱-۱۹۰ به یک تابع هزینه استاندارد، مقدار منفی لگاریتم آن را به عنوان تابع انرژی نهائی ساختار به صورت رابطه ۱۱-۱۹۱ تعریف می کنیم.

$$E_{\theta(\boldsymbol{x})} = -\log(-e^{lse(1, \boldsymbol{x}^T \boldsymbol{x})}) = -lse(1, \boldsymbol{x}^T \boldsymbol{x}) \tag{۱۱-۱۹۱}$$

با توجه به اینکه مقدار متغیرهای واحدهای ساختار را مقادیری حقیقی در نظر گرفته ایم ممکن است وزن های ساختار در صورت وجود مقادیر برای تابع انرژی رابطه می توانیم یک جمله برای تنظیم وزن ها مشابه آنچه در تابع هزینه شبکه های عصبی استفاده می کردیم اضافه کنیم تا از رشد بیش از حد وزن ها جلوگیری کنیم، بدین ترتیب رابطه ۱۹۱-۱۱ به صورت رابطه ۱۹۲-۱۱ خواهد بود.

$$E_{\theta(\boldsymbol{x})} = -lse(1, \boldsymbol{x}^T \boldsymbol{x}) + \frac{1}{2}\boldsymbol{x}^T \boldsymbol{x} \tag{۱۱-۱۹۲}$$

با توجه به قضیه رویه محدب-مقعر، CCCP[1]، می توانیم تابع انرژی رابطه ۱۹۲-۱۱ مجموع دو تابع محدب E_1 و مقعر E_2 مطابق رابطه ۱۹۳-۱۱ است.

$$E_{\theta(\boldsymbol{x})} = \underbrace{-lse(1, \boldsymbol{x}^T \boldsymbol{x})}_{E_1} + \underbrace{\frac{1}{2}\boldsymbol{x}^T \boldsymbol{x}}_{E_2} \tag{۱۱-۱۹۳}$$

برای یک تابع متشکل از دو تابع مقعر و محدب نقطه بهینه سراسری، همان نقطه زینی[2] است که برای دستیابی به آن نقطه به صورت گام به گام[3] با توجه به قضیه CCCP گرادیان تابع محدب گام آینده برابر قرینه گرادیان تابع مقعر در گام فعلی است. بنابراین اگر فرض کنیم تابع انرژی رابطه در نقطه بهینه خود باشد رابطه ۱۹۴-۱۱ را داریم:

$$\nabla E_{1(k+1)} = -\nabla E_{2(k)} \tag{۱۱-۱۹۴}$$

با توجه به روابط ۱۹۵-۱۱ و ۱۹۶-۱۱ رابطه ۱۹۴-۱۱ به صورت رابطه ۱۹۷-۱۱ خواهد بود، این رابطه بیانگر فرم کلی یک ساختار هاپفیلد مدرن است.

$$\nabla lse(\beta, \mathbf{x}) = \mathbf{x}\,\mathrm{softmax}(\beta\mathbf{x}) \tag{۱۱-۱۹۵}$$

$$\mathrm{softmax}(\mathbf{x})_i = \frac{e^{x_i}}{\sum_{j=1}^{n} e^{x_j}}, n = size(\mathbf{x}) \tag{۱۱-۱۹۶}$$

$$\mathbf{x}_{k+1} = \mathbf{x}_k\,\mathrm{softmax}(\beta\mathbf{x}^T\mathbf{x}) \tag{۱۱-۱۹۷}$$

---

[1] The concave-convex procedure (CCCP):این قضیه در قسمت پیوست معرفی شده است

[2] Saddle point

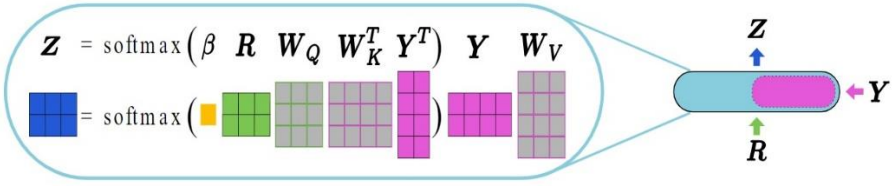
[3] Itterative

رابطه ۱۹۷-۱۱ معادل رابطه ۱۸-۱۱ به روزرسانی همگام شبکه هاپفیلد است. می توان چنین برداشت کرد که ساختار شبکه هاپفیلد مدرن الگوی بازیابی را به ازای الگوی تخریب شده به صورت یک میانگین وزنی نمائی، بیشینه هموار[1]، از الگوهای ذخیره شده محاسبه می کند در این روند برای تبدیل ابعاد مجموعه الگوهای ذخیره شده به ورودی وزن هایی استفاده شده است که همان پارامترهای آموزشی ساختار هستند.

۱۱.۱۳.۱ شبکه هاپفیلد مدرن در ساختارهای عمیق

می توانیم ساختار شبکه هاپفیلد مدرن را به صورت مجزا به عنوان یک ساختار یک لایه و یا به صورت یک لایه ادغام شده در ساختار عمیق استفاده کرد که حالت های ذیل آن را بررسی می کنیم.

- **ساختار هاپفید مدرن با دو ورودی:** این ساختار برای کاربردهایی مناسب است که در روند پیشرو شبکه دو ورودی مجزا وارد ساختار می شود، مانند ساختار واحد LSTM. این ساختار مشابه شکل ۴۶-۱۱ تعریف می شود که در آن Y, R به ترتیب ماتریس ورودی و حافظه و ماتریس های W وزن های آموزشی و β یک اسکالر است که به عنوان پارامتر کنترلی عمل می کند. ماتریس حافظه ممکن است دارای یک دینامیک زمانی (دارای یک ستون) و یا متشکل از چند گام زمانی گذشته باشد همچنین می توان این ساختار را جداگانه به صورت یک ساختار مجزا برای بازیابی الگوهای ذخیره شده مشابه ساختار هاپفیلد مرسوم استفاده کرد در این حالت هر یک از ستون های ماتریس Y همان الگوهای ذخیره شده و R الگوی تخریب شده (دلخواه) ورودی است.

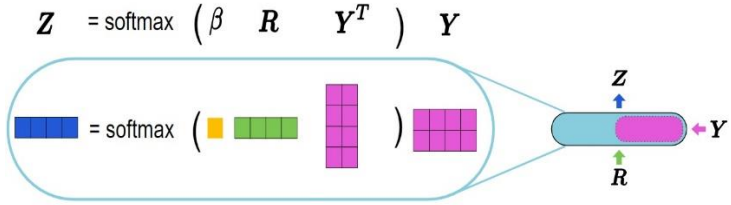


شکل ۴۶-۱۱: ساختار هاپفیلد مدرن[۳۶]

- **لایه هاپفیلد مدرن:** اگر ساختار شکل ۴۶-۱۱ را به عنوان یک لایه در نظر بگیریم می توانیم برای عملکرد بهتر گرادیان های آموزشی را علاوه بر ماتریس وزن های W برای ماتریس حافظه Y نیز تعریف کنیم در این صورت می توان ساختار شکل ۴۶-۱۱ را ساده سازی کرده و به صورت شکل ۴۷-۱۱ نشان دهیم. در شکل ۴۷-۱۱ ماتریس وزن های آموزشی و بردار حافظه با هم ادغام شده و روابط آموزشی صرفا

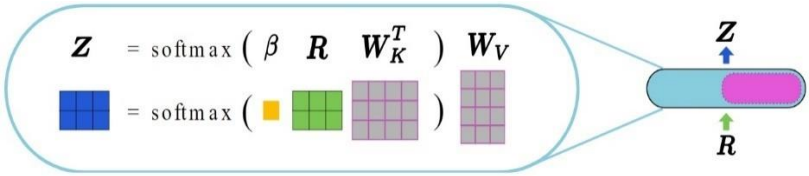
[1] Softmax

برای بردار حافظه تعریف می شود.



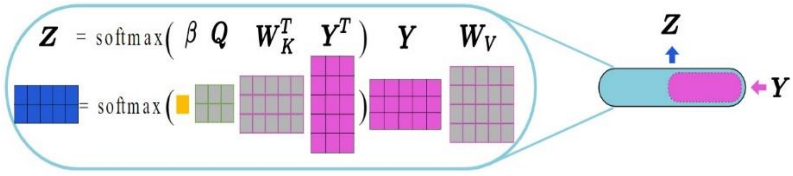
شکل ۴۷-۱۱: ساختار لایه هاپفیلد[۳۶]

- **لایه هاپفیلد مدرن با یک ورودی:** در این حالت ساختار به عنوان یک لایه مرسوم همانند لایه های پرسپترون در یک ساختار عمیق ادغام شده و تنها دارای یک ورودی است، با توجه به محدودیت ورودی در این حالت تعداد ماتریس های وزن نیز کاهش پیدا کند. شکل ۴۸-۱۱ ساختار لایه هاپفیلد با یک ورودی را نشان می دهد.



شکل ۴۸-۱۱: لایه هاپفیلد با یک ورودی[۳۶]

- **لایه هاپفیلد تجمیع کننده[1]:** از ساختار هاپفیلد مدرن می توان به عنوان یک روش کاهش بعد نیز استفاده کرد، در این صورت لایه هاپفیلد بر ماتریس ورودی Y نگاشت هایی اعمال و آن را به صورت ماتریس Z با ابعادکوچک تر به عنوان ماتریس خروجی در می آورد. شکل ۴۹-۱۱ ساختار لایه هاپفیلد تجمیع کننده را نشان می دهد. در این روش تجمیع بر خلاف روش تجمیع پیچشی که در بخش ۳-۹-۱۱ بررسی کردیم برای لایه تجمیع پارامترهای (ماتریس های W و Q) آموزشی تعریف می شود.



شکل ۴۹-۱۱: لایه هاپفیلد تجمیع کننده[۳۶]

[1] Pooling Hopfield layer

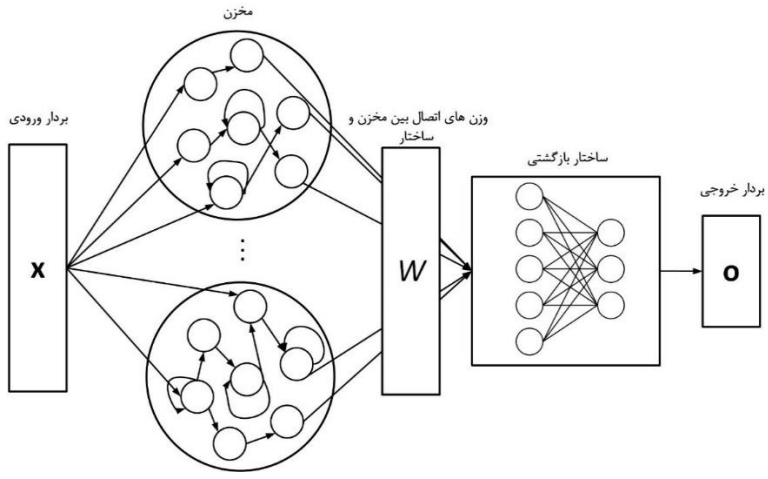
با توجه به ساختارهای معرفی شده گرادیان های آموزشی از خروجی را مطابق رابطه ۱۹۸-۱۱ برای ماتریس Y در ساختار شکل ۴۶-۱۱ تعریف می کنیم.

$$\frac{\partial Z}{\partial Y} = \text{softmax}(\beta RY^T)[1] \times W_v + \text{J}(\text{softmax}(\beta RY^T))Y \times W_v \quad (۱۱-۱۹۸)$$

در رابطه ۱۹۸-۱۱، $J_{(.)}$ ژاکوبین و [1] یک ماتریس با ابعاد ماتریس Y است که مقدار همه درایه های آن برابر یک است. برای سایز پارامترهای آموزشی ساختار نیز روابط مشابهی می توان تعریف کرد.

۱۱.۱۴ رایانش مخزنی[1]

یکی از راهکارهای ارائه شده برای کاهش از پیچیدگی محاسباتی ساختارهای بازگشتی استفاده از روش رایانش مخزنی است، در این روش یک یا چند مخزن با اتصالات تنک که دارای وزن ها و پارامترهای ثابت تصادفی است مطابق شکل ۵۰-۱۱ تعریف شده و خروجی مخزن ها به عنوان ورودی یک ساختار بازگشتی مشابه ساختارها معرفی شده در این فصل در نظر گرفته می شود، در این حالت بردار ورودی اصلی ساختار به جای ورودی مستقیم به ساختار بازگشتی وارد مخزن می شود. به بیانی دیگر در رایانش مخزنی به جای آنکه مستقیما بردار ورودی در ساختار بررسی شود، تاثیر آن بر خروجی مخزن(ها) بررسی می شود. برای اینکه روند این ساختار مشابه روند پیشرو و آموزش شبکه های مرسوم باشد اتصالات بین دو مخزن، اتصالات یک طرفه از مخزن به سمت ورودی و اتصالات یک طرفه از ساختار انتهائی (بازگشتی) به سمت مخزن حذف می شوند.



شکل ۵۰-۱۱: ساختار رایاتش مخزنی

[1] Reservoir computing

۱۱.۱۵ مسائل

۱. ظرفیت بازیابی بدون خطا و با خطا را برای یک شبکه هاپفیلد به ازای ابعاد مختلف ورودی بررسی کرده و مشخص کنید در هر بازه مقدار کدام یک از دو ظرفیت بدون خطا و با خطا بیشتر، کمتر یا با دیگری برابر است. سپس نتایج حاصل را با توجه به ساختار تحلیل کنید.

۲. روابط آموزشی بایاس را برای ساختارهای دروازه ای معرفی شده در این فصل توسعه دهید.

۳. ساختارهای دروازه ای معرفی شده در این فصل به صورت مجزا معرفی شده اند، روابط آن ها را برای یک لایه متشکل از آن ها بنویسید.

۴. تاثیر استفاده از هم پوشانی در لایه های تجمیع ساختارهای پیچشی را بررسی کنید

۵. برای یک مجموعه دادگان ثابت و یک ساختار بازگشتی ثابت، خروجی دو حالت با و بدون استفاده از رایانش مخزنی را مقایسه کنید.

فصل ۱۲
شبکه های عصبی پیچشی[1]

۱۲.۱ مقدمه

در اکثر ساختارهای ژرفی که تا کنون بررسی کرده ایم، معمولا یک بردار به عنوان ورودی ساختار در نظر گرفته می شد. حتی در مواردی که ورودی مدل ساختاری غیر برداری بود، قبل از ورود به مدل تبدیل به بردار می شد، مانند ذخیره تصویر در شبکه هاپفیلد. در حالی که امروزه یکی از عمده کاربردهای شبکه های عمیق پردازش دادگان تصویری است. در فصل ۵ اشاره کردیم که یک تصویر دیجیتال به صورت یک ماتریس، تصاویر سیاه و سفید، و یا یک تانسور سه بعدی، تصاویر رنگی، تعریف می شود. هر یک از درایه های این تانسور که به عنوان پیکسل[2] شناخته می شود می توانند همانند بردارهای ورودی مقادیری، حقیقی، داشته باشد. در واقع تصویر یک سیگنال است که برخلاف سری های زمانی که در بعد زمان تعریف می شد در ابعاد فضائی، طول، عرض و عمق، تعریف می شود. با این توضیحات لزوم توسعه و تعریف ساختارهایی که ورودی آن ها به شکل تانسور بوده و قابلیت پردازش دادگان تصویری را داشته باشند قابل توجیه است. گاهی نیاز است توالی تصاویر در کنار هم، ویدئو، مشابه سری های زمانی پردازش شوند از این رو این ساختارها را باید بتوان به صورت ساختارهای حافظه دار و بازگشتی نیز تعریف کرد.

ساختارهای پیچشی مرسوم ترین و مهم ترین ساختارهای توسعه داده شده برای دادگان تصویری هستند. در این فصل ابتدا به تفاوت ساختارهای پیچشی با سایر ساختارها و لزوم استفاده از آن ها در دادگان تصویری را بررسی می کنیم، سپس عملیات های مربوط به مراحل پیشرو و آموزش ساختارهای پیچشی را معرفی کرده و در ادامه با طرح انواع مختلف ساختارهای پیچشی توسعه داده شده کارکردهای مختلف یادگیری عمیق در زمینه پردازش دادگان تصویری را بررسی می کنیم.

۱۲.۲ لزوم استفاده از ساختارهای پیچشی

در برخی از ساختارهایی که تا کنون بررسی کرده ایم در صورتی که ورودی ساختار به صورت

---

[1] Convolutional neural networks (CNNs)
[2] Pixel

ماتریس، تصویر، تعریف می شد؛ مانند ذخیره و بازیابی تصاویر در ساختار شبکه هاپفیلد که در فصل ۱۱معرفی کردیم، مقادیر ابعاد مختلف ورودی را به صورت خطی کنار هم قرار داده و آن را تبدیل به یک بردار می کردیم تا ابعاد آن برای ساختار های مرسومی مانند شبکه های عصبی پرسپترون مناسب باشد. این روش اگرچه برای تصاویری با ابعاد، تعداد پیکسل، کوچک می تواند کار آمد باشد ولی برای تصاویر با وضوح، تعداد پیکسل، بالاتر که امروزه استفاده از آن ها بسیار مرسوم است با توجه به ابعاد بردار خروجی متناظر تولید شده کاربردی ندارد. یک تصویر رنگی با وضوح Full-HD[1] که امروزه یک وضوح استاندارد مرسوم محسوب می شود را در نظر بگیرید، این تصویر در رایانه به صورت یک تانسور سه بعدی با ابعاد$3 \times 1920 \times 1080$ ذخیره می شود، اگر بخواهیم این تصویر را به یک بردار تبدیل کنیم اندازه آن برابر $1080 \times 1920 \times 3 = 6220800$ خواهد بود، در نظر گرفتن چنین برداری به عنوان ورودی مستلزم تعریف یک ساختار با تعداد لایه و نورون بالائی است. در روند پیاده سازی و آموزش چنین ساختاری انواع مختلف مشکلات و چالش ها مانند پدیده محو شدگی گرادیان ها[2]، بیش برازش[3] و... که پیش از این به آن ها پرداخته ایم، بروز پیدا کنند.

علاوه بر مشکلات ناشی از بعد بالای ورودی ساختار مسئله دیگر در مورد دادگان تصویری وابستگی فضائی[4] پیکسل های دادگان تصویری به یکدیگر است. همان طور که در فصل ۱۱ بررسی کردیم ابعاد مختلف هر یک از نمونه بردار های مجموعه دادگان استخراج شده از یک سری زمانی، سیگنال، به هم وابستگی دارند، به همین دلیل ساختارهای بازگشتی و حافظه داری را معرفی کردیم تا با استفاده از آن ها این وابستگی را مدل سازی کنیم. همان طور که پیش تر اشاره کردیم یک تصویر، سیگنالی است که در ابعاد فضائی تعریف می شود (همانند سیگنال سری زمانی که در بعد زمان تعریف می شود) از این رو بین ابعاد، پیکسل، مختلف یک نمونه، تصویر، نیز این وابستگی وجود داشته و هر یک از این پیکسل ها در موقعیت مناسب خود در کنار هم می توانند اطلاعات مناسبی را فراهم کنند. برای درک بهتر این مطلب تصویری از یک جسم را در نظر بگیرید، اطلاعات مربوط به این جسم تعدادی از پیکسل های یک تصویر را اشغال کرده است و هر یک از این پیکسل ها در موقعیت خود (از نظر طول و عرض تصویر) نمایان گر بخشی از اطلاعات مربوط به تصویر، موقعیت نسبی جسم، رنگ، شدت نور و...، هستند. فرض کنیم موقعیت هر یک پیکسل ها را در تصویر به صورت تصادفی تغییر داده و آن ها را با هم جا به جا کنیم، در این صورت تصویر حاصل اگرچه از همان پیکسل های حاوی اطلاعات جسم تشکیل شده اند ولی نمایش همان جسم اولیه نبوده و اظلاعات مناسبی از آن ارائه نمی کند. از این رو نمی توان این تصویر را معادل همان تصویر جسم مذکور در نظر گرفت. این موضوع در حالتی که همه پیکسل ها را به صورت یک بردار کنار

[1] Full High-Definition
[2] Gradient vanishing
[3] Overfitting
[4] Spatial correlation

هم قرار دهیم چالش برانگیز تر بوده و بردار حاصل نمایش مناسبی از اطلاعات تصویر اولیه نخواهد بود. بنابراین توضیحات برای رفع چنین چالش هایی باید ساختارهایی را تعریف کنیم که ورودی را مستقیما به صورت ماتریس (تانسور) پردازش کنند. در فصل ۱۱ به ساختارهای پیچشی یک بعدی که برای پردازش دادگان سری زمانی اشاره کردیم و بررسی کردیم که این ساختارها به خوبی می توانند وابستگی زمانی بین ابعاد بردار ورودی را در محاسبات دخیل کنند، با توجه به این ویژگی ساختارهای پیچشی با تعمیم عملیات آن ها به ابعاد بالاتر می توانیم از آن ها در پردازش ورودی های تصویری نیز استفاده کرده و وابستگی بین ابعاد (پیکسل های) این دادگان را لحاظ کنیم. در ادامه به معرفی این ساختار های پیچشی می پردازیم.

۱۲.۳ ضرب پیچشی دو بعدی

در یک عملیات ضرب پیچشی گسسته دو بعدی روی یک ماتریس ورودی (تصویر با عمق یک- تصویر سیاه و سفید) مطابق رابطه ۱-۱۲، یک کرنل[1] دو بعدی به صورت ماتریس مطابق رابطه ۲-۱۲ تعریف می کنیم. اعمال ضرب پیچشی دو بعدی بر ماتریس ورودی با استفاده از کرنل رابطه ۲-۱۲ طبق مطالب فصل ۵ معادل اعمال فیلتر بر ماتریس ورودی است؛ ابعاد و مقادیر درایه های کرنل در ضرب پیچشی ویژگی های فیلتر و متعاقبا اعمال آن با گام حرکتی[2] معین به ماتریس ورودی در نهایت یک خروجی متناسب با پارامترهای اعمال شده را در پی خواهد داشت. برخی از این کرنل ها که مطالب مربوط به آن را در فصل ۵ معرفی کردیم برای استخراج لبه[3] های تصویر، کاهش نویز[4] به کار برده می شوند. در این کرنل ها که هر یک برای کاربردی خاص استفاده می شوند مقادیر درایه ها و ابعاد کرنل با توجه به هدف استفاده از آن پیش از اعمال به تصویر ورودی به صورت دستی[5] تعیین می شوند. اما همان طور که در فصل ۱۱ کرنل ضرب پیچشی یک بعدی را به عنوان پارامترهای آموزشی در نظر گرفته و مقادیر آن را با استفاده از روابط مشتقات زنجیره ای آموزش دادیم، در ضرب پیچشی دو بعدی نیز می توانیم یک مدل متشکل از کرنل های پیچشی دو بعدی (و اجزای دیگر) تشکیل داده و مقادیر این کرنل ها را آموزش دهیم، در این صورت لزومی به تعیین مقادیر این کرنل ها به صورت دستی نبوده و صرفا برای آن ها مقادیر اولیه ای با استفاده از یکی از روش هایی، مقادیر تصادفی، He، Xavier و...، که برای مقدار دهی اولیه به پارامترهای مدل های مرسوم مانند شبکه های عصبی پرسپترون استفاده می کردیم، در نظر می گیریم. سپس با تعریف یک روش آموزش با سرپرست[6] و محاسبه مشتقات زنجیره ای در هر گام آموزشی[7] این مقادیر را آموزش می دهیم. در ادامه روابط پیشرو و آموزش برای کرنل ضرب پیچشی و سایر اجزای

[1] Kernel
[2] Stride
[3] Edge
[4] Noise reduction
[5] Manual
[6] Supervised
[7] Epoch

ساختار پیچشی بررسی می کنیم.

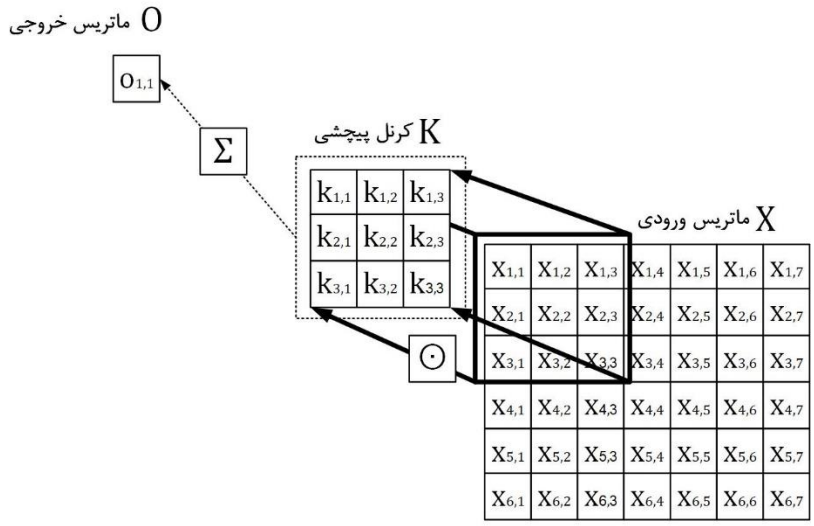
$$X = \begin{bmatrix} x_{1,1} & x_{1,2} & \cdots & x_{1,n} \\ x_{2,1} & \ddots & \cdots & \vdots \\ \vdots & \cdots & \ddots & \vdots \\ x_{m,1} & \cdots & \cdots & x_{m,n} \end{bmatrix}_{m \times n} \quad (۱۲-۱)$$

$$K = \begin{bmatrix} k_{1,1} & k_{1,2} & \cdots & k_{1,h} \\ k_{2,1} & \ddots & \cdots & \vdots \\ \vdots & \cdots & \ddots & \vdots \\ k_{g,1} & \cdots & \cdots & k_{g,h} \end{bmatrix}_{g \times h} \quad (۱۲-۲)$$

۱۲.۳.۱ روابط پیشرو ضرب پیچشی دو بعدی

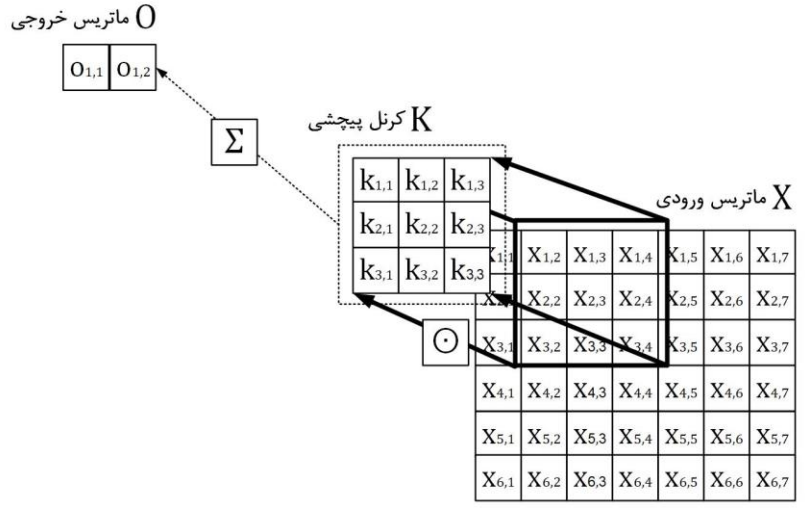
برای عملیات ضـرب پیچشـی دو بعدی با فرض ماتریس ورودی $n \times m$ و کرنل $g \times h$ مطابق روابط ۱-۱۲ و ۲-۱۲، اندازه هر یک از ابعاد کرنل باید از بعد متناظر آن در ماتریس ورودی کوچکتر یا مسـاوی باشـد، $h \leq m$ ،$g \leq n$، ابتدا دو مقدار اسـکالر طبیعی s_2, s_1 به عنوان گام حرکتی، در جهت افقی و عمودی، تعریف می کنیم، این گام های حرکتی، مقدار حرکت کرنل بر روی تصـویر ورودی را در هر گام مشـخص می کند. در ابتدای عملیات ضـرب پیچشـی مطابق شـکل ۱-۱۲ ابتدا کرنل طوری رو به روی تصـویر، ماتریس، ورودی قرار می گیرد که ابعاد ابتدایی آن را پوشـش دهد، سـمت راسـت بالا. سـپس اولین درایه از ماتریس خروجی با محاسـبه مجموع درایه های حاصـل از ضرب درایه به درایه، هادامارد[1]، محاسبه می شود.

[1] Hadamard's product



شکل ۱-۱۲: آغاز روند ضرب پیچشی دو بعدی

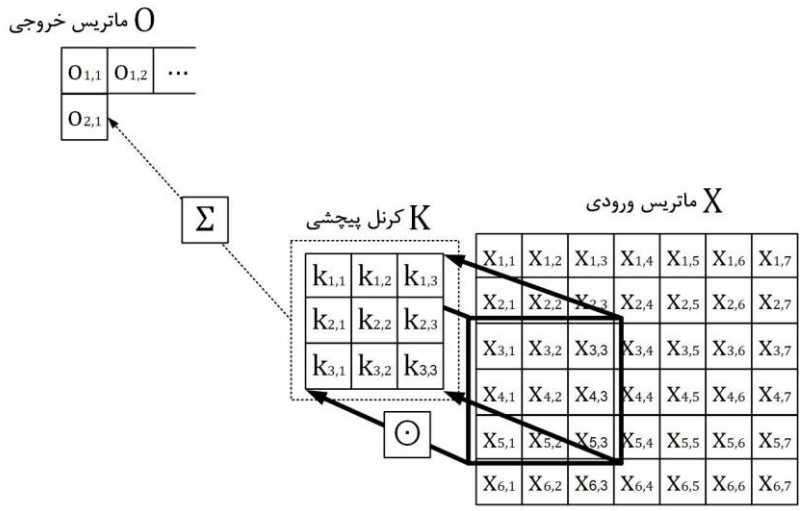
برای محاسبه مقدار پیکسل ها در جهت افقی، درایه های یک سطر از ماتریس، در مرحله بعد کرنل به اندازه گام حرکتی افقی، یک، به جلو حرکت کرده و مطابق شکل ۲-۱۲ رو به روی ابعاد دیگری از ماتریس ورودی قرار گرفته و درایه دیگری از ماتریس خروجی محاسبه می شود.



شکل ۲-۱۲: ادامه روند ضرب پیچشی دو بعدی در جهت افقی

برای محاسبه پیکسل ها در جهت عمودی، درایه های یک ستون، نیز کرنل به اندازه گام حرکتی عمودی، دو، بر روی تصویر ورودی در جهت عمودی مطابق شکل ۳-۱۲ جا به جا شده و

مقدار درایه خروجی محاسبه می شود.



شکل ۳-۱۲: ادامه روند ضرب پیچشی دو بعدی در جهت عمودی

با توجه به توضیحات فوق رابطه پیشرو ۳-۱۲ را برای محاسبه مقادیر درایه های ماتریس خروجی در عملیات ضرب پیچشی دو بعدی تعریف می کنیم.

(۱۲-۳)

$$o_{i,j} = \sum_{p=0}^{g-1}\sum_{t=0}^{h-1} x_{s_1.i+p,s_2.j+t} k_{p+1,t+1} = sum(X_{(s_1.i:s_1.i+g-1,s_2.j:s_2.j+h-1)} \odot K)$$

$$i = 1,...,l_1,\ j = 1,...,l_2$$

در رابطه ۳-۱۲ برای محاسبه طول l_1 و عرض l_2، بعد افقی و عمودی، ماتریس خروجی به ترتیب روابط ۴-۱۲ و ۵-۱۲ را تعریف می کنیم. در این روابط h,g,m,n به ترتیب ابعاد ماتریس ورودی و کرنل، p_1 و p_2 به ترتیب ابعاد بعدگذاری در دو جهت افقی و عمودی که در ادامه رابطه محاسبه آن را معرفی خواهیم کرد و s_1 و s_2 به ترتیب اندازه گام حرکتی افقی و عمودی است. در صورتی که مقدار هر یک از روابط ۴-۱۲ و ۵-۱۲ مقدار صحیحی نباشد، مقدار آن ها را اولین عدد صحیح بزرگ تر از مقدار حاصل رابطه، Ceil، در نظر می گیریم، اما در حالت کلی اندازه پارامترهایی مانند بعدگذاری، گام حرکتی و.. را به گونه در نظر می گیریم که حاصل این روابط مقداری صحیح باشد.

$$l_1 = \frac{n-g+p_1}{s_1} + 1$$

(۱۲-۴)

$$l_2 = \frac{m - h + p_2}{s_2} + 1 \qquad (۱۲-۵)$$

استفاده از مقادیر گام حرکتی بزرگ تر حجم محاسبات لازم برای عملیات را کاهش می دهد ولی در عوض باعث حذف بخشی از اطلاعات که در ماتریس خروجی می شود، ابعاد ماتریس خروجی کاهش می یابد. معمولا برای گام حرکتی افقی و عمودی مقادیر برابر در نظر می گیرند تا در محاسبه مقادیر درایه های ماتریس خروجی، مقادیر ماتریس ورودی به طور یکنواخت در هر دو جهت دخیل باشد. همچنین در ساختار های پیچشی به ماتریس خروجی، نگاشت ویژگی[1] نیز اطلاق می شود. معمولا در ساختارهای پیچشی دو بعدی پس از محاسبه خروجی حاصل از رابطه یک تابع فعال ساز مطابق رابطه ۶-۱۲ به هریک از درایه های آن به عنوان نگاشت غیر خطی اعمال شده و نگاشت ویژگی نهائی، خروجی تابع اعمال شده در نظر گرفته می شود. در صورت استفاده از ترم بایاس، بایاس به صورت یک ماتریس با ابعادی برابر با ماتریس خروجی که از روابط ۴-۱۲ و ۵-۱۲ محاسبه شده است با ماتریس خروجی رابطه ۳-۱۲، مطابق رابطه ۷-۱۲ پیش از اعمال تابع فعال ساز جمع می شود. در رابطه ۷-۱۲ نماد $\odot$ نشان دهنده عملیات ضرب پیچشی دو بعدی دو ماتریس ورودی X و کرنل K، B ماتریس بایاس و f تابع فعال ساز است.

$$o'_{i,j} = f(o_{i,j}) \qquad (۱۲-۶)$$

$$O'_{l_1 \times l_2} = f(X_{m \times n} * K_{g \times h} + B_{l_1 \times l_2}) = f(O_{l_1 \times l_2} + B_{l_1 \times l_2}) \qquad (۱۲-۷)$$

در رابطه ۷-۱۲ منظور از نماد * عملیات ضرب پیچشی بین ماتریس ورودی و کرنل است.

۱۲.۳.۲ توابع فعال ساز در ساختارهای پیچشی دو بعدی

در مطالب فوق اشاره کردیم که به هر یک از ابعاد ماتریس خروجی رابطه ۳-۱۲ یک تابع فعال ساز اعمال می شود. همانند سایر ساختارها می توانیم از روابط غیرخطی مختلفی مانند سیگموئید، تانژانت هیپربولیک و... به عنوان تابع فعال ساز استفاده کنیم اما تابعی که به طور گسترده به عنوان تابع فعال ساز در ساختارهای پیچشی استفاده می شود تابع ReLU[2] است که رابطه آن مطابق رابطه ۸-۱۲ است.

$$f(x) = \mathrm{ReLU}(x) = \begin{cases} x, & if \quad x \geq 0 \\ 0, & if \quad x < 0 \end{cases} \qquad (۱۲-۸)$$

[1] Feature map
[2] Rectified linear unit

دلایل عمده ارجحیت تابع ReLU به سایر توابع فعال ساز بدین شرح است:

- در فصل ۱۰ در مبحث تنک زائی در خودرمزگذارها بررسی کردیم که هدف از اعمال جمله تنک زائی به تابع هزینه این است که هر یک نورون های لایه رمزگذار به ازای تعداد محدودی از ورودی ها فعال شده و به هر نمونه، نمونه های هر کلاس، از دادگان بردار پنهانی نسبت دهند که تا حد ممکن متمایزتر از بردارهای پنهان مربوط به سایر نمونه ها، سایر کلاس ها، باشند. این عامل باعث ایجاد تفکیک پذیری بهتر و در نهایت آموزش بهتر برای ساختار می شد. ساختارهای پیچشی نیز به نوعی یک ساختار بازنمائی با آموزش با سرپرست محسوب می شوند و چنان که در ادامه بررسی می کنیم ساختار متشکل از لایه های مختلف پیچشی در نهایت یک بردار از تصویر ورودی را به عنوان بردار بازنمائی تولید کرده و بردار بازنمائی حاصل به عنوان ورودی یک مدل مرسوم مانند شبکه عصبی چند لایه تولید استفاده می شود. بنابراین لزوم تنک زائی در ساختارهای پیچشی نیز احساس می شود. به طوری که در ساختار به ازای ورودی های مختلف، ورودی ها کلاس های مختلف از تصاویر، فقط تعدادی از توابع فعال ساز، فعال شده و بردار های خروجی نهائی برای نمونه های کلاس های مختلف حدالامکان از هم متمایز شوند. این تنک زائی در ساختار های پیچشی از روش اضافه کردن جمله تنک زائی به تابع هزینه محاسبه نمی شوند بلکه رابطه توابع ReLU خود چنین ویژگی را به ساختار اضافه می کند بدین ترتیب که اعمال یک کرنل پیچشی دو بعدی به تصاویر مربوط به هر کلاس با توجه به ویژگی تصویر و مقادیر پیکسل های آن در نهایت مقادیر مختلفی را در هر یک از درایه های خروجی رابطه ۳-۱۲ ایجاد می کند، برخی مقادیر در بازه منفی و برخی مثبت، این درایه ها در صورت اعمال تابع ReLU به ازای مقادیر مثبت تابع را فعال و به ازای مقادیر منفی تابع را غیر فعال می کنند و در نهایت یک نگاشت ویژگی با الگویی تنک متشکل از درایه های صفر و مثبت ایجاد می شود که با الگوی تصاویر دیگر متفاوت است. بدین ترتیب به هر نمونه از دادگان یک الگوی تنک منحصر به فرد نسبت داده می شود که نسبت به الگوی حاصل از توابع فعال ساز مرسوم دیگر تمایز بین این الگوهای نمونه های مختلف بیشتر مشهود است.

- با توجه به ابعاد بزرگ ورودی در ساختارهای پیچشی معمولا برای این ساختارها تعداد لایه های زیادی تعریف می شود. این امر موجب تشدید پدیده محو

شدگی گرادیان ها[1] می شود. استفاده از تابع فعال ساز ReLU در عین تنک زائی، پدیده محو شدگی گرادیان ها را نیز تا حدود زیادی برطرف کرده و امکان تعریف تعداد لایه بیشتری را برای ساختار فراهم می کند؛ بدین ترتیب که مشتق تابع به ازای ورودی های که آن را فعال می کنند، ورودی های مثبت، همواره مقداری برابر یک و در صورت غیرفعال بودن برابر صفر خواهد بود. با این تفاسیر ویژگی تنک زائی در روابط آموزشی نیز اعمال شده و گرادیان های آموزشی فقط مقادیر پارامترهایی که به ازای آن ها تابع فعال ساز در آن گام، فعال می شود را به روز رسانی می کنند، این ویژگی تا حدودی یادآور روش حذف تصادفی[2] نیز می باشد که در آن به طور تصادفی در روند آموزش ساختار برخی از نورون ها را برای جلوگیری از بیش برازش، معضلی که در ساختارهایی با تعداد پارامتر بالا مانند ساختارهای پیچشی شایع است، حذف می کردیم، مقدار خروجی نورون در روند پیشرو و مقدار گرادیان های آموزشی آن در مرحله پس انتشار خطا صفر در نظر می گرفتیم. با این تفاسیر می توان چنین برداشت کرد که استفاده از ReLU با توجه به وجود برخی مقادیر صفر در روند پیشرو و متعاقبا مقادیر گرادیان صفر برای آن ها، از بروز پدیده بیش برازش[3] نیز که با توجه به تعداد پارامترهای بالای ساختارهای پیچشی در آن ها بسیار شایع است تا حد زیادی جلوگیری می کند.

بنابراین به طور کلی مزایای استفاده از تابع ReLU، تنک زائی، جلو گیری از پدیده محو شدگی گرادیان ها و پدیده بیش برازش و همچنین حجم محاسباتی کمتر در مقایسه با توابعی چون تانژانت هیپربولیک است. در کنار این مزایا این تابع نقاط ضعفی نیز دارد. این تابع در نقطه صفر پیوسته نبوده تغییر با تغییرات جزئی ورودی در بازه اطراف نقطه صفر، که با فرض وجود نویز در دادگان بسیار محتمل است، مقدار خروجی تابع به میزان زیادی تغییر پیدا می کند، این عامل سبب ایجاد عدم قطعیت در روند آموزش ساختار می شود، همچنین تابع در این نقطه مشتق پذیر نیست. مشکل عمده دیگر تابع ReLU مسئله غیرفعال شدن آن[4] به ازای ورودی های منفی است که سبب می شود مقادیر مشتقات زنجیره ای که در محاسبه آن ها مشتق تابع به ازای ورودی منفی در آن ها دخیل است همواره صفر بوده و با وجود محاسبه این مشتقات پارامترهای مربوط به آن ها به روز رسانی نشده و حجم زیادی از محاسبات هدر رود. علاوه بر این موارد با توجه به رابطه خطی تابع به ازای ورودی های مثبت، این تابع نسبت به توابعی چون تانژانت هیپربولیک نگاشت غیرخطی کمتری بر ورودی اعمال می کند. برای مرتفع کردن این مشکلات گونه های مختلفی برای این تابع

[1] Gradient vanishing
[2] Drop-out
[3] Over fitting
[4] Dying

ارائه شده اند که به بررسی چند مورد از آن ها می پردازیم:

۱- **Leaky ReLU:** رابطه این تابع مطابق رابطه ۹-۱۲ است. همان طور که در رابطه ۹-۱۲ مشخص شده است، در بازه اعداد منفی این تابع دارای یک نگاشت خطی با شیب محدود، ۰/۰۱، است.

$$f(x) = \text{LeakyReLU}(x) = \begin{cases} x, & if \quad x \geq 0 \\ 0.01.x, & if \quad x < 0 \end{cases} \tag{۹-۱۲}$$

۲- **PReLU**[1]: رابطه این تابع مطابق رابطه ۱۰-۱۲ بوده و برخلاف رابطه تابع Leaky ReLU مقدار شیب در بازه منفی ثابت نبوده و با استفاده از پارامتر α که یک اسکالر است تعیین می شود، به عبارتی تابع Leaky ReLU یک تابع PReLU با $\alpha = 0.01$ است. در مواردی مقدار این شیب برای هر ورودی به صورت تصادفی تعیین می شود که به آن تابع ReLU تصادفی[2] گفته می شود.

$$f(x) = \text{PReLU}(x) = \begin{cases} x, & if \quad x \geq 0 \\ \alpha x, & if \quad x < 0 \end{cases} \tag{۱۰-۱۲}$$

۳- **ELU**[3]: رابطه این تابع مطابق رابطه ۱۱-۱۲ بوده و برخلاف دو تابع فوق خروجی محدوده منفی یک رابطه غیرخطی دارد. در رابطه ۱۱-۱۲، α یک پارامتر اسکالر کنترلی برای تعیین محدوده خروجی بازه منفی است. این تابع به دلیل وجود پارامتر α که به عنوان پارامتر مقیاس در نظر گرفته می شود، با عنوان تابع SELU[4] نیز شناخته می شود.

$$f(x) = \text{ELU}(x) = \begin{cases} x, & if \quad x \geq 0 \\ \alpha(e^x - 1), & if \quad x < 0 \end{cases} \tag{۱۱-۱۲}$$

۴- **Silu**[5]: رابطه این تابع مطابق رابطه ۱۲-۱۲ است. این رابطه برخلاف روابط پیشین به صورت دو شرطی تعریف نمی شود و از این رو پیچیدگی

[1] Parametric ReLU
[2] Randomized ReLU
[3] Exponential linear unit
[4] Scaled exponential linear unit
[5] Sigmoid linear unit

محاسباتی کمتری دارد، همچنین در این تابع مشکل مربوط به نا پیوستگی نقطه صفر نیز برطرف شده است.

$$f(x) = \text{Silu}(x) = x.\text{sigmoid}(x) \qquad (۱۲-۱۲)$$

۵- **Soft-Plus:** این تابع نیز مشابه تابع Silu بوده و رابطه آن به صورت رابطه ۱۲-۱۳ تعریف می شود، اما بر خلاف تابع Silu خروجی این تابع در محدوده منفی صفر بوده و نسبت به تابع ReLU، تفاوت آن پیوستگی و مشتق پذیری در نقطه صفر است.

$$f(x) = \ln(1+e^{x}) \qquad (۱۲-۱۳)$$

۶- **CReLU**[1]: این تابع رابطه ای مطابق رابطه ۱۲-۱۴ دارد. در این تابع به ازای هر ورودی، دو خروجی با ابعاد برابر با بعد ورودی محاسبه شده و در کنار هم قرار می گیرند[2]. برای مثال اگر ورودی تابع یک ماتریس با ابعاد $m \times n$ باشد خروجی آن دو ماتریس با ابعاد $m \times n$ است که می توان آن ها را در راستایی که اندازه آن دو با هم برابر است کنار هم قرار داده و به صورت یک ماتریس با ابعاد $m \times 2n$ یا $2m \times n$ نشان دهیم و یا اینکه آن ها را به صورت یک تانسور سه بعدی با ابعاد $m \times n \times 2$ نشان دهیم. کنار هم قرار دادن خروجی این تابع به صورت تانسور در ساختارهای پیچشی مرسوم تر است.

$$f(x) = [\max(0, x), \max(0, -x)] \qquad (۱۲-۱۴)$$

۷- **ReLU6:** رابطه این تابع به صورت تابع ReLU، به صورت شرطی مطابق رابطه تعریف می شود. در این تابع مشکلات مربوط به تابع ReLU کماکان وجود داشته و تنها مزیت آن نسبت به تابع ReLU محدود بودن بازه خروجی آن در کران بالا به شش است که بنابر نتایج تجربی باعث بهبود تنک زائی و متعاقبا روند آموزش ساختار می شود، این محدودیت همچنین از ایجاد مقادیر خروجی بزرگ که باعث رشد بیش از حد گرادیان ها می شود نیز جلوگیری می کند.

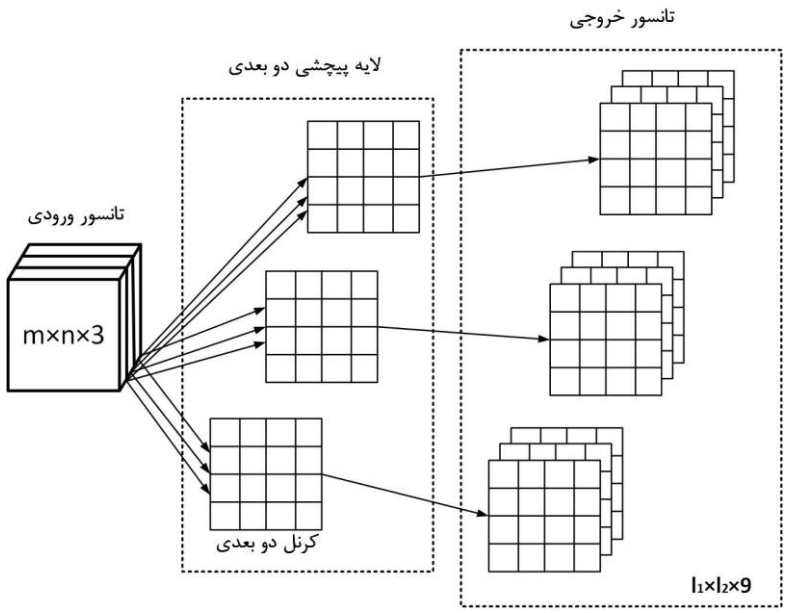
[1] Concatenated ReLU
[2] Concatenate

$$f(x) = \text{Relu6}(x) = \begin{cases} 6, & if \quad x > 6 \\ x, & if \quad 0 \le x \le 6 \\ 0, & if \quad x < 0 \end{cases} \tag{۱۲-۱۵}$$

۱۲.۳.۳ لایه پیچشی دو بعدی[1]

همان طور که در بررسی ساختار پیچشی یک بعدی در فصل ۱۱ اشاره کردیم یک عملیات ضرب پیچشی شامل یک ورودی و یک کرنل را می توان به صورت یک لایه پرسپترون متشکل از چند نورون با اتصالات محلی[2]، تنک، در ورودی و با وزن های مشترک[3] و برابر برای همه نورون ها در نظر گرفت. با این دیدگاه آموزش کرنل پیچشی و توسعه روابط آموزشی برای نیز قابل توجیه است. همچنین یک لایه پیچشی یک بعدی را لایه ای متشکل از چند کرنل پیچشی معرفی کردیم که به ازای هر یک از آن ها یک عملیات پیچشی برای ورودی تعریف شده و خروجی متناظر با آن محاسبه می شد، در نهایت خروجی این کرنل ها در کنار هم خروجی لایه را تشکیل می دادند. این تعریف را برای لایه پیچشی دو بعدی نیز تعمیم می دهیم. یک لایه پیچشی دو بعدی مطابق شکل ۱۲-۴ متشکل از تعدادی کرنل دو بعدی با ابعاد یکسان است که به ازای هر یک از آن ها یک ضرب پیچشی دو بعدی تعریف شده و در نهایت خروجی همه این کرنل ها در کنار هم قرار گرفته و تانسور خروجی نهائی لایه را تشکیل می دهند. در صورتی که ورودی لایه پیچشی از چند ماتریس تشکیل شده باشد، تانسور، مانند یک تصویر رنگی که از سه ماتریس هم اندازه تشکیل شده است، تانسور سه بعدی که اندازه بعد سوم آن سه است، هر یک از کرنل های لایه پیچشی در همه ماتریس های ورودی، در طول عمق تانسور ورودی، ضرب می شوند. به عنوان مثال ورودی اگر ورودی یک لایه پیچشی متشکل از سه ماتریس با ابعاد $m \times n$، یک تانسور با ابعاد $m \times n \times 3$، و لایه پیچشی متشکل از کرنل هایی با ابعاد $g \times h$ باشد در این صورت هر یک از کرنل های لایه جداگانه در هر ماتریس ورودی ضرب شده و ماتریس خروجی تولید می کنند، در نهایت با توجه به این که لایه دارای کرنل v است در خروجی لایه ماتریس، یک تانسور با عمق $3v$، تولید می شود. شکل ۱۲-۴ نمایش یک لایه پیچشی دو بعدی است.

[1] 2-D convolutional layer
[2] Locally connected
[3] Shared parameters



شکل ۴-۱۲: لایه پیچشی دو بعدی

۱۲.۳.۴ بعد گذاری[1]

با توجه به شکل های ۱-۱۲ تا ۳-۱۲ و روند محاسبه ماتریس خروجی حاصل از یک عملیات ضرب دو بعدی در روند پیشرو، مشاهده می کنیم که برخی درایه های ماتریس ورودی مانند درایه های چهار گوشه، کمتر در محاسبه مقادیر ماتریس خروجی دخیل بوده و کرنل پیچشی تعداد دفعات کمتری رو به روی آن ها قرار می گیرد. حتی در مواردی با توجه به ابعاد ورودی، کرنل و گام حرکتی ممکن است یکی از سطرها یا ستون های ماتریس ورودی در عملیات ضرب پیچشی کلا دخیل نشود، مانند سطر ششم ماتریس ورودی در تصاویر ۱-۱۲ تا ۳-۱۲. برای رفع این مسئله می توانیم در اطراف ماتریس ورودی با اضافه کردن ابعادی با مقادیر صفر[2] اطلاعات موجود در درایه های ذکر شده را بیش از پیش در محاسبه مقادیر درایه های ماتریس خروجی دخیل کنیم. بدین ترتیب در آغاز عملیات ضرب پیچشی کرنل در موقعیت شکل قرار گرفته و سپس در جهات افقی و عمودی حرکت می کند، با توجه به این که حاصل ضرب درایه به درایه، هادامارد، این ابعاد افزوده شده همواره صفر است، اضافه کردن آن ها به ماتریس ورودی خللی به محاسبات پیشرو طبق رابطه وارد نمی کند. تعداد بعد های صفر لازم در دو جهت افقی و عمودی به ترتیب از روابط ۱۶-۱۲ و ۱۷-۱۲محاسبه می شود. تعداد ابعاد لازم حاصل از روابط ۱۶-۱۲ و ۱۷-۱۲ به تعداد حاصل از رابطه ۱۸-۱۲ در سمت چپ و راست ماتریس ورودی و به تعداد حاصل از رابطه ۱۹-۱۲ در بالا و پایین ماتریس ورودی اضافه می شود. در صورتی که مقدار هر یک از روابط مقدار صحیحی نباشد،

[1] Padding
[2] Zero padding

مقدار آن ها را اولین عدد صحیح بزرگ تر از مقدار حاصل رابطه، Ceil، در نظر می گیریم.

$$p_1 = g - 1 \quad (۱۲-۱۶)$$

$$p_2 = h - 1 \quad (۱۲-۱۷)$$

$$p_1' = \frac{p_1}{2} \quad (۱۲-۱۸)$$

$$p_2' = \frac{p_2}{2} \quad (۱۲-۱۹)$$

با توجه به روابط ۴-۱۲ و ۵-۱۲ و تاثیر روش بعدگذاری در ابعاد خروجی، از این روش می توان به عنوان یک روش برای کنترل ابعاد خروجی استفاده کرد، در چنین کاربردی تعداد ابعاد لازم لزوما از روابط ۱۶-۱۲ و ۱۷-۱۲ محاسبه نمی شد و در هر یک از جهات افقی و عمودی به تعداد دلخواه بعد صفر اضافه کرد. علاوه بر افزایش تاثیر درایه های چهارگوشه در عملیات و کنترل ابعاد خروجی این روش کاربردهای دیگری نیز دارد که در ادامه به آن ها اشـــاره خواهیم کرد. شـــکل ۵-۱۲ بعد گذاری یک ماتریس را نشان می دهد که در آن $p_1 = 2$ و $p_2 = 4$ است.

0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0
0	$X_{1,1}$	$X_{1,2}$	$X_{1,3}$	$X_{1,4}$	$X_{1,5}$	$X_{1,6}$	$X_{1,7}$	0
0	$X_{2,1}$	$X_{2,2}$	$X_{2,3}$	$X_{2,4}$	$X_{2,5}$	$X_{2,6}$	$X_{2,7}$	0
0	$X_{3,1}$	$X_{3,2}$	$X_{3,3}$	$X_{3,4}$	$X_{3,5}$	$X_{3,6}$	$X_{3,7}$	0
0	$X_{4,1}$	$X_{4,2}$	$X_{4,3}$	$X_{4,4}$	$X_{4,5}$	$X_{4,6}$	$X_{4,7}$	0
0	$X_{5,1}$	$X_{5,2}$	$X_{5,3}$	$X_{5,4}$	$X_{5,5}$	$X_{5,6}$	$X_{5,7}$	0
0	$X_{6,1}$	$X_{6,2}$	$X_{6,3}$	$X_{6,4}$	$X_{6,5}$	$X_{6,6}$	$X_{6,7}$	0
0	$X_{7,1}$	$X_{7,2}$	$X_{7,3}$	$X_{7,4}$	$X_{7,5}$	$X_{7,6}$	$X_{7,7}$	0
0	$X_{8,1}$	$X_{8,2}$	$X_{8,3}$	$X_{8,4}$	$X_{8,5}$	$X_{8,6}$	$X_{8,7}$	0
0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0

شکل ۵-۱۲: بعد گذاری یک ماتریس

۱۲-۳-۵- روابط آموزشی برای ضرب پیچشی دو بعدی

برای محاسبه گرادیان های کرنل یک ضرب پیچشی دو بعدی برای سادگی محاسبات فرض می کنیم ساختار فقط از یک لایه شامل یک کرنل پیچشی تشکیل شده باشد ولی ورودی آن ماتریس با ابعاد باشد (یک تانسور با ابعاد) در این صورت کرنل در هر یک از ماتریس های ورودی جداگانه ضرب شده و یک تانسور با عمق در خروجی تشکیل خواهد داد. همچنین برای یک تابع هزینه مدل مجموع مربعات خطا در نظر می گیریم. با این توضیحات مشتقات زنجیره ای کرنل ضرب پیچشی دو بعدی از رابطه ۱۲-۲۰ محاسبه می شود. همچنین برای سادگی محسابات خطای $\boldsymbol{e}$ را به صورت یک ماتریس با ابعاد برابر با خروجی $\acute{o}$ فرض می کنیم.

$$\frac{\partial E}{\partial k_{r,q}} = \frac{\partial E}{\partial \mathbf{e}}_{e} \sum_{i=1}^{l_1}\sum_{j=1}^{l_2} \frac{\partial \mathbf{e}}{\partial o_{i,j}}_{-1} \frac{\partial o'_{i,j}}{\partial o_{i,j}}_{f'(o_{i,j})} \frac{\partial o_{i,j}}{\partial k_{r,q}} \qquad (۱۲-۲۰)$$

$$\frac{\partial o_{i,j}}{\partial k_{r,q}} = \frac{\partial}{\partial k_{r,q}} (sum(X_{(s_1.i:s_1.i+g-1,s_2.j:s_2.j+h-1)} \odot K)) \qquad (۱۲-۲۱)$$

$$\frac{\partial}{\partial k_{r,q}} (sum(X_{(s_1.i:s_1.i+g-1,s_2.j:s_2.j+h-1)} \odot K)) =$$

$$\frac{\partial}{\partial k_{r,q}} (k_{1,1}x_{s_1.i,s_2.j} + k_{1,2}x_{s_1.i,s_2.j+1} + k_{r,q}x_{s_1.i+r-1,s_2.j+q-1} + ...) = x_{s_1.i+r-1,s_2.j+q-1} \qquad (۱۲-۲۲)$$

با توجه به رابطه ۱۲-۲۰ مقادیر کرنل ضرب پیچشی در هر گام آموزشی مطابق رابطه ۱۲-۲۳ به روزرسانی می شود.

$$k_{r,q(k+1)} = k_{r,q(k)} - \eta \frac{\partial E}{\partial k_{r,q}}_{(k)} \qquad (۱۲-۲۳)$$

برای آموزش تانسور بایاس که پیش از اعمال تابع فعال ساز با تانسور خروجی ضرب پیچشی جمع می شود رابطه مشتقات زنجیره ای مطابق رابطه را داریم:

$$\frac{\partial E}{\partial B} = \frac{\partial E}{\partial \mathbf{e}}_{e} \frac{\partial \mathbf{e}}{\partial O'}_{-1} \frac{\partial O'}{\partial B}_{f'(O+B)} \qquad (۱۲-۲۴)$$

با توجه به رابطه ۱۲-۲۴ ماتریس بایاس در هر گام آموزشی مطابق رابطه ۱۲-۲۵ به روزرسانی می شود.

$$B_{(k+1)} = B_{(k)} - \eta \frac{\partial E}{\partial B_{(k)}} \tag{۱۲-۲۵}$$

اگر فرض کنیم لایه پیچشی به صورت سری بین لایه های یک ساختار قرار گرفته باشد در این صورت مشتق خروجی نسبت به ورودی لایه پیچشی را طبق رابطه ۱۲-۲۶ تعریف می کنیم تا با ادامه جملات این رابطه پارامترهای لایه های پیش از آن را نیز آموزش دهیم.

$$\underset{e}{\frac{\partial E}{\partial \mathbf{e}}} \sum_{i=1}^{l_1} \sum_{j=1}^{l_2} \underset{-1}{\frac{\partial \mathbf{e}}{\partial o_{i,j}}} \underset{f'(o_{i,j})}{\frac{\partial o'_{i,j}}{\partial o_{i,j}}} \frac{\partial o_{i,j}}{\partial x_{r,q}} \frac{\partial x_{r,q}}{\dots} \dots \tag{۱۲-۲۶}$$

$$\frac{\partial o_{i,j}}{\partial x_{r,q}} = \frac{\partial}{\partial x_{r,q}} (sum(X_{(i:i+g-1,\, j:j+h-1)} \odot K)) \tag{۱۲-۲۷}$$

$$\frac{\partial}{\partial x_{r,q}} (k_{1,1} x_{i,j} + k_{1,2} x_{i,j+1} + k_{s,d} x_{r,q} + \dots) = \begin{cases} k_{s,d}, x_{r,q} \in X_{(i:i+g-1,\, j:j+h-1)} \\ 0, x_{r,q} \notin X_{(i:i+g-1,\, j:j+h-1)} \end{cases} \tag{۱۲-۲۸}$$

در رابطه ۱۲-۲۸، $k_{s,d}$ درایه متناظر از کرنل است که در آن گام حرکتی در درایه $x_{r,q}$ ورودی ضرب می شود.

۱۲.۴ لایه تجمیع دو بعدی[1]

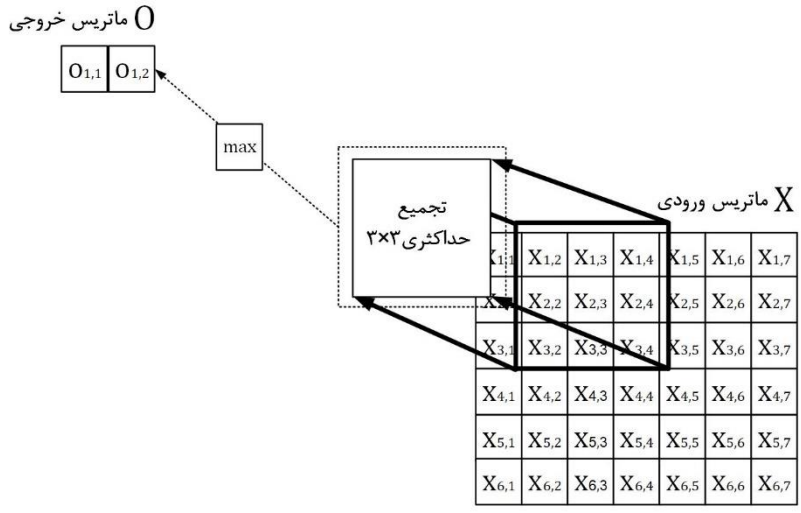
در ساختارهای پیچشی دو بعدی نیز همانند ساختار پیچشی یک بعدی با اعمال لایه تجمیع می توانیم ابعاد ماتریس، تانسور، ورودی به لایه را کاهش دهیم. لایه تجمیع به دو صورت، تجمیع حداکثری و میانگین تعریف می شود که در ادامه به بررسی روابط پیشرو و آموزش آن می پردازیم. در حالت کلی اعمال لایه تجمیع مشابه اعمال کرنل ضرب پیچشی دو بعدی به ورودی است، همان طور که در روابط ۱۲-۴ و ۱۲-۵ ابعاد خروجی عملیات ضرب پیچشی دو بعدی متاثر از ابعاد کرنل و گام های حرکتی و... است، در لایه تجمیع نیز با تنظیم پارمترهای مشابه می توان ابعاد ماتریس ورودی را کاهش دهیم. در در حالت کلی در لایه تجمیع اندازه گام های حرکتی با اندازه لایه تجمیع برابر بوده و اگر آن را مانند ضرب پیچشی در نظر بگیریم در هر گام حرکتی محل کرنل با

[1] 2-D pooling layer

محل گام قبل آن هم پوشانی[1] ندارد اما در برخی ساختارها چنان که در ادامه بررسی خواهیم کرد با در نظر گرفتن اندازه گام های حرکتی کوچک تر برای آن به طوری که با مکان گام قبل هم پوشانی داشته باشد باعث کاهش احتمال بیش برازش در روند آموزش ساختار می شود. بدیهی است که یک لایه تجمیع با گام های حرکتی کوچک تر ابعاد ورودی را به میزان کمتری کاهش می دهد.

۱۲.۴.۱ تجمیع حداکثری[2] دو بعدی

در این روش ماتریس لایه تجمیع مطابق شکل ۶-۱۲ در هر گام حرکتی مشابه کرنل پیچشی رو به روی تعدادی از درایه های ماتریس ورودی قرار گرفته و مقدار درایه ای که بیشترین مقدار را بین مقادیر رو به روی ماتریس تجمیع دارد به عنوان درایه خروجی آن گام حرکتی تعیین شده و سپس ماتریس تجیمع با گام های حرکتی تعیین شده در دو جهت افقی و عمودی همانند ضرب پیچشی دو بعدی حرکت کرده و سایر ابعاد خروجی را می سازد. در صورتی که ورودی لایه تجمیع ماتریس (یک تانسور با عمق) باشد عملیات تجمیع دو بعدی جداگانه بر روی هر یک از این ماتریس ها اعمال شده و در نهایت یک تانسور با عمق برابر با تانسور ورودی ولی با طول و عرض کوچک تر تشکیل می شود. در شکل ۶-۱۲ گام حرکتی تجمیع حداکثری دو بعدی با اندازه ۳×۳ در دو جهت یک فرض شده است.



شکل ۶-۱۲: تجمیع حداکثری دو بعدی

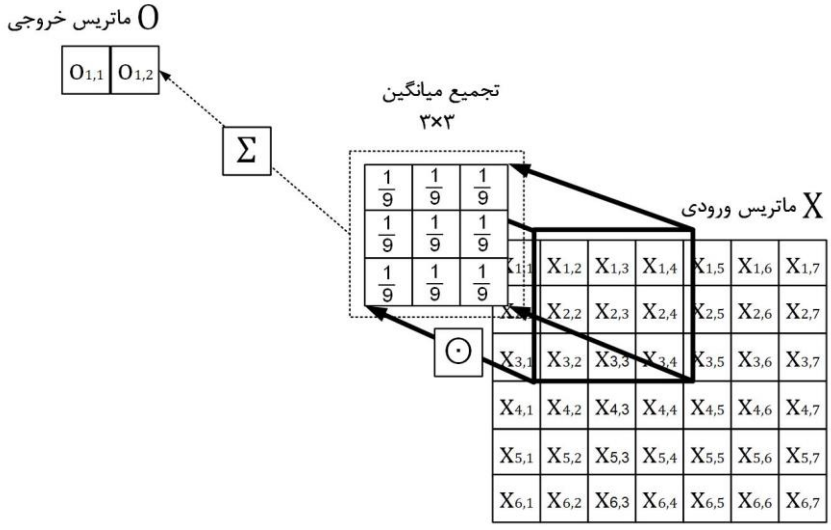
[1] Overlapping
[2] Max pooling

رابطه پیشرو برای تجمیع حداکثری مطابق رابطه ۱۲-۲۹ تعریف می شود.

$$o_{i,j} = \arg\max X_{(i:i+g-1,\, j:j+h-1)} \tag{۱۲-۲۹}$$

۱۲.۴.۲ تجمیع میانگین[1]

روش تجمیع میانگین معادل اعمال یک کرنل پیچشی دو بعدی با ابعاد $g \times h$ به ورودی است به طوری که اندازه هر یک از درایه های این کرنل مقداری برابر $\frac{1}{g \times h}$ دارند. حجم محاسبات در روش تجمیع میانگین بیشتر از روش تجمیع حداکثری است اما برخلاف روش تجمیع حداکثری که در آن تنها تعداد محدودی از درایه های ورودی در محاسبه خروجی دخیل بودند همه درایه ها وارد محاسبات می شوند. شکل ۱۲-۷ روند روش تجمیع میانگین را نشان می دهد.



شکل ۱۲-۷: تجمیع میانگین دو بعدی

رابطه پیشرو برای تجمیع حداکثری مطابق رابطه ۱۲-۳۰ تعریف می شود.

$$o_{i,j} = \frac{1}{g \times h} sum(X_{(i:i+g-1,\, j:j+h-1)}) \tag{۱۲-۳۰}$$

۱۲.۴.۳ مشتقات لایه تجمیع

با توجه به ماهیت لایه تجمیع، این لایه فاقد پارامترهای آموزشی بوده و صرفا برای کاهش بعد به کار برده می شود. اما اگر این لایه بین لایه های یک ساختار متشکل از توالی لایه های پیچشی و

[1] Average pooling

تجیمع باشد برای آموزش لایه های پیش از آن لازم است که مشتق خروجی این لایه نسبت به ورودی آن را که تعدادی از جملات مشتقات زنجیره ای پارامترهای آموزشی لایه قبل را تشکیل می دهند محاسبه کنیم. مشتقات زنجیره ای خروجی لایه تجیمع نسبت به ورودی آن به ترتیب برای روش تجمیع حداکثری و میانگین مطابق روابط ۳۱-۱۲ و ۳۳-۱۲ است.

$$\underset{e}{\frac{\partial E}{\partial \mathbf{e}}}\ \underset{-1}{\frac{\partial \mathbf{e}}{\partial o_{i,j}}}\ \frac{\partial o_{i,j}}{\partial x_{r,q}}\cdots \tag{۱۲-۳۱}$$

$$\frac{\partial o_{i,j}}{\partial x_{r,q}} = \begin{cases} 1, & if \quad o_{i,j} = \arg\max X_{(i:i+g-1,\, j:j+h-1)} \\ 0, & if \quad o_{i,j} \neq \arg\max X_{(i:i+g-1,\, j:j+h-1)} \end{cases} \tag{۱۲-۳۲}$$

$$\underset{e}{\frac{\partial E}{\partial \mathbf{e}}}\ \underset{-1}{\frac{\partial \mathbf{e}}{\partial O}}\ \underset{\frac{1}{g\times h}}{\frac{\partial O}{\partial X}}\cdots \tag{۱۲-۳۳}$$

در روابط ۳۱-۱۲ تا ۳۳-۱۲ به منظور تسهیل محاسبات خطای $\boldsymbol{e}$ به صورت یک ماتریس با بعدی برابر با ماتریس خروجی O در نظر گرفته شده است. همان طور که پیش تر اشاره کردیم در روش تجیمع میانگین همه درایه های ورودی برای محاسبه خروجی دخیل هستند، این عامل همان طور که از روابط ۳۱-۱۲ تا ۳۳-۱۲ نیز استنباط می شود باعث می شود که مقادیر مشتقات زنجیره ای از همه درایه های ورودی لایه تجمیع میانگین برای آموزش پارامترهای لایه های پیشین عبور کنند در حالی که این مقادیر در روش تجمیع حداکثری از تعداد محدودی از درایه های ورودی عبور می کنند.

۱۲.۵ نرمال سازی دسته ای[1]

مجموعه تصاویر آموزشی که به عنوان ورودی ساختارهای پیچشی استفاده می شوند نیز همانند سایر مجموعه دادگان آموزشی قبل از آغاز روند آموزش ساختار با استفاده از یکی از روش های معرفی شده در فصل نرمال سازی می شوند. پارامترهای عملیات نرمال سازی در ساختار های پیچشی که یکی از مراحل پیش پردازش محسوب می شود، برای هر درایه از تصویر به طور جداگانه به ازای همه نمونه های مجموعه محاسبه می شود، به عنوان مثال اگر از روش نرمال سازی توزیع

[1] Batch normalization

استاندارد[1]، میانگین صفر و واریانس یک، استفاده کنیم برای هر یک از پیکسل های موجود در طول و عرض یک تصویر باید یک مقدار میانگین μ و واریانس σ متناظر محاسبه کنیم، هر یک از پیکسل ها به عنوان یک بعد در نظر گرفته می شود، برای محاسبه این مقادیر میانگین و واریانس مقدار عددی پیکسل مورد نظر در همه تصاویر، در محاسبات لحاظ می شوند. با این توضیحات بدیهی است که ابعاد همه تصاویر یک مجموعه دادگان آموزشی باید یکسان باشند تا هر پیکسل از یک نمونه تصویری دارای یک پیکسل متناظر با موقعیت مشابه در سایر نمونه ها باشد، در همه تصاویر بعد های مشابهی به عنوان ابعاد ورودی به ساختار تعریف شود). با توجه به ساختار شبکه های پیچشی استفاده از روش نرمال سازی توزیع استاندارد برای این تصاویر مناسب تر است؛ زیرا این روش علاوه بر این که مقادیر را در بازه منفی یک تا یک قرار داده و باعث می شود تا توابع فغال ساز ReLU به صورت تنک فعال شوند، با تجیمع اکثر مقادیر نزدیک به نقطه صفر از ایجاد مقادیر خروجی بزرگ تر در بازه مثبت توسط تابع ReLU که متعاقبا باعث رشد بیش از حد گرادیان ها[2] و وزن ها[3] در محاسبه روابط آموزش می شود جلوگیری می کند.

مزایای فوق الذکر برای نرمال سازی بر اساس توزیع استاندارد برای دادگان ورودی فقط در لایه اول پیچشی مفید واقع می شود در لایه های بعدی با توجه به استفاده از توابع فعال ساز ReLU در این ساختارهای پیچشی دو بعدی، انتظار داریم که خروجی لایه پیچشی شامل مقادیری مثبت و صفر باشد. در صورت استفاده از توابع جایگزینی مانند Leaky ReLU نیز خروجی لایه اگرچه متشکل از مقادیر مثبت و منفی تشکیل شده، اما همواره مقیاس اعداد مثبت بزرگتر اعداد منفی است. با این تفاسیر، مقدار میانگین محاسبه شده برای هر یک از درایه های نگاشت ویژگی به ازای همه نمونه های مجموعه دادگان آموزشی همواره مقداری مثبت و مقدار واریانس آن نیز مقداری غیر از ۱ خواهد داشت[4]. از آن جائی که هر نگاشت ویژگی (خروجی لایه) به عنوان ورودی لایه بعد در نظر گرفته می شود این انحراف مقادیر میانگین و واریانس از مقادیر توزیع استاندارد همانند عدم استفاده از روش نرمال سازی فوق الذکر در آغاز روند آموزش بوده و می تواند در توالی چند لایه باعث ایجاد مقادیر بزرگ در خروجی تابع ReLU شده و بروز پدیده رشد بیش از حد گرادیان های آموزشی متاثر از این مقادیر خروجی لایه ها و متعاقبا رشد بیش از حد وزن های ساختار شود. علاوه بر پدیده رشد بیش از حد وزن ها خروجی یک لایه تابع فعال ساز ReLU در صورتی که به عنوان ورودی وارد لایه پیچشی بعدی شود شامل مقادیر مثبت زیادی است، این عامل با ادامه روند پیشرو سبب فعال شدن تعداد بیشتری از توابع فعال ساز ReLU در لایه های نزدیک به خروجی شده و باعث اختلال در روند تنک زائی می شود. با این توضیحات لازم است همواره مقادیر خروجی لایه های پیچشی پس از اعمال تابع فعال ساز ReLU همانند مجموعه دادگان ورودی نرمال سازی

[1] Standard Gaussian distribution
[2] Exploding gradients
[3] Exploding weights
[4] Variance shift

شـوند. روش نرمال سـازی دسـته ای بدین منظور اسـتفاده می شـود. روند اجرای این روش همانند نرمال سـازی با توزیع اسـتاندارد دادگان ورودی اسـت. در این روش برای هر یک از درایه های هر نگاشـت ویژگی در خروجی یک لایه پیچشـی دو بعدی، هر یک از درایه های تانسـور خروجی، یک مقدار میانگین و واریانس محاسبه می شود. برای محاسبه این مقادیر باید روند پیشرو به ازای همه نمونه های مجموعه دادگان اعمال شـود تا مقادیر نگاشـت ویژگی متناظر با نمونه های مختلف محاسـبه و در نهایت به ازای آن ها برای هر درایه مقدار میانگین و واریانس تعریف شـود. رابطه محاسبه مقدار میانگین واریانس هر یکی از درایه های تانسور خروجی یکی از لایه های پیچشی به ترتیب مطابق روابط ۳۴-۱۲ و ۳۵-۱۲ اسـت. این مقادیر برای نرمال سـازی دسـته ای یک تانسـور، باید برای همه درایه های تانسور محاسبه شوند.

$$\mu_{o_{i,j}} = \frac{1}{N}\sum_{k=1}^{N} o_{i,j} \qquad (۱۲-۳۴)$$

$$\sigma_{o_{i,j}} = \sqrt{\frac{1}{N-1}\sum_{k=1}^{N-1}(o_{i,j}^2 - \mu_{o_{i,j}}^2)} \qquad (۱۲-۳۵)$$

پس از محاسـبه مقادیر روابط ۳۴-۱۲ و ۳۵-۱۲ مقدار عددی هر یک از درایه های تانسـور خروجی مطابق رابطه نرمال سازی شده و مقادیر نرمال به عنوان خروجی نهائی در نظر گرفته شده و وارد لایه بعد می شـوند. با این تفاسـیر می توانیم روش نرمال سـازی دسـته ای را یک لایه مجزا فرض کنیم که روند پیشـرو آن مطابق رابطه ۳۶-۱۲ و گرادیان خروجی آن نسبت به ورودی مطابق رابطه ۳۷-۱۲ است که برای آموزش لایه های پیش از آن به عنوان تعدادی از جملات رابطه مشتق زنجیره ای لایه قبل مورد اسـتفاده قرار می گیرد. همان طور که از روابط پیشـرو لایه نرمال سـازی دسـته ای نیز اسـتنباط می شـود مقادیر میانگین و واریانس هر یک از درایه ها به صـورت تحلیلی محاسبه شده و از این رو پارامتر آموزشی محسوب نمی شوند، اما در مواردی برای ساده سازی و رفع نیاز به مقادیر پیشرو همه نمونه ها در هر گام آموزشی و همچنین ایجاد امکان استفاده از روش های بهینه سـازی تصـادفی[1]، با وجود تناقض تئوریک مقادیر میانگین و واریانس را به عنوان پارامتر آموزشـی لایه در نظر گرفته و برای آن ها گرادیان های آموزشـی مطابق روابط ۳۸-۱۲ و ۳۹-۱۲ تعریف می کنیم، آموزش مقادیر میانگین و واریانس به عنوان جایگزین روابط ۳۴-۱۲ و ۳۵-۱۲ محسـوب شـده و معضـل مربوط به لزوم محاسـبه مقادیر پیشـرو همه نمونه ها را برطرف می کند؛ بدین صورت که در اولین گام مقادیر پیشرو برای همه نمونه های دادگان آموزشی محاسبه شده و بر اسـاس آن ها مقادیر میانگین و واریانس روابط ۳۴-۱۲ و ۳۵-۱۲ محاسـبه می شـوند، پس از آن مقادیر این پارامترهای با اسـتفاده از روابط آموزشـی، مطابق روابط ۴۱-۱۲ و ۴۲-۱۲ به روزرسـانی

[1] Stochastic optimization algorithms

شده و دیگر لزومی به استفاده از روابط ۳۴-۱۲ و ۳۵-۱۲ در ادامه روند آموزش نیست در حالی که در صورت عدم آموزش این مقادیر در هر گام آموزشی باید به ازای مقادیر پیشرو به ازای همه نمونه های مجموعه دادگان محاسبه شوند. در روابط ذیل به منظور ساده سازی فرض کرده ایم خطای $\boldsymbol{e}$ یک تانسور هم اندازه با تانسور $\hat{\boldsymbol{O}}$ بوده و همچنین تابع هزینه مجموع مربعات خطا است. در این روابط μ_O و σ_O به ترتیب تانسورهایی هم اندازه با تانسور $\boldsymbol{O}$، متشکل از مقادیر میانگین و واریانس متناظر با درایه های ماتریس $\boldsymbol{O}$ است. همچنین مقدار ε در رابطه ۳۶-۱۲ یک اسکالر است که در صورت صفر بودن واریانس از ایجاد مقدار صفر در مخرج جلوگیری می کند.

$$o'_{i,j} = \frac{o_{i,j} - \mu_{o_{i,j}}}{\sqrt{\sigma^2_{o_{i,j}} + \varepsilon}} \tag{۱۲-۳۶}$$

$$\frac{\partial E}{\partial \mathbf{e}} \frac{\partial \mathbf{e}}{\partial \mathbf{O}'} \frac{\partial \mathbf{O}'}{\partial \mathbf{O}} \frac{\partial \mathbf{O}}{\dots} \dots \tag{۱۲-۳۷}$$

$$\frac{\partial E}{\partial \mu_O} = \frac{\partial E}{\partial \mathbf{e}} \frac{\partial \mathbf{e}}{\partial \mathbf{O}'} \frac{\partial \mathbf{O}'}{\partial \mu_O} \tag{۱۲-۳۸}$$

$$\frac{\partial E}{\partial \sigma_O} = \frac{\partial E}{\partial \mathbf{e}} \frac{\partial \mathbf{e}}{\partial \mathbf{O}'} \frac{\partial \mathbf{O}'}{\partial \sigma_O} \tag{۱۲-۳۹}$$

$$\frac{\partial \mathbf{O}'}{\partial \sigma_O} = \frac{-1}{2}(2\sigma_O) \times (\sigma_O^2 + \varepsilon) \tag{۱۲-۴۰}$$

$$\mu_{(k+1)} = \mu_{(k)} - \eta \frac{\partial E}{\partial \mu}(k) \tag{۱۲-۴۱}$$

$$\sigma_{(k+1)} = \sigma_{(k)} - \eta \frac{\partial E}{\partial \sigma}(k) \tag{۱۲-۴۲}$$

با توجه به اینکه انحراف مقادیر خروجی از توزیع استاندارد پیش از اعمال تابع فعال ساز ReLU، با اعمال ضرب پیچشی مطابق رابطه ۳-۱۲ نیز ممکن است رخ دهد، می توان از روش نرمال سازی دسته ای با توجه به ساختار تعریف شده پیش از اعمال تابع فعال ساز نیز استفاده کرد، اما استفاده از آن چندان مرسوم نیست. همچنین با توجه به دو رابطه ۳۴-۱۲ و ۳۵-۱۲ برای اجرای روش نرمال سازی دسته ای باید روند پیشرو همه نمونه های مجموعه دادگان اجرا شود تا نگاشت ویژگی

متناظر با همه آن ها برای محاسبات رابطه ۳۶-۱۲ استفاده شود، بنابراین در صورت استفاده از این روش باید از روش های بهینه سازی دسته ای[1] که در فصل ۹ بررسی کردیم برای آموزش ساختار استفاده کنیم، البته می توان مقادیر میانگین و واریانس رابطه و را به ازای چند نمونه از مجموعه دادگان نیز محاسبه کرده و از روش بهینه سازی دسته ای کوچک[2] استفاده کنیم، در این صورت روابط و به صورت بوده و تانسور نگاشت ویژگی در هر دسته کوچک به طور مستقل از سایر دسته ها نرمال سازی می شود این روش باعث می شود بتوانیم روند آموزش ساختار را به صورت موازی پیاده سازی کنیم. امکان استفاده از روش های بهینه سازی تصادفی در صورت استفاده از نرمال سازی دسته ای وجود ندارد، مگر اینکه به جای محاسبه روابط ۳۴-۱۲ و ۳۵-۱۲ مطابق روابط ۴۱-۱۲ و ۴۲-۱۲ این مقادیر را آموزش دهیم که در این صورت نیز علاوه بر تناقض تئوریک روش پیاده سازی شده، با توجه به این که مقادیر حاصل متفاوت از مقادیر میانگین و واریانس تحلیلی دادگان است روند نرمال سازی و آموزش ساختار و متعاقبا عملکرد ساختار مطلوب نباشد. روش نرمال سازی دسته ای مختص ساختارهای پیچشی نبوده و در سایر ساختارها نیز می تواند مورد استفاده قرار گیرد، به عنوان مثال می توانیم خروجی بردار خروجی یک لایه از شبکه پرسپترون چند لایه با استفاده از این روش پیش از ورود به لایه بعدی نرمال سازی کنیم.

۱۲.۶ نرمال سازی لایه ای[3]

استفاده از روش نرمال سازی دسته ای با وجود مزایای ذکر شده برای آن و بهبود روند کلی آموزش ساختار دارای معایبی نیز می باشد؛ در صورت استفاده از این روش حتما باید از روش های بهینه سازی دسته ای یا دسته ای کوچک استفاده کنیم و امکان پیاده سازی روش های تصادفی وجود ندارد، مگر با آموزش مقادیر میانگین و واریانس که خود باعث مشکلاتی می شود. با توجه به روند همگرائی کند روش های بهینه سازی دسته ای، معمولا روش های دسته ای کوچک در صورت استفاده از روش نرمال سازی دسته ای اعمال می شوند اما پیاده سازی روش نرمال سازی دسته ای با استفاده از روش های بهینه سازی دسته ای کوچک در ساختارهایی که نمونه های مجموعه دادگان آموزشی نسبت به هم وابستگی دارند، مانند مجموعه دادگان سری زمانی که در فصل ۱۱ بررسی کردیم، با توجه به لزوم در دسترس بودن سایر نمونه های مجموعه بسیار دشوار بوده و معمولا امکان پیاده سازی آن به صورت موازی وجود ندارد. برای رفع مشکلات مربوط به روش نرمال سازی دسته ای، روش نرمال سازی لایه ای معرفی شده است. بر خلاف روش نرمال سازی دسته ای امکان به کارگیری این روش با استفاده از روش های بهینه سازی تصادفی، در کنار روش های بهینه سازی دسته ای و دسته ای کوچک وجود دارد.

در روش نرمال سازی لایه ای مقادیر میانگین و واریانس به ازای همه ابعاد خروجی یک لایه

[1] Batch optimization algorithms
[2] Mini-batch optimization algorithms
[3] Layer normalization

محاسبه می شود، به عنوان مثال مقدار میانگین و واریانس طبق روابط ۴۳-۱۲ و ۴۴-۱۲ با استفاده از همه از درایه های تانسور نگاشت ویژگی با ابعاد $l_1 \times l_2 \times C$ در خروجی یک لایه پیچشی به ازای یک نمونه ورودی محاسبه می شود.

$$\mu_O = \frac{1}{l_1 \times l_2 \times C} \sum_{k=1}^{C} \sum_{i=1}^{l_1} \sum_{j=1}^{l_2} o_{i,j,k} \tag{۱۲-۴۳}$$

$$\sigma_O = \sqrt{\frac{1}{l_1 \times l_2 \times C - 1} \sum_{k=1}^{C} \sum_{i=1}^{l_1} \sum_{j=1}^{l_2} (o_{i,j,k}^2 - \mu_O^2)} \tag{۱۲-۴۴}$$

در روابط فوق مقادیر μ_O و σ_O، به ترتیب مقدار میانگین یک لایه به صورت اسکالر هستند. پس از محاسبه مقادیر میانگین و واریانس از روابط ۴۳-۱۲ و ۴۴-۱۲ هر یک درایه های تانسور خروجی با استفاده از رابطه ۴۵-۱۲ نرمال سازی شده و در نهایت تانسور نرمال به عنوان ورودی لایه بعدی در نظر گرفته می شود.

$$o'_{i,j} = \frac{o_{i,j} - \mu_O}{\sqrt{\sigma_O^2 + \varepsilon}} \tag{۱۲-۴۵}$$

روش نرمال سازی لایه ای را همچون روش نرمال سازی دسته ای می توانیم یک لایه مجزا در نظر بگیریم، در این صورت مشتق خروجی لایه نسبت به ورودی آن که برای محاسبه روابط مشتق زنجیره ای برای پارامترهای لایه های پیش از آن مورد استفاده قرار می گیرد مطابق رابطه ۴۶-۱۲ است. همچنین با وجود رفع نیاز به محاسبه مقادیر پیشرو به ازای همه نمونه ها، در این روش نیز می توانیم مقادیر میانگین و واریانس را به عنوان پارامتر آموزشی در نظر بگیریم، البته با توجه به این که این مقادیر به صورت تحلیلی از روابط ۴۳-۱۲ و ۴۴-۱۲ محاسبه می شوند، فرض پارامتر آموزشی همچون فرض پارامتر آموزشی برای این مقادیر در روش نرمال سازی دسته ای، از نظر تئوریک متناقض است.

$$\frac{\partial E}{\partial \mathbf{e}} \frac{\partial \mathbf{e}}{\partial \mathbf{O}'} \frac{\partial \mathbf{O}'}{\partial \mathbf{O}} \frac{\partial \mathbf{O}}{\dots} \dots \qquad e \quad -1 \quad \frac{1}{\sqrt{\sigma_O^2 + \varepsilon}} \tag{۱۲-۴۶}$$

روش نرمال سازی لایه ای را می توانیم برای همه ساختارهای دیگر نیز استفاده کنیم، همچنین مانند روش نرمال سازی دسته ای قبل از اعمال تابع فعال ساز نیز می توانیم آن را در ساختار

تعریف کنیم، اما این حالت چندان مرسوم نیست. در روش نرمال سازی لایه ای میانگین و واریانس به ازای ابعاد مختلف یک بردار، خروجی، محاسبه می شود، در حالی که این موضوع در تناقض با تعریف کلی روش نرمال سازی است که در آن باید مقادیر هر بعد جداگانه به ازای نمونه های مختلف مجموعه دادگان نرمال سازی شود. اما با توجه به این که در لایه های یک ساختار نگاشت، تابع فعال ساز، یکسانی به همه ابعاد ورودی آن لایه اعمال می شود و همچنین توزیع ابعاد مختلف پیش از ورودی به ساختار در مرحله پیش پردازش نرمال شده و توزیع و بازه عددی یکسانی برای همه ابعاد در نظر گرفته شده است می توانیم این تناقض را یک فرض برای سادگی محاسبات در نظر گرفته و با وجود نقض تئوریک آن همچنان از این روش استفاده کنیم.

۱۲.۷ نرمال سازی پاسخ محلی[1]

یک روش دیگر برای نرمال سازی خروجی یک تانسور خروجی لایه پیچشی روش نرمال سازی پاسخ محلی است که نسبت به دو روش نرمال سازی مطرح شده حجم محاسباتی کمتری دارد. این روش بعد از اعمال تابع فعال ساز غیرخطی به تانسور اعمال می شود. در این روش مقادیر هر یک از درایه های تانسور نسبت به مقادیر درایه های اطراف خود نرمال سازی می شود. این روش از ویژگی مهار جانبی[2] نورون های بیولوژیکی الهام گرفته شده است که در آن هر نورون سعی در کاهش فعالیت نورون های مجاور دارد. روش نرمال سازی پاسخ محلی به دو روش داخل یک کانال و بین کانال ها اجرا می شود که به بررسی هر کدام از آن ها می پردازیم.

۱۲.۷.۱ روش نرمال سازی داخل هر کانال[3]

در این روش ابتدا تانسور ورودی را به صورت ماتریس های مجزا در نظر می گیریم. برای هر یک از درایه های هر یک از ماتریس ها یک مقدار اسکالر به عنوان اندازه همسایگی در نظر گرفته و مقدار هر درایه را به با استفاده از مقادیر درایه های اطراف آن به فاصله مقدار اندازه همسایگی آن درایه مطابق رابطه ۴۷-۱۲ نرمال می کنیم. برای مثال همسایگی به اندازه ۱ برای درایه a در یک ماتریس شامل یک محدوده ۳×۳ می شود که درایه a در مرکز آن قرار دارد. با توجه به رابطه ۴۷-۱۲ برای درایه هایی که برخی از درایه های همسایه برای آن تعریف نمی شود فقط از درایه های موجود برای نرمال سازی استفاده می کنیم؛ برای مثال درایه گوشه سمت چپ یک ماتریس، $a_{1,1}$، را در نظر بگیرید در صورت تعریف همسایگی یک باید برای نرمال سازی آن فقط درایه های $a_{1,1}, a_{2,1}, a_{1,2}, a_{2,2}$ استفاده کنیم زیرا سایر درایه های $a_{0,1}, a_{0,0}, a_{1,0}, a_{2,0}$ که در محدوده همسایگی یک درایه $a_{1,1}$ است تعریف نمی شوند. در رابطه ۴۷-۱۲، n اندازه همسایگی، α, β ثابت

[1] Local response normalization (LRN)
[2] Lateral inhibition
[3] Intra-channel LRN

های اسکالر و k اسکالری برای جلوگیری از مخرج صفر است. در صورتی که بخواهیم نرمال سازی به صورت توزیع استاندارد باشد باید مقادیر α, β یک و مقدار k صفر باشد.

$$o'_{i,j} = \frac{o_{i,j}}{(k + \alpha \sum_{p=\max(0,i-n)}^{\min(l_1,i+n)} \sum_{y=\max(0,j-n)}^{\min(l_2,j+n)} {o_{p,y}}^2)^{\beta}}$$
$$i = 0,1,...,l_1,\ j = 0,1,...,l_2 \qquad (۱۲-۴۷)$$

۱۲.۷.۲ روش نرمال سازی بین کانال ها[1]

در این روش نیز تانسور ورودی را به صورت ماتریس های مجزا از هم در نظر گرفته و یک مقدار اسکالر صحیح مثبت را به عنوان اندازه همسایگی تعریف می کنیم. سپس مقدار هر یک از درایه های این ماتریس ها را با استفاده از مقادیر همان درایه در ماتریس های مجاور پیش و پس از آن با استفاده از رابطه ۴۸-۱۲ نرمال می کنیم. محدوده محاسبات در نرمال سازی هر درایه در این روش برای اندازه همسایگی n به صورت یک بردار با اندازه $2n + 1$ خواهد بود. با توجه به رابطه ۴۸-۱۲ در این روش نیز همانند روش پیشین در صورتی که درایه ماتریسی در همسایگی آن ماتریس مورد نظر تعریف نشده باشد در محاسبات دخیل نمی شود. برای مثال $a_{n,n,1}$ را در نظر بگیرید که در اولین ماتریس، اولین عمق تانسور، قرار دارد در صورت تعریف همسایگی یک باید مقدار آن نسبت به درایه های $a_{n,n,2}$ و $a_{n,n,1}$ و نرمال سازی شود و برای آن درایه $a_{n,n,0}$ تعریف نمی شود. در رابطه ۴۸-۱۲، n اندازه همسایگی، α, β ثابت های اسکالر و k اسکالری برای جلوگیری از مخرج صفر است. در صورتی که بخواهیم نرمال سازی به صورت توزیع استاندارد باشد باید مقادیر α, β یک و مقدار k صفر باشد.

$$o'_{i,j,k} = \frac{o_{i,j,k}}{(k + \alpha \sum_{y=\max(0,k-n)}^{\min(l_3,k+n)} {o_{i,j,y}}^2)^{\beta}}$$
$$i = 0,1,...,l_1,\ j = 0,1,...,l_2, k = 0,1,...,l_3 \qquad (۱۲-۴۸)$$

۱۲.۸ لایه مسطح ساز[2]

یک ساختار پیچشی معمولا به صورت توالی لایه های پیچشی، تجمیع و نرمال سازی تعریف می شود که هدف آن تبدیل تصویر، تانسور، ورودی به یک قالب قابل استفاده برای ساختارهای

[1] Inter-channel LRN
[2] Flatten layer

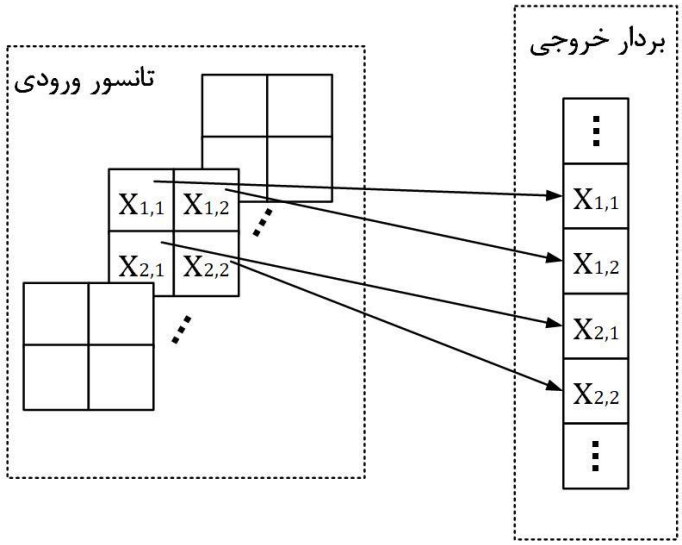
مرسوم مانند شبکه های عصبی پرسپترون چند لایه است، از این منظر می توانیم این ساختار را مشابه ساختار خودرمزگذارهای پشته ای در نظر بگیریم با این تفاوت که آموزش پارامترهای این ساختار با آموزش با سرپرست[1] است. با این اوصاف خروجی این ساختار باید به صورت یک بردار باشد تا به عنوان ورودی، وارد ساختارهایی مانند شبکه های عصبی پرسپترون چند لایه شود، اما همان طور که بررسی کردیم خروجی همه ی لایه های ذکر شده در یک ساختار پیچشی به صورت تانسور است. لایه مسطح ساز به منظور تبدیل تانسور خروجی به یک بردار تعریف می شود. این لایه فاقد پارامترهای آموزشی بوده و فقط درایه های مختلف هر یک از ماتریس های خروجی، هر یک از عمق های تانسور خروجی، ساختار پیچشی را به ترتیب در کنار هم قرار داده و یک بردار می سازد. با این توضیحات اگر تانسور خروجی ساختار پیچشی دارای ابعاد $m \times n \times C$ باشد، اندازه بردار خروجی حاصل از اعمال لایه مسطح ساز $m \times n \times C = mnC$ خواهد بود. برای این لایه می توانیم یک رابطه پیشرو مطابق رابطه ۴۹-۱۲ تعریف کنیم، همچنین شکل ۸-۱۲ روند تبدیل یکی از ماتریس های خروجی، یکی از عمق های تانسور، ساختار پیچشی به ابعاد یک بردار در لایه مسطح ساز را نشان می دهد، روند نمایش داده شده در شکل ۸-۱۲ از اولین ماتریس، عمق تانسور، خروجی آغاز شده و تا آخرین عمق ادامه پیدا می کند، ابعاد متناظر حاصل برای بردار از هر یک از این ماتریس ها، عمق های تانسور، به ترتیب در کنار ابعاد قبلی بردار قرار می گیرند.

$$x'_{((k-1)mn)+((i-1)n)+j} = x_{i,j,k}$$
$$i = 1,\ldots,l_1,\ j = 1,\ldots,l_2,\ k = 1,\ldots,C \qquad (۱۲-۴۹)$$

مشتق خروجی نسبت به ورودی لایه مسطح ساز که برای آموزش لایه های ساختار پیچشی قبل از آن مورد نیاز است را به صورت رابطه ۵۰-۱۲ تعریف می کنیم.

$$\underbrace{\frac{\partial x'_{((k-1)mn)+((i-1)n)+j}}{\partial x_{i,j,k}}}_{1} \frac{\partial x_{i,j,k}}{\cdots} \cdots \qquad (۱۲-۵۰)$$

[1] Supervised



شکل ۸-۱۲: لایه مسطح ساز

۱۲.۹ لایه تجمیع میانگین سراسری دو بعدی[1]

اشاره کردیم که هدف استفاده از لایه مسطح ساز، تبدیل تانسور به یک بردار قابل استفاده برای ساختارهایی مانند شبکه های عصبی پرسپترون چند لایه است، اما مشکل استفاده از این لایه این است که بردار حاصل از یک تانسور همان طور که در بخش ۷-۱۲ بررسی کردیم دارای ابعاد بزرگی بوده و استفاده از آن برای ساختاری مانند شبکه عصبی پرسپترون چند لایه باعث بروز مشکلاتی در روند آموزش آن می شود. برای رفع معضل مربوط به بعد بالای بردار لایه مسطح ساز از لایه تجیمع میانگین سراسری استفاده می کنیم، اندازه بردار حاصل از لایه تجمیع میانگین سراسری برای یک تانسور ورودی با ابعاد $m \times n \times C$ برابر C است که همان عمق تانسور، تعداد ماتریس هایی به ابعاد C، است. رابطه پیشرو لایه تجمیع میانگین سراسری به صورت رابطه ۵۱-۱۲ تعریف می شود. همان طور که از رابطه استنباط می شود در این روش به ازای هر یک از عمق های تانسور ورودی تنها یک بعد در بردار خروجی ایجاد می شود که همان مقدار میانگین محاسبه شده به ازای همه ابعاد هر عمق است.

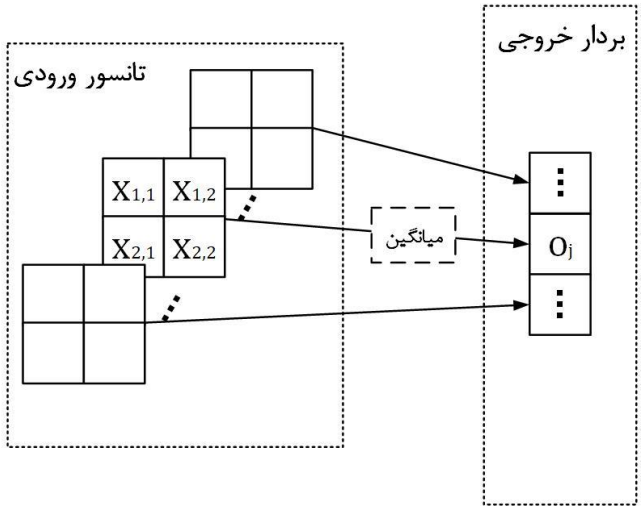
$$o_k = \frac{1}{m \times n} \sum_{i=1}^{m} \sum_{j=1}^{n} x_{i,j,k}, k = 1, ..., C \qquad (۱۲-۵۱)$$

مشتق خروجی یک لایه تجیمع میانگین سراسری نسبت به ورودی آن را به صورت رابطه ۵۲-۱۲ تعریف می کنیم.

[1] 2-D Global average pooling (GAP)

$$\frac{\partial \mathbf{o}}{\partial X}\frac{\partial X}{\ldots}\ldots$$
$$\frac{1}{m\times n} \qquad (۱۲-۵۲)$$

با وجود حجم محاسبات بیشتر لایه تجیمع میانگین سراسری در مقایسه با لایه مسطح ساز این لایه اطلاعات کمتری را در بردار خروجی خود لحاظ می کند. استفاده از هر دو لایه مسطح ساز و تجمیع میانگین سراسری در ساختارهای پیچشی مرسوم بوده و با توجه به ساختار هر یک از آن ها می تواند عملکرد مناسبی داشته باشد. شکل ۹-۱۲ یک لایه تجمیع سراسری دو بعدی را نشان می دهد.

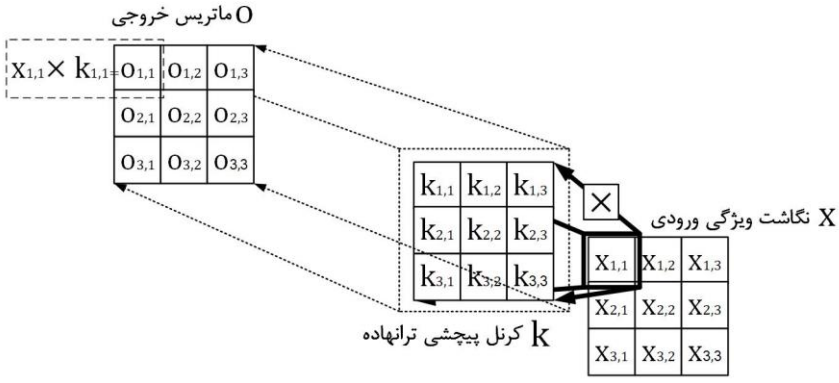


شکل ۹-۱۲: لایه تجمیع سراسری دو بعدی

در لایه تجیمع سراسری می توان به جای عملیات میانگین، مقدار حداکثری را از هر یک از ماتریس های خروجی، عمق های تانسور خروجی، انتخاب کنیم که در این صورت به آن لایه تجمیع سراسری حداکثری دو بعدی اطلاق می شود. از روش تجمیع سراسری حداکثری به دلیل گرادیات های آموزشی محدودتری که برای لایه های پیش از خود دارد، نسبت به تجمیع سراسری میانگین کمتر استفاده می شود.

۱۲.۱۰ ضرب پیچشی دو بعدی ترانهاده[1]

اشاره کردیم که خروجی یک ساختار پیچشی تبدیل به بردار شده و به عنوان ورودی به یک ساختار دیگر مانند شبکه عصبی پرسپترون چند لایه داده می شود. این حالت کلی برای مواردی است که مقادیر مطلوب هر یک از نمونه های مجموعه تصاویر آموزشی به صورت بردار تعریف شده باشند؛ کاربرد هایی مانند استفاده از ساختارهای پیچشی برای طبقه بندی از این دسته هستند. اما در مواردی خروجی یک ساختار پیچشی نیز به صورت تصویر، تانسور، تعریف می شود، در چنین مواردی همانند ساختار رمزگذار و کدگشا در خودرمزگذارها، لازم است پس از اعمال لایه های پیچشی، لایه هایی تعریف شوند که نگاشت ویژگی خروجی این لایه های پیچشی را افزایش بعد[2] داده و به ابعاد تصویر خروجی تبدیل کنند. عملیاتی که برای این لایه ها تعریف می شود ضرب پیچشی دو بعدی ترانهاده است. در این عملیات مشابه عمیات ضرب پیچشی دو بعدی یک کرنل به صورت ماتریس تعریف می شود. یک ماتریس نگاشت ویژگی به عنوان ورودی این عملیات در نظر گرفته شده و برای حرکت کرنل بر روی ماتریس خروجی دو گام حرکتی در جهت افقی و عمودی تعریف می شود، سپس مطابق شکل ۹-۱۲ همه درایه های کرنل در اولین درایه نگاشت ویژگی، $x_{1,1}$، ضرب شده، ضرب اسکالر در ماتریس، و بخشی از درایه های ماتریس خروجی را تشکیل می دهد.

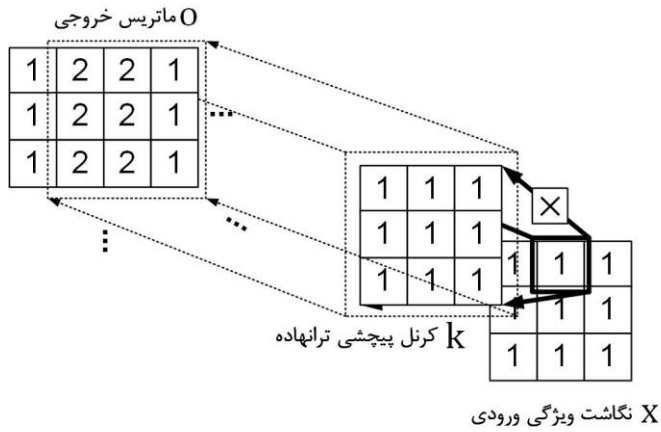


شکل ۱۰-۱۲: ضرب پیچشی ترانهاده دو بعدی

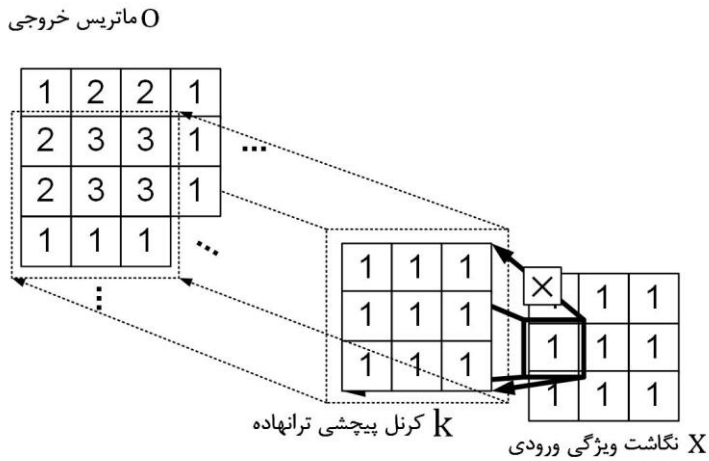
در ادامه روند عملیات مقادیر درایه های کرنل در هر گام حرکتی در یکی از درایه های نگاشت ویژگی ضرب شده و با حرکت افقی و عمودی در هر گام حرکتی ابعاد دیگر تصویر خروجی را می سازد. شکل ۱۱-۱۲ و ۱۲-۱۲ شکل به ترتیب ضرب کرنل در درایه x_{12} و x_{21} نگاشت ویژگی و حرکت افقی و عمودی کرنل در ماتریس خروجی را نشان می دهد. برای درک بهتر در شکل های

[1] 2-D transpose convolution
[2] Upsampling

۱۱-۱۲ و ۱۲-۱۲ برای مقادیر درایه های ورودی و کرنل، مقدار یک در نظر گرفته ایم.



شکل ۱۱-۱۲: ادامه روند ضرب پیچشی ترانهاده دوبعدی در جهت افقی



شکل ۱۲-۱۲: ادامه روند ضرب پیچشی ترانهاده در جهت عمودی

رابطه پیشرو بخشی از درایه های ماتریس خروجی که در هر گام حرکتی مطابق شکل های ۱۰-۱۲، ۱۱-۱۲ و ۱۲-۱۲ تعدادی از آن ها محاسبه می شوند مطابق رابطه ۵۳-۱۲ است.

$$O_{(i:i+g,\,j:j+h)} = x_{i,j} \times K_{g\times h} \qquad (۱۲-۵۳)$$

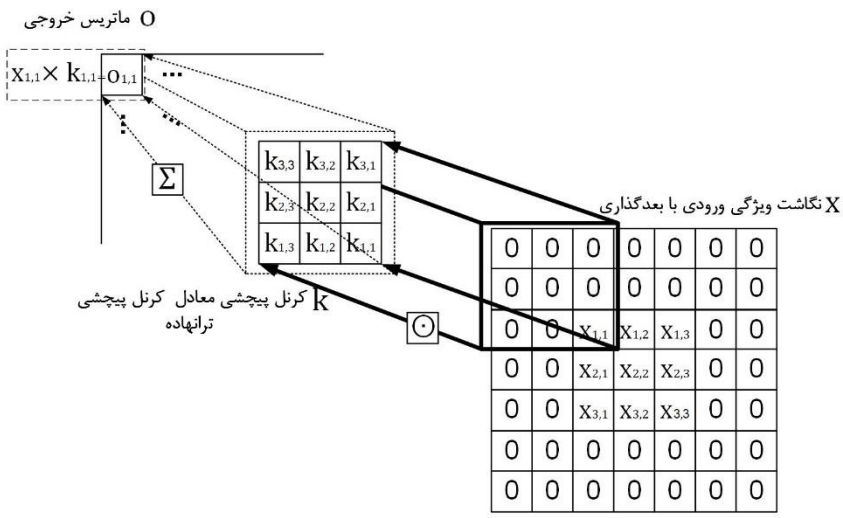
روند ضرب پیچشی دوبعدی ترانهاده را بار دیگر از دیدگاه درایه های ماتریس خروجی بررسی می کنیم؛ درایه $o_{1,1}$ ماتریس خروجی را در نظر بگیرید این درایه از ضرب درایه $x_{1,1}$ ماتریس

ورودی در درایه $k_{1,1}$ ماتریس کرنل محاسبه می شود، همچنین درایه $o_{1,2}$ از مجموع دو گام حرکتی حاصل از ضرب درایه $x_{1,1}$ ماتریس ورودی در درایه $k_{1,2}$ ماتریس کرنل در گام حرکتی اول و ضرب درایه $x_{1,2}$ ماتریس ورودی در درایه $k_{1,1}$ ماتریس کرنل در گام حرکتی دوم است. روابط ۱۲-۵۴ و ۱۲-۵۵ به ترتیب نشان دهنده این روند محاسبات برای دو درایه خروجی مذکور هستند. روابط مشابهی برای هر یک از درایه های ماتریس خروجی می توان تعریف کرد.

$$o_{1,1} = x_{1,1}k_{1,1} \tag{۱۲-۵۴}$$

$$o_{1,2} = x_{1,1}k_{1,2} + x_{1,2}k_{1,1} \tag{۱۲-۵۵}$$

با توجه به روابط ۱۲-۵۴ و ۱۲-۵۵ و تعمیم آن ها به سایر درایه های خروجی می توانیم چنین نتیجه گیری کنیم که می توان روند ضرب پیچشی دو بعدی ترانهاده را به صورت یک ضرب پیچشی مطابق شکل ۱۲-۱۳ نیز تعریف کنیم. در صورت تبدیل ضرب پیچشی ترانهاده به ضرب پیچشی ماتریس کرنل ترانهاده حول قطر فرعی مطابق شکل ۱۲-۱۳ قرینه شده و یک کرنل پیچشی حاصل می شود. همچنین اندازه بعدگذاری اعمال شده به ماتریس ورودی دو برابر مقدار حاصل از روابط ۱۲-۱۶ و ۱۲-۱۷ است. با تبدیل ضرب پیچشی ترانهاده به ضرب پیچشی روند پیاده سازی آن به ویژه در مرحله محاسبه مشتقات زنجیره ای ساده تر می شود؛ طبق شکل های ۱۲-۱۱ و ۱۲-۱۲ مقدار نهائی هر یک از درایه های ماتریس خروجی از مجموع مقادیر حاصل از گام های حرکتی مختلف محاسبه می شود، در صورتی که بخواهیم کرنل پیچشی ترانهاده را آموزش دهیم باید روابط مشتقات زنجیره ای آن را برای هر گام حرکتی محاسبه کنیم که بدین منظور باید مقدار نهائی هر یک از درایه های خروجی را به مقادیر مجموع حاصل از هر گام تجزیه کنیم که تحقق آن مستلزم ذخیره مقدار هر گام به طور جداگانه بوده و باعث اشغال حافظه رایانه می شود. در صورتی که با تبدیل ضرب پیچشی ترانهاده به ضرب پیچشی مشتقات زنجیره ای مطابق روابط تعریف شده برای ضرب پیچشی دو بعدی، روابط ۱۲-۲۰ تا ۱۲-۲۲، به راحتی محاسبه می شوند.



شکل ۱۳-۱۲: تبدیل ضرب پیچشی ترانهاده دو بعدی به ضرب پیچشی دو بعدی

ضرب پیچشی ترانهاده را می توانیم معکوس عملیات پیچشی در نظر بگیریم. بر همین اساس به ضرب پیچشی ترانهاده، ضرب پادپیچشی[1] نیز اطلاق می شود. همان طور که در ضرب پیچشی با تعریف گام های حرکتی بزرگ تر، ۱ < گام حرکتی، می توانستیم ابعاد خروجی را با توجه به روابط ۴-۱۲و ۵-۱۲ نسبت به خروجی گام حرکتی ۱ کاهش دهیم، بالعکس می توانیم با تعریف گام های حرکتی کوچک تر از یک برای ضرب پیچشی ترانهاده ابعاد خروجی حاصل از ضرب پیچشی ترانهاده را نسبت به گام حرکتی یک افزایش دهیم. فرض کنیم در یک عملیات ضرب پیچشی اندازه گام حرکتی را n تعریف کنیم، در این صورت اعمال این ضرب پیچشی به ماتریس ورودی سبب ایجاد یک ماتریس خروجی با ابعادی حاصل از روابط ۴-۱۲ و ۵-۱۲ ورودی می شود، برای معکوس کردن روند و تبدیل ماتریس خروجی ضرب پیچشی به یک ماتریس با ابعادی برابر با ابعاد ماتریس ورودی ضرب پیچشی همان طور که اشاره کردیم از ضرب پیچشی ترانهاده استفاده می کنیم، در این صورت اگر اندازه کرنل تعریف شده در هر دو عملیات مساوی باشد، اندازه گام حرکتی ضرب پیچشی ترانهاده برابر $\frac{1}{n}$ خواهد بود. با این تفاسیر همواره بین اندازه گام حرکتی یک ضرب پیچشی و ضرب پیچشی ترانهاده معکوس آن به طوری که اندازه کرنل آن ها برابر بوده و ابعاد خروجی یکی برابر ابعاد خروجی دیگری باشد رابطه ۵۶-۱۲ برقرار است. با توجه به تعریف مقادیر گام های حرکتی کسری در ضرب پیچشی ترانهاده به آن ضرب پیچشی با گام کسری[2] نیز گفته می شود.

$$n_{Conv.} \leftrightarrow \frac{1}{n_{TRS.Conv.}} \qquad (۱۲-۵۶)$$

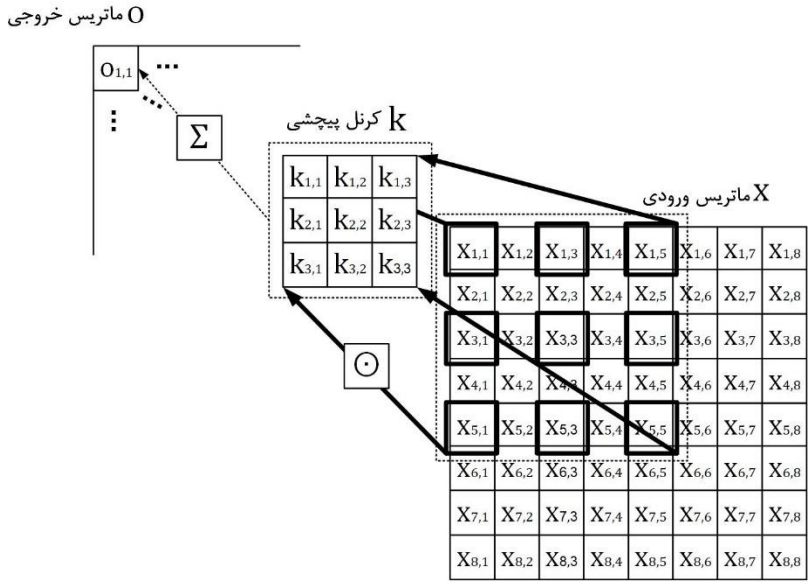
با توجه به توضیحات برای افزایش بیشتر ابعاد خروجی عملیات ضرب پیچشی ترانهاده گام

[1] Deconvolution
[2] Fractional stride convolution

حرکتی آن را به صورت کسری، مقادیر کوچک تر از یک، تعریف می کنیم در صورتی که امکان پیاده سازی آن با تعریف اصلی ضرب ترانهاده وجود نداشته و رابطه ۵۱-۱۲ صرفا برای گام حرکتی یک صدق می کند. برای تعریف گام های حرکتی کسری برای عملیات ضرب پیچشی ترانهاده باید آن را به ضرب پیچشی معادل تبدیل کرده و سپس با استفاده از ایده ای مشابه اعمال اتساع به ضرب پیچشی که در ادامه بررسی می کنیم گام های حرکتی مذکور را تعریف کنیم. بنابراین تبدیل ضرب پیچشی ترانهاده به ضرب پیچشی معادل علاوه کاهش پیچیدگی پیاده سازی، امکان تعریف گام های حرکتی کسری را نیز فراهم می کند.

۱۲.۱۰.۱ ضرب پیچشی دو بعدی با اتساع[1]

روند این عملیات مشابه ضرب پیچشی دو بعدی است با این تفاوت که یک پارامتر اسکالر طبیعی به نام نرخ اتساع[2] برای آن تعریف می شود. در ضرب پیچشی دو بعدی هنگام اجرای عملیات ضرب در هر گام حرکتی، ابعاد مختلف کرنل مطابق شکل ۱-۱۲ رو به روی بازه ای متوالی از درایه های ماتریس ورودی قرار می گرفتند، در حالی که با اعمال اتساع به ضرب پیچشی در هر گام حرکتی درایه های ورودی آن گام به صورت بازه متوالی نبوده و با فاصله ای برابر با نرخ اتساع در بین آن ها به صورت شکل ۱۴-۱۲ انتخاب می شوند. معمولا نرخ اتساع، تعداد فواصل بین درایه های هر گام، برای جهات مختلف، افقی و عمودی، برابر است. در شکل ۱۴-۱۲ نرخ اتساع برابر یک است.



شکل ۱۴-۱۲: ضرب پیچشی دو بعدی با اتساع

[1] Dilated convolution
[2] Dilation rate

ابعاد ماتریس خروجی در صورت اعمال اتساع به ترتیب برای طول و عرض ماتریس مطابق رابطه ۱۲-۵۷ و ۱۲-۵۸ محاسبه می شود، در این روابط d نرخ اتساع، و بقیه پارامترها مشابه روابط ۱۲-۴ و ۱۲-۵ است.

$$l_1 = \frac{n-(g+(g-1)d)+p_1}{s_1}+1 \tag{۱۲-۵۷}$$

$$l_2 = \frac{m-(h+(h-1)d)+p_2}{s_2}+1 \tag{۱۲-۵۸}$$

رابطه پیشرو برای یک ضرب پیچشی دو بعدی در صورت اعمال اتساع مطابق رابطه ۱۲-۵۹ است.

$$o_{i,j} = \sum_{p=0}^{g-1}\sum_{t=0}^{h-1} x_{(pd+1)i+p,(td+1)j+t} k_{p+1,t+1} =$$

$$sum(X_{(i:d+1:g+(g-1)d,\, j:d+1:h+(h-1))} \odot K), i = 1,...,l_1,\ j = 1,...,l_2 \tag{۱۲-۵۹}$$

بدین ترتیب برای ضرب پیچشی دو بعدی با اتساع مشتقات زنجیره ای مطابق روابط ۱۲-۶۰ تا ۱۲-۶۲ تعریف می شود.

$$\frac{\partial E}{\partial k_{r,q}} = \frac{\partial E}{\partial \mathbf{e}}_{e} \sum_{i=1}^{l_1}\sum_{j=1}^{l_2} \frac{\partial \mathbf{e}}{\partial o_{i,j}}_{-1} \frac{\partial o'_{i,j}}{\partial o_{i,j}}_{f'(o_{i,j})} \frac{\partial o_{i,j}}{\partial k_{r,q}} \tag{۱۲-۶۰}$$

$$\frac{\partial o_{i,j}}{\partial k_{r,q}} = \frac{\partial}{\partial k_{r,q}} (sum(X_{(i:d+1:g+(g-1)d,\, j:d+1:h+(h-1))} \odot K)) \tag{۱۲-۶۱}$$

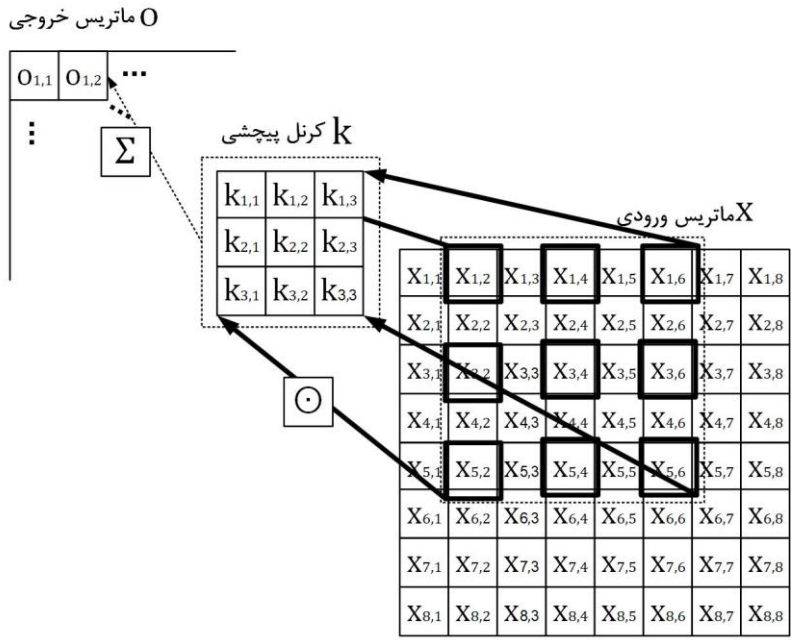
$$\frac{\partial}{\partial k_{r,q}} (sum(X_{(i:d+1:g+(g-1)d,\, j:d+1:h+(h-1))} \odot K)) =$$

$$\frac{\partial}{\partial k_{r,q}} (k_{1,1}x_{i,j} + k_{1,2}x_{i,j+d+1} + k_{r,q}x_{i+(r-1)(d+1),\, j+(q-1)(d+1)} + ...) =$$

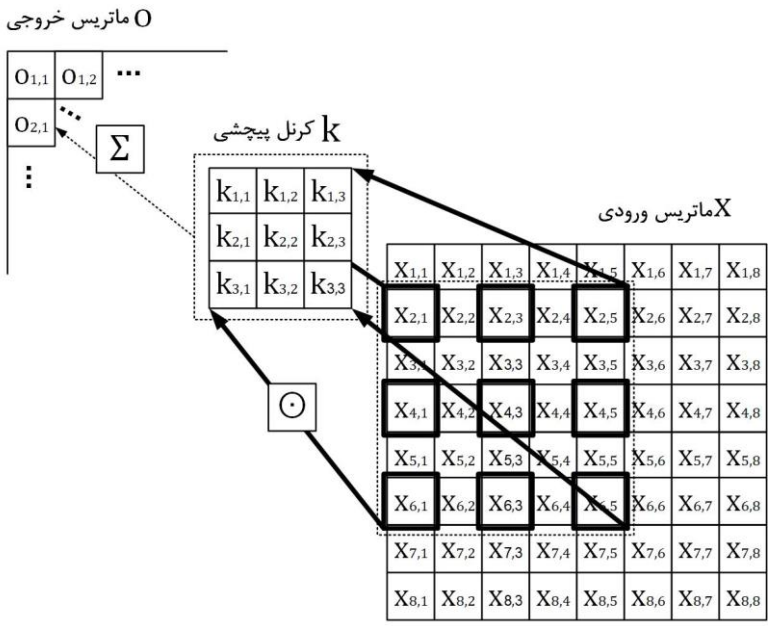
$$x_{i+(r-1)(d+1),\, j+(q-1)(d+1)} \tag{۱۲-۶۲}$$

در روابط ۱۲-۵۹ تا ۱۲-۶۲ منظور از زیروند $i:d+1:g+(g-1)$ و $j:d+1:h+(h-1)$

انتخاب درایه هایی از ماتریس ورودی است که در عملیات ضرب پیچشی با اتساع دخیل می شوند که در اینجا به صورت یک دنباله که ابتدای آن به ترتیب i, j، انتهای آن به ترتیب $g + (g - 1)$ و $h + (h - 1)$ و گام آن ها $d + 1$ است نشان داده شده اند. ادامه روند ضرب پیچشی با اتساع، مشابه روند ضرب پیچشی بوده و محل درایه های دخیل در ضرب در هر گام حرکتی به اندازه گام افقی یا عمودی تعریف شده مشابه شکل های ۱۵-۱۲ و ۱۶-۱۲ جا به جا می شوند. در شکل های ۱۵-۱۲ و ۱۶-۱۲ گام حرکتی افقی و عمودی یک تعریف شده است.



شکل ۱۵-۱۲: ادامه روند ضرب پیچشی دو بعدی با اتساع در جهت افقی



شکل ۱۶-۱۲: ادامه روند ضرب پیچشی دو بعدی با اتساع در جهت عمودی

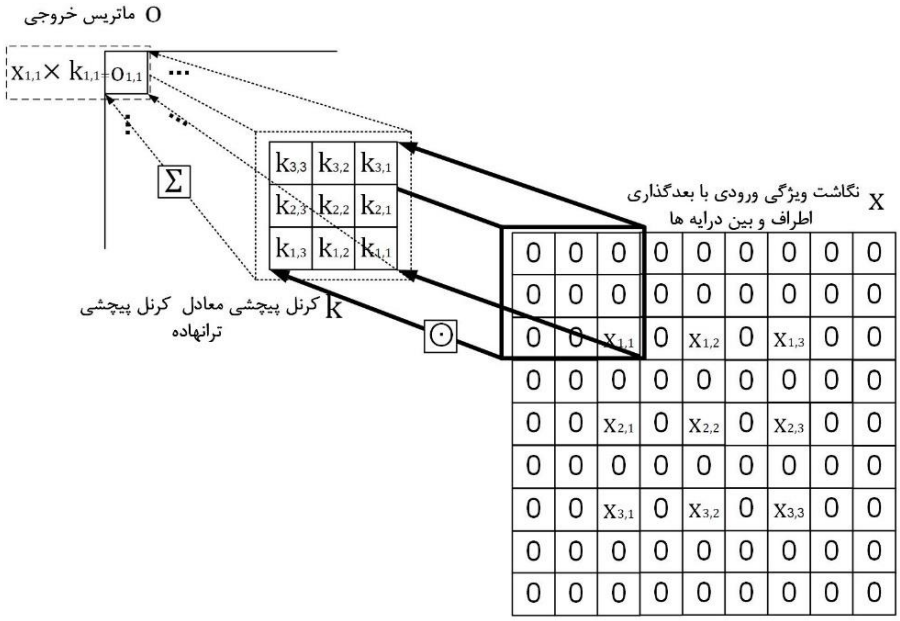
۱۲.۱۰.۲ ضرب پیچشی دو بعدی ترانهاده به عنوان ضرب پیچشی

همان طور که اشـاره کردیم در روند اجرای عملیات ضـرب پیچشـی دو بعدی ترانهاده، مقادیر برخی از درایه های ماتریس خروجی که در چند گام حرکتی هم پوشـانی دارند به صـورت مجموع مقادیر محاسـبه شـده به ازای همه گام ها در نظر گرفته می شـود، در صورت وجود مقادیری از قبل در درایه های ماتریس خروجی، مقادیر جدید در هر گام حرکتی با مقادیر قبلی جمع می شوند. این موضوع سبب دشواری روند محاسبه مشتقات زنجیره ای برای آموزش کرنل ضرب پیچشی دو بعدی ترانهاده و نیز پارامتر های لایه های پیش از آن می شـود، زیرا لازم اسـت خروجی مربوط به هر گام حرکتی از این مقدار مجموع نهائی جدا شده و محاسبات مشتقات به ازای آن محاسبه شود که برای تحقق آن باید مقدار خروجی هر گام حرکتی جداگانه ذخیره شـــود تا امکان تجزیه مقدار نهائی به مقادیر هر گام وجود داشـــته باشـــد. علاوه بر این همان طور که اشـــاره کردیم گام های حرکتی در ضرب پیچشی دو بعدی ترانهاده به صورت کسری نیز تعریف می شوند در صورتی که امکان پیاده سـازی آن در روشـی که برای ضـرب پیچشـی ترانهاده تعریف کردیم وجود ندارد. بدین ترتیب برای سـهولت پیاده سازی و محاسـبه مشـتقات زنجیره ای، ضرب پیچشی دو بعدی ترانهاده را همان طور که پیش تر اشاره کردیم به صورت ضرب پیچشی دو بعدی پیاده سازی می کنیم. اما با تبدیل ضرب پیچشـــی ترانهاده دو بعدی به ضـــرب پیچشـــی دو بعدی همچنان امکان تعریف گام های حرکتی کســـری برای ایجاد خروجی با ابعاد برزگ تر وجود ندارد، برای رفع این مشـــکل به ورودی ضـــرب

پیچشی معادل ضرب پیچشی ترانهاده بعد های صفر[1] اعمال کرده و مشابه روش اتساع بین درایه های ورودی دخیل در هر گام آموزشی فاصله ایجاد می کنیم شکل ۱۷-۱۲ اعمال این بعدها را به نگاشت ویژگی مطابق شکل ۱۳-۱۲ را نشان می دهد. تعداد بعد های لازم بین سطر ها و ستون های نگاشت ویژگی ورودی برای تعریف گام حرکتی کسری به ترتیب از روابط ۶۳-۱۲ و ۶۴-۱۲ محاسبه می شود، در این روابط n همان مخرج گام کسری تعریف شده برای ضرب پیچشی ترانهاده دو بعدی است. در شکل ۱۷-۱۲ گام حرکتی در هر دو جهت افقی و عمودی $\frac{1}{2}$ تعریف شده است.

$$\frac{1}{n_{horizental}} \rightarrow d_{horizental} = n_{horizental} - 1 \qquad (۱۲-۶۳)$$

$$\frac{1}{n_{vertical}} \rightarrow d_{vertical} = n_{vertical} - 1 \qquad (۱۲-۶۴)$$



شکل ۱۷-۱۲: ضرب پیچشی معادل ضرب پیچشی ترانهاده با گام حرکتی کسری

۱۲.۱۱ ضرب پیچشی سه بعدی[2]

همان طور که در معرفی ساختارهای پیچشی دو بعدی، روابط و تعاریف مطرح شده برای ضرب پیچشی یک بعدی را به دو بعد تعمیم دادیم، می توانیم با تعمیم این روابط به سه بعد ساختارهای پیچشی سه بعدی را تعریف کرده و لایه های پیچشی، تجمیع، پیچشی ترانهاده و... را در سه بعد

[1] Zero pads
[2] 3-D convolution

توسعه دهیم. در ضرب پیچشی دو بعدی برای عملیات ضرب پیچشی یک کرنل دو بعدی به صورت ماتریس تعریف می کردیم، این کرنل در ضرب پیچشی سه بعدی به صورت یک تانسور سه بعدی تعریف شده و رابطه پیشرو آن مطابق رابطه ۶۵-۱۲ خواهد بود.

$$o_{i,j,k} = \sum_{u=0}^{w-1}\sum_{p=0}^{g-1}\sum_{t=0}^{h-1} x_{i+p,j+t,k+u} k_{p+1,t+1,u+1} =$$
$$sum(\mathbf{X}_{(i:i+g-1,\, j:j+h-1, k:k+w-1)} \odot \mathbf{K})$$
$$i = 1,\ldots,l_1,\ j = 1,\ldots,l_2, k = 1,\ldots,l_3 \qquad (۱۲-۶۵)$$

در معرفی تعریف لایه پیچشی دو بعدی بررسی کردیم که اگر ورودی لایه به صورت تانسور سه بعدی باشد هر یک از کرنل های ماتریسی لایه پیچشی به طور جداگانه در هر یک از ابعاد تانسور ضرب شده و در نهایت عمق تانسور سه بعدی خروجی برای یک لایه پیچشی دو بعدی متشکل از n کرنل دو بعدی با یک تانسور ورودی سه بعدی با عمق z، برابر $z \times n$ خواهد بود. در صورت تعریف یک لایه پیچشی سه بعدی متشکل از کرنل های سه بعدی هر کرنل سه بعدی در یک عملیات ضرب پیچشی سه بعدی در طول عمق تانسور ورودی نیز ضرب شده و عمق تانسور سه بعدی خروجی حاصل از ضرب پیچشی تانسور سه بعدی ورودی با عمق z برابر z خواهد بود. ابعاد مربوط به دو بعد اول تانسور خروجی، طول و عرض، از روابط ۴-۱۲ و ۵-۱۲، مشابه ضرب پیچشی دو بعدی و اندازه بعد سوم، عمق، از رابطه ۶۶-۱۲ محاسبه می شود که در این رابطه p_3, s_3, w, z, l_3 به ترتیب اندازه عمق خروجی، اندازه عمق ورودی، عمق کرنل سه بعدی، گام حرکتی بعد سوم و ابعاد افزوده به روش بعدگذاری در بعد سوم هستند.

$$l_3 = \frac{z - w + p_3}{s_3} + 1 \qquad (۱۲-۶۶)$$

برای آموزش کرنل ضرب پیچشی سه بعدی روابط مشتقات زنجیره ای مطابق روابط ۶۷-۱۲ تا ۶۹-۱۲ تعریف می شوند، در این روابط برای ساده سازی از مجموع مربعات خطا به عنوان تابع هزینه استفاده شده و همچنین خطای $\boldsymbol{e}$ به صورت یک تانسور هم اندازه با تانسور خروجی لایه $\boldsymbol{\acute{o}}$ در نظر گرفته شده است.

$$\frac{\partial E}{\partial k_{r,q,v}} = \frac{\partial E}{\partial \mathbf{e}}_{e} \sum_{k=1}^{l_3}\sum_{i=1}^{l_1}\sum_{j=1}^{l_2} \frac{\partial \mathbf{e}}{\partial o'_{i,j,k}}_{-1} \frac{\partial o'_{i,j,k}}{\partial o_{i,j,k}}_{f'(o_{i,j})} \frac{\partial o_{i,j,k}}{\partial k_{r,q,v}} \qquad (۱۲-۶۷)$$

$$\frac{\partial o_{i,j,k}}{\partial k_{r,q,v}} = \frac{\partial}{\partial k_{r,q,v}}(sum(\mathbf{X}_{(i:i+g-1,\,j:j+h-1,k:k+w-1)} \odot \mathbf{K})) \tag{۱۲-۶۸}$$

$$\frac{\partial}{\partial k_{r,q,v}}(sum(\mathbf{X}_{(i:i+g-1,\,j:j+h-1,k:k+w-1)} \odot \mathbf{K})) =$$

$$\frac{\partial}{\partial k_{r,q,v}}(k_{1,1,1}x_{i,j,k} + k_{1,2,1}x_{i,j+1,k} + k_{r,q,v}x_{i+r-1,j+q-1,k+v-1} + ...) =$$

$$x_{i+r-1,j+q-1,k+v-1} \tag{۱۲-۶۹}$$

بنابر روابط فوق مقادیر درایه های کرنل سه بعدی در هر گام آموزشی مطابق رابطه ۱۲-۷۰ محاسبه می شود:

$$k_{r,q,v(k+1)} = k_{r,q,v(k)} - \eta \frac{\partial E}{\partial k_{r,q,v}}_{(k)} \tag{۱۲-۷۰}$$

برای آموزش لایه های پیش از لایه پیچشی سه بعدی مشتق خروجی نسبت به ورودی آن را به صورت رابطه ۱۲-۷۱ تعریف می کنیم:

$$\frac{\partial E}{\partial \mathbf{e}}_{e} \sum_{k=1}^{w}\sum_{i=1}^{l_1}\sum_{j=1}^{l_2} \frac{\partial \mathbf{e}}{\partial o_{i,j,k}} \frac{\partial o'_{i,j,k}}{\partial o_{i,j,k}} \frac{\partial o_{i,j,k}}{\partial x_{r,q,v}} \frac{\partial x_{r,q,v}}{\dots} \dots \tag{۱۲-۷۱}$$

$$\frac{\partial o_{i,j,k}}{\partial x_{r,q,v}} = \frac{\partial}{\partial x_{r,q,v}}(sum(\mathbf{X}_{(i:i+g-1,\,j:j+h-1,k:k+w-1)} \odot \mathbf{K})) \tag{۱۲-۷۲}$$

$$\frac{\partial}{\partial x_{r,q,v}}(k_{1,1,1}x_{i,j,k} + k_{1,2,1}x_{i,j+1,k} + k_{s,d,f}x_{r,q,v} + ...) =$$

$$\begin{cases} k_{s,d,f}, x_{r,q,v} \in \mathbf{X}_{(i:i+g-1,\,j:j+h-1,k:k+w-1)} \\ 0, x_{r,q,v} \notin \mathbf{X}_{(i:i+g-1,\,j:j+h-1,k:k+w-1)} \end{cases} \tag{۱۲-۷۳}$$

در لایه های پیچشی سه بعدی می توان از تصاویر رنگی که به صورت تانسور سه بعدی با عمق

سه، در برخی موارد چهار[1]، ذخیره می شوند به عنوان ورودی استفاده کرد، همچنین از این عملیات می توانیم برای پردازش تصاویر سه بعدی که به صورت یک تانسور سه بعدی از واکسل[2]، تصاویر سه بعدی سیاه و سفید، و یا تانسور های چهار بعدی، تصاویر سه بعدی متشکل از واکسل های رنگی، ذخیره می شوند استفاده کنیم؛ در صورتی که تانسور ورودی در چهار بعد به صورت $m \times n \times z \times u$ تعریف شده باشد، کرنل سه بعدی عملیات ضرب پیچشی سه بعدی به طور جداگانه در هر یک از ابعاد بعد چهارم،u، تانسور ضرب شده و در نهایت تشکیل یک تانسور چهار بعدی به عنوان خروجی که اندازه بعد چهارم آن u است می دهد، در صورت استفاده از چند، b، کرنل سه بعدی به صورت یک لایه لندازه بعد چهارم خروجی $u \times b$ خواهد بود. این عملیات همچنین می تواند در پردازش توالی زمانی تصاویر، ویدئو، که در راستای بعد زمان مشابه دادگان سری زمانی در کنار هم به صورت یک تانسور چهار بعدی قرار گرفته اند نیز به کار برده شوند که در ادامه به بررسی آن ها خواهیم پرداخت. یک ویدئو به صورت یک تانسور چهار بعدی که سه بعد آن مربوط به کانال های رنگی هر تصویر که در گام زمانی مشخص ثبت شده و بعد چهارم مربوط به زمان است، ذخیره می شود. در صورتی که ویدئو سیاه سفید باشد به صورت یک تانسور سه بعدی ذخیره می شود که دو بعد آن مربوط به تصویر در هر گام زمانی بوده و بعد سوم آن نیز زمان است، همچنین در صورتی که تصاویر دارای چهار کانال رنگی باشند، ویدئو به صورت تانسور پنج بعدی خواهد بود.

همان طور که یک عملیات ضرب پیچشی را از یک بعد به دو و سه بعد تعمیم دادیم می توانیم با ادامه این روند تعمیم عملیات ضرب پیچشی n-بعدی نیز تعریف کنیم که در آن کرنل پیچشی به صورت یک تانسور n-بعدی تعریف می شود، در این صورت این عملیات را می توان برای تانسور ورودی با بعد بزرگ تر یا مساوی n، بعد تانسور ورودی$\leq n$، به کار برد.

۱۲.۱۱.۱ بعد گذاری در ضرب پیچشی سه بعدی

تعداد بعد های لازم در جهت عمق، بعد سوم، از در ضرب پیچشی سه بعدی،n-بعدی، از رابطه ۱۲-۷۲ محاسبه می شود. که در آن است. اعمال بعدگذاری در راستای بعد سوم برای یک تصویر رنگی که به صورت یک تانسور سه بعدی است همانند افزودن یک کانال دیگر به تصویر است، به طوری که اندازه همه ابعاد آن صفر باشد. اندازه بعدگذاری دو بعد اول همانند ضرب پیچشی دو بعدی از روابط ۱۲-۱۶ تا ۱۲-۱۹ محاسبه می شود. رابطه ۱۲-۷۴ را می توانیم برای ضرب پیچشی تعریف شده برای ابعاد بزرگ تر از سه به صورت رابطه ۱۲-۷۵ تعمیم دهیم که در آن اندازه بعد n-ام کرنل پیچشی است.

$$p_3 = w - 1 \qquad (۱۲-۷۴)$$

[1] تصاویر CMYK

[2] Voxel

$$p_n = n_k - 1 \tag{۱۲-۷۵}$$

از روابط ۱۲-۷۴ و ۱۲-۷۵ تعداد بعدگذاری به ازای کل یک بعد از تانسور محاسبه می شود که از آن ها مطابق روابط ۱۲-۷۶ و ۱۲-۷۷ برای محاسبه تعداد بعدهای لازم در ابتدا و انتهای بعد مورد نظر استفاده می کنیم.

$$p'_3 = \frac{p_3}{2} \tag{۱۲-۷۶}$$

$$p'_n = \frac{p_n}{2} \tag{۱۲-۷۷}$$

همان طور که در روابط ضرب پیچشی دو بعدی نیز اشاره کردیم، تعداد ابعاد، در روش بعدگذاری می تواند لزوما از روابطی مشابه روابط ۱۲-۷۴ و ۱۲-۷۵ محاسبه نشده و دارای مقادیر دلخواه نسبت به کاربرد مد نظر داشته باشد.

۱۲.۱۱.۲ لایه تجمیع سه بعدی[1]

همان طور که اشاره کردیم می توانیم تعریف مربوط به لایه های تجمیع را نیز به سه بعد، n-بعد، تعمیم دهیم، در این صورت رابطه پیشرو برای یک لایه تجمیع حداکثری و میانگین سه بعدی به ترتیب به صورت روابط ۱۲-۷۸ و ۱۲-۷۹ خواهد بود.

$$o_{i,j,k} = \arg\max \mathbf{X}_{(i:i+g-1,\, j:j+h-1, k:k+w-1)} \tag{۱۲-۷۸}$$

$$o_{i,j,k} = \frac{1}{g \times h \times w} sum(\mathbf{X}_{(i:i+g-1,\, j:j+h-1, k:k+w-1)}) \tag{۱۲-۷۹}$$

۱۲.۱۱.۳ لایه تجمیع سراسری سه بعدی[2]

در حالت کلی می توان لایه تجمیع سراسری به صورت m-بعدی تعریف کرد، که در آن همه مقادیر موجود در m بعد اول تانسور ورودی با بعد بزرگ تر یا مساوی n، بعد تانسور ورودی$= m \leq n$، میانگین گیری شده[3]، یا مقدار حداکثری از میان آن ها انتخاب شده[4]، و یک تانسور $m - n$ بعدی به عنوان خروجی لایه تشکیل می شود. بنابراین می توانیم این عملیات، لایه، را به صورت سه بعدی نیز تعریف کنیم، در صورتی که خروجی ساختار پیچشی به صورت تانسور چهاربعدی باشد، اعمال لایه تجمیع سراسری سه بعدی به آن، حاصلی به صورت یک بردار داشته و می توان آن را به عنوان ورودی یک ساختار مرسوم مانند شبکه عصبی پرسپترون چند لایه استفاده

[1] 3-D pooling
[2] 3-D global pooling
[3] N-D global average pooling (GAP)
[4] N-D global max pooling (GMP)

کرد. همانند لایه تجمیع سراسری دو بعدی می توان، لایه تجمیع n-بعدی را به صورت حداکثری نیز تعریف کرد که استفاده از آن نسبت به روش میانگین محدود تر است.

۱۲.۱۱.۴ اتساع سه بعدی

اعمال اتساع به یک کرنل پیچشی را می توانیم به کرنل های n-بعدی نیز تعمیم دهیم. همان طور که در اعمال اتساع به یک کرنل دو بعدی میان درایه های کرنل در دو جهت افقی و عمودی فاصله ایجاد می شد، در صورت اعمال آن به یک کرنل سه بعدی، n-بعدی، نیز بین درایه های آن در امتداد سه بعد، n-بعد، فواصلی به اندازه نرخ اتساع اعمال می شود. البته می توان بسته به کاربرد در مواردی، نرخ اتساع برای هر بعد مقدار متفاوتی داشته و یا برای برخی از ابعاد اعمال نشود.

۱۲.۱۲ ضرب پیچشی جدائی پذیر فضائی[1]

فرض کنیم برای یک عملیات ضرب پیچشی دو بعدی کرنل تعریف شده به صورت رابطه ۱۲-۸۰ باشد. در صورت اجرای روند ضرب پیچشی با کرنل تعریف شده برای یک ورودی دو بعدی به شکل ماتریس رابطه ۱۲-۸۱، در هر گام حرکتی با قرار گرفتن کرنل رو به روی تعدادی از درایه های ماتریس ورودی، یک عملیات ضرب درایه به درایه، هادامارد، بین دو ماتریس ۳×۳ تعریف می شود.

$$K=\begin{bmatrix}1 & 3 & 10\\ 2 & 6 & 20\\ 3 & 9 & 30\end{bmatrix}_{3\times 3} \tag{۱۲-۸۰}$$

$$X=\begin{bmatrix}x_{1,1} & x_{1,2} & \cdots & x_{1,n}\\ x_{2,1} & \ddots & \cdots & \vdots\\ \vdots & \cdots & \ddots & \vdots\\ x_{m,1} & \cdots & \cdots & x_{m,n}\end{bmatrix}_{m\times n} \tag{۱۲-۸۱}$$

کرنل رابطه ۱۲-۸۰ را می توانیم به صورت حاصل ضرب دو ماتریس تجزیه کرده و مطابق رابطه ۱۲-۸۲ نیز نشان دهیم.

[1] Spatial separable convolution

$$K=\begin{bmatrix}1 & 3 & 10\\ 2 & 6 & 20\\ 3 & 9 & 30\end{bmatrix}_{3\times3}=\begin{bmatrix}1\\ 2\\ 3\end{bmatrix}_{3\times1}\times\begin{bmatrix}1 & 3 & 10\end{bmatrix}_{1\times3}=K_1\times K_2 \qquad (۱۲-۸۲)$$

فرض کنیم در یک گام حرکتی ضرب پیچشی، ضرب کرنل K در قسمتی از درایه های ماتریس ورودی X و محاسبه بعد معادل خروجی به صورت رابطه ۱۲-۸۳ باشد.

$$o_{i,j}=X_{i:i',j:j'}*K=sum(X_{i:i',j:j'}\odot K)=$$
$$sum(\begin{bmatrix}5 & 7 & 17\\ 2 & 1 & 12\\ 3 & 0 & 4\end{bmatrix}_{3\times3}\odot\begin{bmatrix}1 & 3 & 10\\ 2 & 6 & 20\\ 3 & 9 & 30\end{bmatrix}_{3\times3})=$$
$$sum(\begin{bmatrix}5 & 21 & 170\\ 4 & 6 & 240\\ 9 & 0 & 120\end{bmatrix}_{3\times3})=575 \qquad (۱۲-۸۳)$$

اگر روند رابطه ۱۲-۸۳ را در دو مرحله ضـــرب پیچشـــی متوالی، با ماتریس های رابطه ۱۲-۸۲ اجرا کنیم روابط ۱۲-۸۴ و ۱۲-۸۵ را خواهیم داشت.

$$X'_{i,j:j'}=X_{i:i',j:j'}*K_1=$$
$$sum_{column-wise}(\begin{bmatrix}5 & 7 & 17\\ 2 & 1 & 12\\ 3 & 0 & 4\end{bmatrix}_{3\times3}\times\begin{bmatrix}1\\ 2\\ 3\end{bmatrix}_{3\times1})=$$
$$sum_{column-wise}(\begin{bmatrix}5 & 7 & 17\\ 4 & 2 & 24\\ 9 & 0 & 12\end{bmatrix}_{3\times3})=\begin{bmatrix}18 & 9 & 53\end{bmatrix}_{1\times3} \qquad (۱۲-۸۴)$$

$$o_{i,j} = X'_{i,j:j'} * K_2 =$$
$$sum(\begin{bmatrix}18 & 9 & 53\end{bmatrix}_{1\times3} \odot \begin{bmatrix}1 & 3 & 10\end{bmatrix}_{1\times3}) =$$
$$sum(\begin{bmatrix}18 & 27 & 530\end{bmatrix}_{1\times3}) = 575 \quad (۱۲-۸۵)$$

ضرب پیچشی در رابطه ۱۲-۸۴ با توجه به ابعاد کرنل به صورت ضرب بردار در ماتریس تعریف می شود که معادل سه مرحله ضرب درایه به درایه هر یک از ستون های ورودی در کرنل است. با مقایسه حاصل رابطه ۱۲-۸۳ و ۱۲-۸۵ مشاهده می کنیم که نتایج حاصل از اعمال یک ماتریس ۳ × ۳ به عنوان کرنل پیچشی دو بعدی به ورودی همانند اعمال دو ماتریس حاصل از تجزیه آن مطابق ۱۲-۸۲ در دو عملیات متوالی ضرب پیچشی است. با وجود یکسان بودن نتایج حاصل از هر دو روش، در روش دوم عملیات تعریف شده در هر گام به صورت ضرب درایه به درایه دو ماتریس ۱×۳ یا ۳×۱ است در حالی که هر گام در روش اول به صورت ضرب درایه به درایه دو ماتریس ۳×۳ تعریف می شود. با توجه به اینکه پیچیدگی محاسباتی در ضرب درایه به درایه به تعداد درایه های ماتریس وابسته است برای روش اول مرتبه آن معادل $O(m \times n) = O(m \times m) = O(m^2), m = n = 3$ است در حالی که در روش دوم این مرتبه معادل $O(m \times n) = O(m \times 1) = O(m), m = 3, n = 1$ است. بدین ترتیب در روش دوم پیچیدگی محاسباتی کمتر بوده و در نتیجه روند اجرای شبکه متشکل از این عملیات سرعت بیشتری خواهد داشت. به این حالت تجزیه یک عملیات ضرب پیچشی دو بعدی به دو عملیات متوالی، ضرب پیچشی جدائی پذیر فضائی گفته می شود، این روش صرفا در ضرب پیچشی دو بعدی تعریف می شود. علاوه بر کاهش پیچیدگی محاسباتی در روش ضرب پیچشی جدائی پذیر فضائی از آن جایی که در هر یک از دو مرحله متوالی ضرب پیچشی بعدگذاری ماتریس ورودی با توجه به ابعاد کرنل و روابط ۱۲-۱۶ و ۱۲-۱۷، فقط در یک بعد مورد نیاز است تعداد کل ضرب های اجرا شده، ضرب اسکالر در اسکالر، در روش دوم حدود ۱۰ درصد کمتر از روش اول است.

با وجود عملکرد مناسب استفاده از روش ضرب پیچشی جدائی پذیر فضائی، در بسیاری از موارد مقادیر درایه های ماتریس کرنل به گونه ای است که امکان تجزیه ماتریس آن به دو ماتریس سطری و ستونی مطابق رابطه ۱۲-۸۲ وجود ندارد. همچنین با توجه به تغییر مقادیر درایه های کرنل ها در روند آموزش یک ساختار امکان استفاده از روش ضرب پیچشی جدائی پذیر در روند آموزش یک ساختار پیچشی نبوده، به دلیل لزوم بررسی امکان تجزیه ماتریس هر کرنل، و باید پس از اتمام مرحله آموزش ساختار در صورت امکان تجزیه مقادیر هر یک از کرنل های ساختار به دو ماتریس سطری و ستونی، در مرحله تست[1] و اعتبارسنجی[2] برای برخی از کرنل های مدل که می توان همانند رابطه ۱۲-۸۲ آن ها را به دو ماتریس سطری و ستونی تبدیل کرد از این روش استفاده

[1] Test
[2] Validation

کنیم. بنابراین علیرغم عملکرد مناسب استفاده از این روش در کاهش محاسبات موارد استفاده از آن بسیار محدود بوده و عملا در ساختارهای پیچشی مرسوم از آن استفاده نمی شود.

۱۲.۱۳ ضرب پیچشی جدائی پذیر مبتنی بر عمق[1]

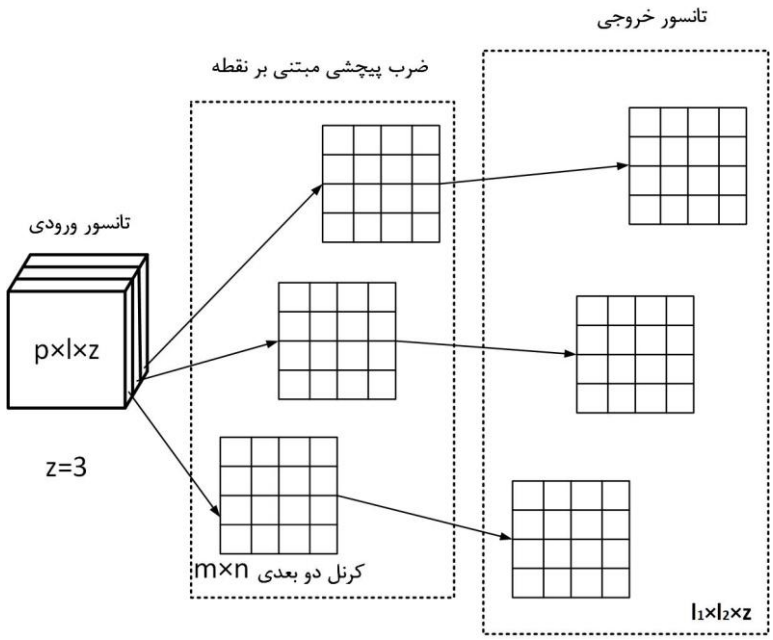
در بسیاری از ساختارهای پیچشی مرسوم که در ادامه معرفی خواهیم کرد از لایه های پیچشی سه بعدی استفاده می شود، در این لایه های پیچشی سه بعدی عمق کرنل سه بعدی هم اندازه عمق تانسور ورودی در نظر گرفته شده و در هر گام حرکتی تانسور کرنل فقط در دو جهت طول و عرض حرکت می کند. در صورت تعریف یک لایه پیچشی سه بعدی در یک ساختار می توانیم با استفاده از روش ضرب پیچشی جدائی پذیر مبتنی بر عمق همانند روش ضرب پیچشی جدائی پذیر فضائی، با تجزیه عملیات پیچشی سه بعدی به عملیات های پیچشی مبتنی بر عمق[2] و مبتنی بر نقطه[3] پیچیدگی و حجم محاسبات را تا حد زیادی کاهش دهیم. روش ضرب پیچشی جدائی پذیر مبتنی بر عمق برخلاف روش ضرب پیچشی جدائی پذیر فضائی، به مقادیر درایه های کرنل پیچشی وابسته نبوده و از این رو می توان آن را به طور گسترده در لایه ای پیچشی سه بعدی و در همه مراحل، آموزش، تست و...، استفاده کنیم. همچنین برخلاف روش ضرب پیچشی جدائی پذیر فضائی که صرفا محدود به ضرب پیچشی دو بعدی بود، از روش ضرب پیچشی جدائی پذیر مبتنی بر عمق می توانیم در لایه های پیچشی سه بعدی و بالاتر از آن استفاده کرده از حجم محاسبات ضرب مرسوم پیچشی بکاهیم.

فرض کنیم یک لایه پیچشی سه بعدی متشکل از u کرنل سه بعدی با ابعاد $m \times n \times z$ در یک ساختار تعریف شده و ورودی آن لایه نیز به صورت یک تانسور سه بعدی با عمقی برابر با عمق کرنل های لایه به صورت $p \times l \times z$ باشد. در صورت اجرای روند پیشرو این لایه به صورت مرسوم، برای هر یک از کرنل های سه بعدی لایه، یک ضرب پیچشی با تانسور ورودی تعریف می شود. در این صورت ابعاد تانسور خروجی هر یک از این ضرب های پیچشی سه بعدی $l_1 \times l_2 \times 1$ خواهد بود، که مقادیر l_1 و l_2 از روابط ۴-۱۲ و ۵-۱۲ مقدار یک برای بعد سوم از رابطه ۶۶-۱۲ محاسبه می شوند از آنجائی که لایه شامل u کرنل است در نهایت تانسور خروجی دارای ابعادی برابر با $l_1 \times l_2 \times u$ خواهد بود. در این صورت در روند پیشرو این لایه در هرگام حرکتی هریک از این کرنل ها بر روی تانسور ورودی $p \times l \times z$ عملیات ضرب، ضرب اسکالر در اسکالر، اجرا می شود، این تعداد برای یک گام حرکتی به ازای همه کرنل های لایه برابر $p \times l \times z \times u$ ضرب خواهد بود.

برای پیاده سازی این روند با استفاده از روش ضرب پیچشی جدائی پذیر مبتنی بر عمق، ابتدا مقادیر هر یک از کرنل های سه بعدی را در راستای عمق تجزیه کرده و به صورت z ماتریس با ابعاد

[1] Depth wise separable convolution
[2] Depth wise convolution
[3] Point wise convolution

$m \times n$ در نظر می گیریم که به هر یک از ماتریس ها کرنل های پیچشی مبتنی بر نقطه[1] گفته می شود. برای هر یک از این ماتریس ها با بعد متناظر خود در تانسور ورودی یک ضرب پیچشی دو بعدی مطابق شکل ۱۸-۱۲ تعریف می شود؛ باید در نظر داشته باشیم که این ضرب متفاوت از عملیات لایه پیچشی دو بعدی است که در آن هر یک از کرنل های دو بعدی لایه پیچشی در همه عمق های ورودی ضرب می شود. با اعمال این عملیات ضرب در نهایت خروجی یک کرنل مبتنی بر نقطه به صورت یک تانسور با ابعاد $l_1 \times l_2 \times z$ خواهد بود. به این قسمت از عملیات، ضرب پیچشی مبتنی بر عمق گفته می شود. مجموع ضرب های اجرا شده هر گام حرکتی این مرحله به ازای همه کرنل های دو بعدی حاصل از تجزیه کرنل سه بعدی است.

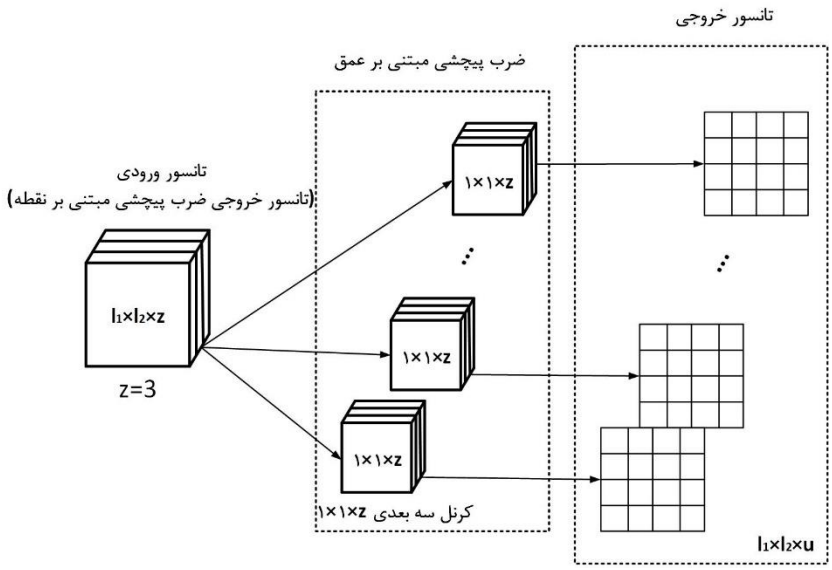


شکل ۱۸-۱۲: ضرب پیچشی مبتنی بر نقطه

تانسور خروجی مرحله قبل به عنوان ورودی مرحله ضرب پیچشی مبتنی بر نقطه استفاده می شود، در این مرحله تعداد u تانسور با ابعاد $z \times 1 \times 1$ به عنوان کرنل ضرب پیچشی مبتنی بر عمق[2] تعریف شده و برای هر کدام از آن ها در تانسور خروجی ضرب پیچشی مبتنی بر نقطه یک ضرب پیچشی سه بعدی تعریف با گام حرکتی یک در جهت طول و عرض تعریف می شود. خروجی هر کدام از این ضرب ها مطابق شکل ۱۹-۱۲ یک تانسور با ابعاد $1 \times l_1 \times l_2$ که با کنار هم قرار دادن خروجی همه کرنل های یک تانسور با ابعاد $l_1 \times l_2 \times u$ حاصل می شود که با تانسور خروجی

[1] Point wise convolutional kernels
[2] Depth wise convolutional kernel

ضـرب پیچشـی سـه بعدی مرسـوم از نظر مقادیر و ابعاد برابر اسـت. برای درک میزان کاهش حجم محاسـباتی در این روش، یک تانسـور ورودی با ابعاد ۳×۱۲×۱۲ در نظر بگیرید که برای آن یک لایه ضرب پیچشی سه بعدی با تعداد ۱۰۰ کرنل با ابعاد ۳×۵×۵ تعریف شده است. هر یک از این کرنل ها در راستای طول و عرض هشت گام حرکتی دارد (۸×۸)، بنابراین تعداد کل ضرب های اسکالر در اسـکالر در این عملیات برابر ۴۸۰۰۰۰=۳×۵×۵×۸×۸×۱۰۰ اسـت، حال این اگر این عملیات را به صورت دو عملیات پیچشی متوالی مبتنی بر نقطه و عمق پیاده سازی کنیم در مرحله ضرب مبتی بر نقطه با توجه به اینکه سـه کرنل دو بعدی ۵×۵ تعریف می شـود تعداد ضـرب ها برابر ۴۸۰۰=۵×۵×۸×۸×۳ اسـت. طول و عرض تانسـور خروجی ضـرب پیچشـی مبتنی بر نقطه مطابق روابط ۴-۱۲ و ۵-۱۲ با فرض گام حرکتی یک و بعد گذاری صفر برابر هشت خواهد بود، بنابراین در مرحله ضـرب پیچشـی مبتنی بر عمق تعداد گام های حرکتی هر کرنل بر روی تانسـور ورودی در راسـتای طول و عرض برابر ۸، ۸×۸، بوده و همچنین تعداد کرنل ها ۱۰۰ اسـت، با توجه به ابعاد ۳×۱×۱ کرنل در این مرحله تعداد ضرب های اجرا شده برابر ۱۹۲۰۰=۳×۱×۱×۸×۸×۱۰۰ است. در مجموع تعداد ضرب های پیچشی مبتنی بر عمق و نقطه برابر۲۴۰۰۰=۱۹۲۰۰+۴۸۰۰ است که در مقایسه با تعداد ضرب های ضرب پیچشی مرسوم سه بعدی، ۴۸۰۰۰۰، کاهش قابل توجهی دارد.



شکل ۱۹-۱۲: ضرب پیچشی مبتنی بر عمق

در ضـرب پیچشـی جدائی پذیر مبتنی بر عمق با توجه با اینکه در مقادیر طول و عرض تانسـور نهائی، $l_1 \times l_2$، فقط ضـرب پیچشـی مبتنی بر نقطه دخیل است لذا مقادیر گام حرکتی بزرگ تر از یک در راستای طول و عرض فقط در این مرحله اعمال شده و گام حرکتی در راستای طول و عرض

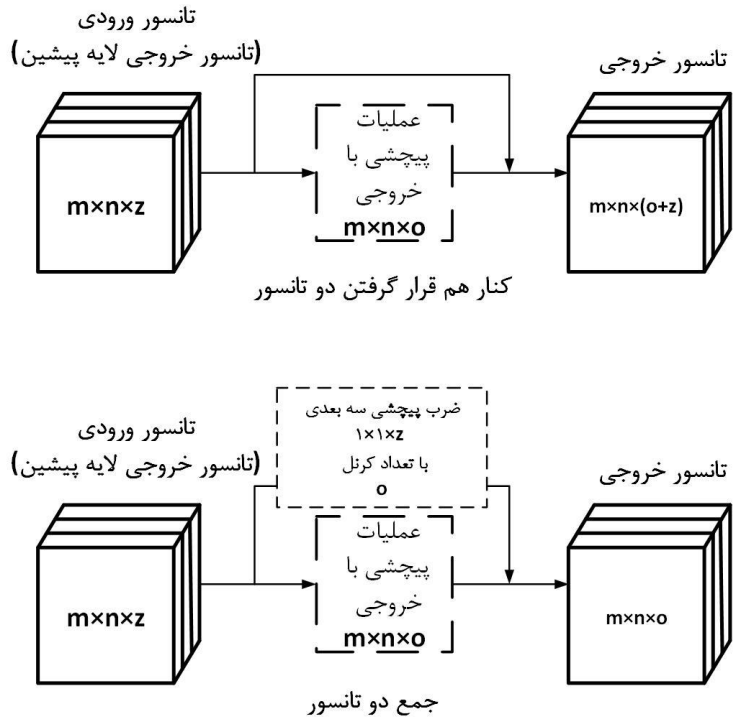
ضرب پیچشی مبتنی بر عمق همواره یک است. همچنین ضرب پیچشی مبتنی عمق، مرحله دوم عملیات ضرب پیچشی جدائی پذیر مبتنی بر عمق، در برخی ساختارهای پیچشی به تنهایی به منظور کنترل عمق یک تانسور استفاده می شود.

۱۲.۱۴ اتصالات جهش[1]

یک ساختار پیچشی مانند سایر ساختارهای عمیق متشکل از تعدادی لایه است که در هر کدام از این لایه ها عملیاتی مانند ضرب پیچشی، تجیمع، تابع فعال ساز و... تعریف می شود. خروجی لایه های اولیه در یک ساختار پیچشی در برگیرنده ویژگی های سطح پایین[2] تصویر ورودی مانند شکل لبه های تصویر، محتوای رنگی و... است، این ویژگی ها مفهوم کلی از یک تصویر ارائه نمی کنند ولی دارای جزئیات بیشتری از تصویر هستند. از طرفی خروجی لایه های انتهائی ساختار دارای ویژگی های سطح بالای[3] تصویر است که در شامل جزئیات تصویر ورودی نیست اما اطلاعات از کلیات تصویر را ارائه می کند. در برخی ساختارها وقتی ویژگی های سطح بالا و پایین در کنار هم به عنوان ورودی یک لایه در ساختار پیچشی تعریف می شوند با توجه به گستره ویژگی ها شامل ویژگی های سطح بالا و پایین عملکرد ساختار بهبود می یابد. بدین منظور اتصالات جهش تعریف می شوند، در این صورت تانسور خروجی لایه های قبل تر در کنار تانسور خروجی لایه فعلی قرار گرفته و به عنوان ورودی لایه بعدی در نظر گرفته می شود.

عملیات تعریف شده برای اتصالات جهش شامل دو عملیات جمع و کنار هم قرار دادن[4] دو تانسور است. در صورتی که اتصال جهش به صورت جمع تعریف شده باشد تانسور خروجی لایه فعلی و لایه قبل باید دارای ابعاد یکسانی باشند که با اعمال اتصال جهش درایه های نظیر به نظیر آن ها با هم جمع شده و حاصل به صورت یک تانسور با همان ابعاد می شود. همچنین در صورتی که این عملیات به صورت کنار هم قرار دادن دو تانسور باشد، تانسور لایه فعلی و قبلی باید دارای طول و عرض یکسانی باشند تا در راستای عمق کنار هم قرار بگیرند. شکل ۲۱-۱۲ دو عملیات را نشان می دهد.

[1] Skip connections
[2] Low level features
[3] High level features
[4] Concatenation



شکل ۲۰-۱۲: اتصالات جهش

همان طور که در شکل ۲۰-۱۲ نیز نشان داده شده است خروجی عملیات در راستای طول و عرض با ورودی برابر است، برای تحقق این مورد بعدگذاری مطابق روابط ۴-۱۲ و ۵-۱۲ به گونه ای تنظیم می شود تا ابعاد تانسور اصلی، تانسور بدون اعمال بعدگذاری، در طول عملیات ثابت بماند، بدین ترتیب تاثیر کاهش بعد در عملیات هایی مانند ضرب پیچشی با گام حرکتی بزرگ تر از یک و یا تجمیع فقط به تانسوری که به آن بعدگذاری اعمال شده است محدود شده و ابعاد اصلی همچنان ثابت باقی می ماند. در حالت جمع نیز برای تنظیم عمق تانسور ورودی با عمق تانسور حاصل از عملیات، اتصال جهش به صورت یک لایه ضرب پیچشی سه بعدی ۱×۱ تعریف می شود.

اتصالات جهش علاوه بر گسترش بازه ویژگی های ورودی به لایه های پیش رو، مقادیر گرادیان های محاسبه شده برای لایه پیش از خود را نیز افزایش داده و بدین ترتیب مانع پدیده محو شدگی گرادیان ها می شوند. رابطه ۸۶-۱۲ حالت کلی جمله مشتق زنجیره ای را نشان می دهد.

$$\cdots \frac{\partial O}{\partial X} \frac{\partial X}{\cdots} \frac{\partial O}{\partial f_{(X)}} \frac{\partial f_{(X)}}{\partial X} + 1 \qquad (۱۲-۸۶)$$

در رابطه ۸۶-۱۲ جمله یک مربوط به اضافه شدن اتصال جهش است، در صورتی که اتصال جهش به صورت ضرب پیچشی سه بعدی ۱×۱ تعریف شود این جمله برابر با مشتق خروجی نسبت به ورودی مربوط به این لایه خواهد بود. این جمله یک در مجموع باعث افزایش مقدار کلی مشتق زنجیره ای و جلوگیری از پدیده محو شدگی گرادیان ها می شود.

۱۲.۱۵ ساختارهای پیچشی و کاربرد آن ها

در این بخش به معرفی چند مورد از ساختارهای مختلف پیچشی که هر یک برای کاربردی خاص تعریف شده پرداخته و ویژگی های هر کدام را بررسی می کنیم. با توجه به طیف گسترده ساختارهای پیچشی ارائه شده صرفا به بررسی ساختارهایی خواهیم پرداخت که به عنوان ساختار مبنای توسعه سایر ساختارها شناخته شده و بسیاری از ساختارهای دیگر براساس آن ها تعریف می شوند.

۱۲.۱۵.۱ ساختارهای پیچشی با لایه های تماما متصل[1]

شکل کلی ساختارهای پیچشی با لایه های تماما متصل که به عنوان مرسوم ترین نوع ساختارهای پیچشی شناخته می شوند متشکل از چند لایه پیچشی و تجمیع، دو بعدی یا سه بعدی، است که به صورت متوالی کنار یکدیگر قرار گرفته اند و به یک لایه مسطح ساز یا تجیمع سراسری ختم می شوند، خروجی این لایه مسطح ساز یا تجیمع سراسری که به صورت یک بردار است وارد چند لایه تماما متصل، لایه پرسپترون، شده و در نهایت خروجی ساختار، مشابه خروجی ساختار شبکه های عصبی پرسپترون چند تعریف می شود. همچنین ورودی این ساختارها به صورت تصاویر سیاه سفید، ماتریس، و یا رنگی، تانسور سه بعدی، است. از این ساختارها با توجه به امکان تعریف مقادیر مطلوبی مشابه مقادیر مطلوب شبکه های پرسپترون به ازای ورودی های متناظر تصویری می توان برای کاربردهایی مانند طبقه بندی[2] تصاویر استفاده کرد.

در هر گام آموزشی روند پیشرو در این ساختارها بدین صورت است که ابتدا یک تصویر از یک مجموعه دادگان آموزشی شامل تصاویر و مقادیر مطلوب متناظر با هر تصویر، برای مثال کلاس هر تصویر، وارد ساختار شده و با عبور از لایه های پیچشی و تجمیع به یک تانسور سه بعدی تبدیل می شود. این تانسور سه بعدی پس از عبور از یک لایه مسطح ساز یا تجمیع سراسری به صورت یک بردار شده و به عنوان ورودی، وارد لایه های تماما متصل انتهای ساختار می شود. در نهایت خروجی لایه های تماما متصل که همان خروجی نهائی ساختار است به صورت یک بردار با ابعادی برابر با ابعاد بردار مقادیر مطلوب مجموعه دادگان آموزشی خواهد بود. در ادامه برای آموزش ساختار بردار خروجی نهائی با بردار مقدار مطلوب متناظر با هر ورودی در تابع هزینه تعریف شده برای ساختار

[1] Fully connected layers
[2] Classification

مقایسه شده و با توجه به مقدار حاصل از تابع هزینه، گرادیان های آموزشی برای همه پارامترهای ساختار شامل وزن های لایه های تماما متصل، وزن های لایه های پیچشی، بایاس ها و... محاسبه شده و ترتیب پارامترهای هر لایه به روز رسانی می شوند.

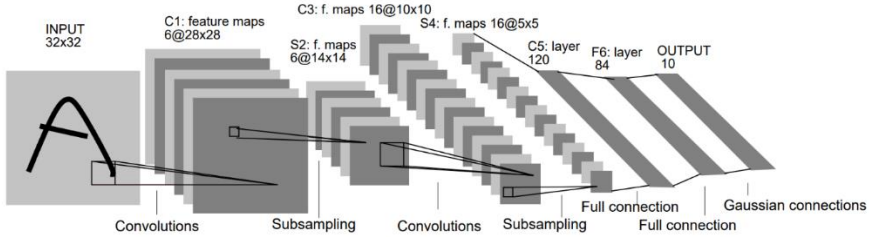
همان طور که اشاره کردیم یک لایه پیچشی همانند یک لایه پرسپترون با اتصالات محلی[1] و وزن های مشترک[2] بین همه نورون ها است، از طرفی لایه های استفاده شده در انتهای ساختار نیز، همان لایه های پرسپترون با ساختاری مشابه با ساختار شبکه های پرسپترون چند لایه است که در آن هر یک از نورون ها وزن مخصوص به خود را داشته و همه اتصالات تعریف شده است، از این رو در ساختار های پیچشی با توجه به ماهیت یکسان لایه های پیچشی و لایه های انتهائی ساختار، لایه پرسپترون، به لایه های انتهای ساختار لایه های تماما متصل اطلاق شده و در نامگذاری آن ها از عنوان لایه های پرسپترون استفاده نمی کنند. در ادامه به معرفی تعدادی از این ساختارها می پردازیم.

۱۲.۱۵.۱.۱ شبکه LeNet-5

این ساختار یکی از اولین و ساده ترین ساختارهای پیچشی معرفی شده برای طبقه بندی تصاویر است. این ساختار در سال ۱۹۹۸ میلادی معرفی شده است. ساختار شبکه LeNet-5 مشابه شکل ۲۱-۱۲ از دو لایه پیچشی با گام حرکتی یک، C، به ترتیب با ۶ و ۱۶ کرنل ۵×۵ و ۱۰×۱۰ و دو لایه تجیمع دوبعدی، S، بدون هم پوشانی به ترتیب با لندازه های ۲×۲ و ۵×۵ است، در مجموع ۵ لایه، که در نهایت وارد دو لایه تماما متصل با تابع فعال ساز تانژانت هیپربولیک شده و خروجی این دو لایه در انتهای ساختار وارد لایه فعال ساز با تابع فعال ساز شعاعی[3] شده و خروجی نهائی ساختار را می سازد. ورودی این ساختار به صورت ماتریس، تصاویر سیاه و سفید، با ابعاد ۳۲×۳۲ تعریف می شود. با توجه به این که این ساختار یکی از اولین ساختارهای معرفی شده است نحوه پیاده سازی آن با تعاریف فعلی ساختارهای پیچشی تفاوت هایی دارد، برای مثال در لایه های پیچشی از تابع فعال ساز استفاده نشده و همچنین برای تبدیل تانسور خروجی لایه های پیچشی به بردار از لایه مسطح ساز یا تجمیع سراسری استفاده نشده است در عوض ابعاد کرنل های لایه های پیچشی و تجمیع به گونه ای تنظیم شده است که تانسور خروجی لایه های پیچشی در نهایت دارای ابعاد ۱۲۰×۱×۱، یک بردار با اندازه ۱۲۰، باشد. همچنین خروجی هر گام حرکتی در لایه های تجمیع این ساختار به صورت مجموع درایه های هر گام حرکتی تقسیم بر یک پارامتر آموزشی تعریف می شود که تا حدودی مشابه تعریف لایه تجمیع میانگین است، این پارامتر آموزشی لایه تجمیع برای همه گام های حرکتی یکسان است.

---

[1] Locally connected
[2] Shared weights
[3] Radial based function (RBF)



شکل ۲۱-۱۲: ساختار پیچشی LeNet-5 [۱۶۴]

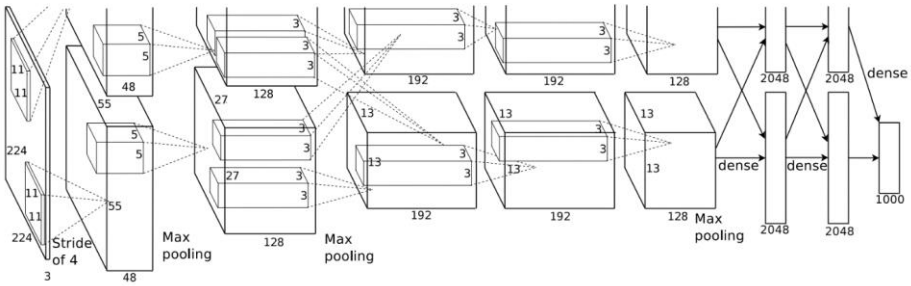
در تعریف لایه پیچشی دوبعدی اشاره کردیم که هر یک از کرنل های لایه در همه ابعاد ورودی ضرب پیچشی می شود که در این ساختار صرفا در لایه پیچشی اول چنین است و در بقیه لایه های پیچشی هر کرنل لایه فقط در تعدادی از ابعاد تانسور ورودی ضرب شده است تا علاوه بر کاهش حجم محاسبات، هر کرنل پیچشی ویژگی های متفاوتی بسازد. این ساختار با توجه به محدودیت ها و عدم انعطاف پذیری مناسب در تعریف ابعاد ورودی و سایر قسمت ها در مقایسه با ساختارهایی که بعدتر از آن ارائه شده اند عملکرد مطلوبی نداشته و معمولا از آن استفاده نمی شود و بیشتر به عنوان یک ساختار اولیه که منشا توسعه سایر ساختارها شده است شناخته می شود.

۱۲.۱۵.۱.۲ شبکه AlexNet

این ساختار که در سال ۲۰۱۲ میلادی معرفی شده است در مقایسه با ساختار LeNet پیشرفت های چشمگیری دارد. ساختار این شبکه مشابه شکل ۲۲-۱۲ است. در این شبکه از لایه های پیچشی سه بعدی با تابع فعال ساز ReLU و لایه های تجمیع حداکثری سه بعدی استفاده شده است. پس از اعمال توابع فعال ساز ReLU در این ساختار از نرمال سازی پاسخ محلی بین کانال استفاده شده است. همان طور که در شکل ۲۲-۱۲ نیز مشخص شده است با توجه به محدودیت های سخت افزاری در زمان توسعه این شبکه پیاده سازی کل ساختار در یک واحد پردازش گرافیکی[1] با توجه به حجم محاسبات آن، لایه های ساختار به دو بخش تقسیم مساوی شده و هر کدام از آن ها که در شکل زیر هم نمایش داده شده اند در یک واحد پردازش گرافیکی مجزا پیاده سازی شده اند که در نهایت در آخرین لایه تماما متصل بردار خروجی هر دو واحد با هم ادغام شده و خروجی نهائی ساختار را می سازند. با وجود محدودیت های مختلف در این ساختار امکان تعریف تصاویر رنگی، تانسور سه بعدی، با ابعاد ۲۲۴×۲۲۴ به عنوان ورودی فراهم شده و عملکرد آن در طبقه بندی تصاویر ورودی نسبت به ساختارهای پیش از خود بهبود چشمگیری داشته است. در این ساختار برای اولین بار تاثیر هم پوشانی در لایه های تجمیع بررسی شد، طبق نتایج تجربی استفاده از گام حرکتی کوچک تر از اندازه لایه تجمیع و اعمال هم پوشانی باعث بهبود تنک زائی شده و در نهایت باعث ارتقای عملکرد کلی ساختار می شود. برای جلوگیری از بیش برازش در لایه

[1] Graphical process unit (GPU)

های تماما متصل ساختار در مرحله آموزش از روش حذف تصادفی[1] استفاده شده است.



شکل ۲۲-۱۲: ساختار پیچشی AlexNet [۱۶۵]

۱۲.۱۵.۱.۳ شبکه VGG

طبق نتایج تجربی استفاده از تعداد لایه های پیچشی بیشتر به صورت متوالی در یک ساختار پیچشی با تعداد کرنل کمتر نسبت به ساختاری که دارای تعداد لایه کمتر ولی تعداد کرنل بیشتری باشد با فرض اینکه تعداد کرنل ها، اندازه گام حرکتی، ابعاد کرنل ها و... در هر دو حالت به گونه ای باشند که حجم محاسبات کل در دو ساختار برابر باشند، عملکرد بهتری دارد حتی در مواردی ساختاری با تعداد لایه بیشتر ولی تعداد کرنل، پارامتر، کمتر که حجم محاسباتی کمتری نسبت به ساختاری که تعداد لایه کمتر و تعداد کرنل بیشتری دارد نیز عملکرد بهتری دارد. بنابراین نتایج ساختارهای پیچشی متفاوتی با تعداد لایه های بیشتر و ساختار عمیق تر معرفی شدند. استفاده از تعداد لایه های بیشتر باعث بروز مشکل محو شدگی گردایان ها به ویژه در لایه های ابتدایی می شود که برای رفع آن در هر ساختار راهکارهای متنوعی ارائه شده است. شبکه VGG یکی از این ساختارهای عمیق پیچشی است که در سال ۲۰۱۴ میلادی معرفی شده است. ساختار این شبکه به صورت ۱۱، ۱۳، ۱۶ و ۱۹ لایه دارای پارامتر آموزشی، لایه پیچشی سه بعدی و تماما متصل، تعریف می شود که حالت ۱۶ و ۱۹ لایه بهترین عملکرد را داشته و به ترتیب با عنوان VGG-16 و VGG-19 شناخته می شوند. در جدول ۱-۱۲ ساختار های مختلف این شبکه نشان داده شده اند. در این ساختار از لایه های پیچشی با اندازه کرنل ۱×۱ استفاده شده است که ابعاد تانسور ورودی را حفظ کرده و هدف از آن اعمال نگاشت غیرخطی تابع فعال ساز است. ورودی این ساختار همانند ساختار AlexNet به صورت تصویر رنگی با ابعاد ۲۲۴×۲۲۴ تعریف می شود. همان طور که در جدول ۱-۱۲ نیز مشخص شده است این ساختار به بلوک های مجزا تقسیم شده که هر بلوک دارای تعدادی عملیات پیچشی است. برخی از این بلوک ها عینا در ساختار تکرار شده اند.

[1] Drop-out

جدول ۱-۱۲: ساختار پیچشی VGG [۱۶۶]

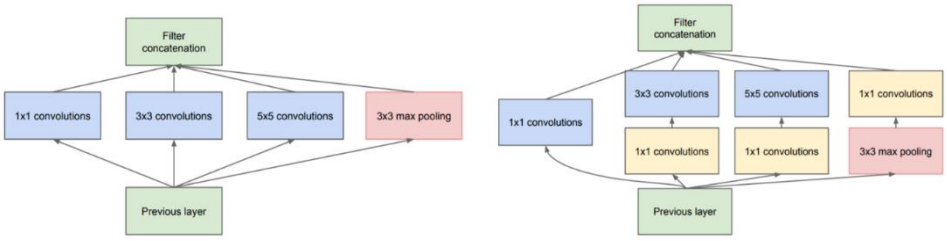
ConvNet Configuration					
A	A-LRN	B	C	D	E
11 weight layers	11 weight layers	13 weight layers	16 weight layers	16 weight layers	19 weight layers
input (224 × 224 RGB image)					
conv3-64	conv3-64 **LRN**	conv3-64 **conv3-64**	conv3-64 conv3-64	conv3-64 conv3-64	conv3-64 conv3-64
maxpool					
conv3-128	conv3-128	conv3-128 **conv3-128**	conv3-128 conv3-128	conv3-128 conv3-128	conv3-128 conv3-128
maxpool					
conv3-256 conv3-256	conv3-256 conv3-256	conv3-256 conv3-256	conv3-256 conv3-256 **conv1-256**	conv3-256 conv3-256 **conv3-256**	conv3-256 conv3-256 conv3-256 **conv3-256**
maxpool					
conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512 **conv1-512**	conv3-512 conv3-512 **conv3-512**	conv3-512 conv3-512 conv3-512 **conv3-512**
maxpool					
conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512 **conv1-512**	conv3-512 conv3-512 **conv3-512**	conv3-512 conv3-512 conv3-512 **conv3-512**
maxpool					
FC-4096					
FC-4096					
FC-1000					
soft-max					

۱۲.۱۵.۱.۴ شبکه InceptionNet

شبکه InceptionNet که در سال ۲۰۱۴ میلادی معرفی شده است. در توسعه این شبکه تاثیر افزایش عرض شبکه، اضافه کردن کرنل به هر لایه، مورد بررسی قرار گرفته است. همان طور که در معرفی ساختار VGG اشاره کردیم افزودن کرنل به یک لایه پیچشی نسبت به افزودن یک لایه به ساختار تاثیر کمتری در بهبود عملکرد ساختار دارد اما این موضوع از منظر دیگری در شبکه InceptionNet بررسی شده است بدین صورت که به جای افزودن کرنل های مشابه به یک لایه پیچشی برای این ساختار بلوک های تکرار شونده مطابق شکل ۱۲-۲۳ تعریف شده است که از کرنل های متفاوت با ابعاد مختلف تشکیل شده است. تانسور خروجی هریک از انشعابات بلوک شکل در انتهای بلوک در راستای بعد سوم، عمق، کنار یکدیگر قرار گرفته[1] و تانسور خروجی نهائی بلوک

[1] Concatenate

را تشـکیل می دهد. بدیهی اسـت که در مرحله تشـکیل خروجی نهائی که تانسـورهای انشـعابات مختلف در راسـتای عمق کنار هم قرار می گیرند، باید اندازه آن ها در دیگر ابعاد، طول و عرض، یکسـان باشـند. برای تحقق این موضـوع با توجه به رابطه ۴-۱۲ و ۵-۱۲ باید بعدگذاری به گونه ای اعمال شـود تا اندازه خروجی همه انشـعابات با هم برابر باشـند. بدین منظور در این سـاختار اندازه بعدگذاری را در هر بعد همواره به اندازه یک واحد کمتر از اندازه در آن بعد کرنل پیچشـی انتخاب می کنند که همان روابط ۱۶-۱۲ و ۱۷-۱۲ هستند. در واقع روابط ۱۶-۱۲ و ۱۷-۱۲ برای اولین بار در این سـاختار تعریف شـده اند که علاوه بر یکسـان سـازی طول و عرض ابعاد خروجی هر انشـعاب، ابعاد تانسـور ورودی به عملیات ضـرب پیچشـی را نیز حفظ می کنند. هر یک از انشـعابات در بلوک های این ساختار یک لایه پیچشی سه بعدی محسوب شده و دارای تعدادی کرنل پیچشی سه بعدی است که در شکل ۲۳-۱۲ فقط اندازه طول و عرض آن ها مشخص شده است، اندازه بعد سوم هر یک از کرنل های سـه بعدی برابر اندازه بعد سـوم تانسـور ورودی بوده و تعداد آن ها در هر انشـعاب بنابر کاربرد می تواند مقادیر مختلفی داشـته و از این نظر سـاختار انعطاف پذیری مناسـبی دارد. در این سـاختار هدف از تعریف ضـرب های پیچشـی با ابعاد کرنل ۱×۱ کاهش بعد تانسـور در جهت عمق و افزودن نگاشت غیر خطی به ساختار است؛ بدین صورت که اگر فرض کنیم تانسور ورودی به ضـرب پیچشـی با کرنل ۱×۱ دارای ابعادی $m \times n \times k$ باشـد در این صـورت هر یک از کرنل های لایه پیچشـی سـه بعدی در این لایه دارای ابعادی برابر با $1 \times 1 \times k$ خواهد بود که اعمال آن به تانسور ورودی، تانسوری، ماتریسی، با ابعاد $m \times n \times 1$ تشکیل می دهد، در صورتی که تعداد کرنل های سه بعدی در لایه g، $g < k$، باشد در این صورت خروجی نهائی لایه پیچشی سه بعدی دارای ابعادی برابر با $m \times n \times g$ خواهد بود که طول و عرض آن برابر با ورودی بوده ولی عمق آن کاهش پیدا کرده است. به این لایه های پیچشی با ابعاد کرنل $1 \times 1 \times k$ لایه تنگنا[1] گفته می شود.



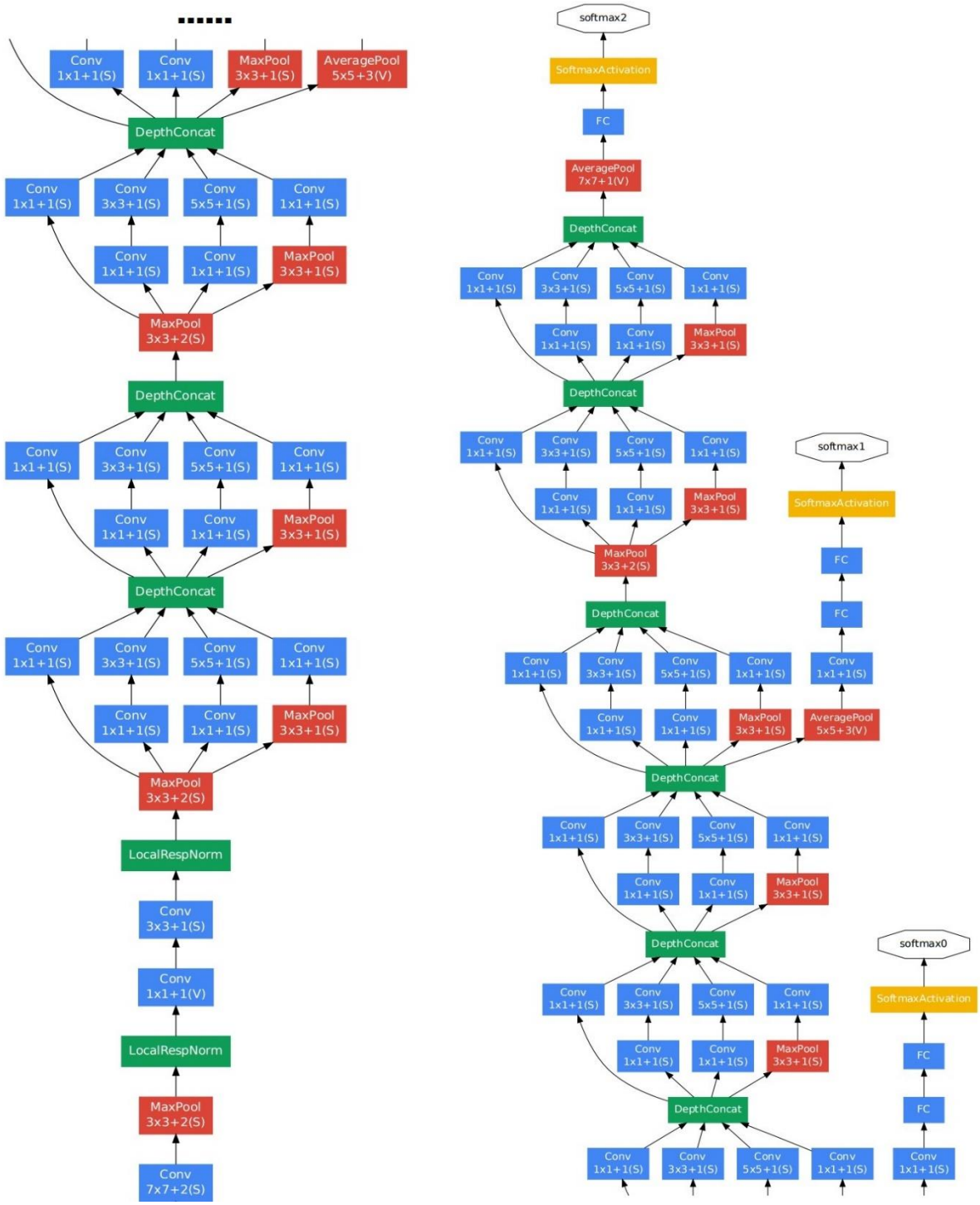
الف) بلوک پایه **ب) بلوک با لایه های تنگنا**

شکل ۲۳-۱۲:ساختار بلوک Incpetion [۱۰۶]

بلوک های این سـاختار از سـیسـتم بینائی که در آن اطلاعات در مقیاس های متفاوت به مغز ارسال می شود الهام گرفته شده و دلیل نام گذاری آن نیز همین موضوع است. این شبکه با عنوان

[1] Bottleneck layer

GoogleNet نیز شناخته می شود. این شبکه در کل از ۷ بلوک که به صورت متوالی در کنارهم قرار گرفته اند تشکیل شده است، در بین اتصال بین بلوک ها از لایه تجمیع حداکثری استفاده شده و در مجموع این ساختار دارای ۲۲ لایه با پارامتر آموزشی و در کل ۲۷ لایه دارد. با توجه به افزایش تعداد کرنل های پیچشی در مجموع این ساختار با وجود عملکرد مناسب در طبقه بندی تصاویر، حجم محاسباتی بسیار بالایی دارد. در ادامه توسعه این ساختار در سال ۲۰۱۵ میلادی در نسخه های دوم و سوم آن که به ترتیب با عنوان InceptionNet V.2 و InceptionNet V.3 شناخته می شوند برای اولین بار در ساختارهای پیچشی از نرمال سازی دسته ای استفاده شد و عملکرد ساختار به طور قابل توجهی بهبود یافت. افزایش تعداد کرنل ها در انشعابات بلوک ها، استفاده از ضرب جدائی پذیر فضایی برای کاهش حجم محاسبات و بهینه سازی ساختار با تبدیل کرنل های ۵×۵ و ۷×۷ به ترتیب به دو و سه کرنل ۳×۳ که به طور سری کنار هم قرار گرفته اند از دیگر ارتقاهای صورت گرفته در نسخه های جدیدتر این ساختار است. در این ساختار لایه SoftMax که بردار خروجی نهائی را تولید می کند علاوه بر انتهای ساختار در لایه های میانی نیز تعریف شده است که هدف آن افزایش قدرت تفکیک پذیری ساختار با ویژگی های سطح پایین تر است. خروجی این لایه های کمکی با مقدار مطلوب مقایسه شده و در نهایت با یک ضریبی بین صفر تا یک با مقدار تابع هزینه نهائی جمع شده و به ازای هر کدام از آن ها یک مسیر آموزشی برای لایه های قبل از خود تعریف می شود. این لایه های SoftMax کمکی فقط در مرحله آموزش تعریف شده و در سایر مراحل از ساختار شبکه حذف می شوند. شکل ۲۴-۱۲ ساختار کلی شبکه InceptionNet را نشان می دهد.



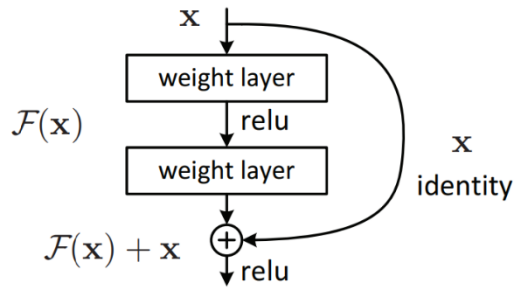
شکل ۲۴-۱۲: ساختار شبکه InceptionNet [۱۰۶]

در سال ۲۰۱۶ میلادی عملیات تعریف شده در انشعابات بلوک مجددا بهینه سازی شده و ساختار جدید تحت عنوان InceptionNet V.4 معرفی شد، علاوه بر این ساختاری حاصل از ادغام آن با ساختار ResNet معرفی شد که با عنوان InceptionResNet شناخته می شود. همچنین ساختار دیگری با عنوان Xception که در سال ۲۰۱۷ میلادی اساس بلوک های Inception معرفی شده است که در آن از ضرب جدائی پذیر فضائی مبتنی بر عمق استفاده شده است.

۱۲.۱۵.۱.۵ شبکه ResNet

با توجه به بروز پدیده محو شدگی گرادیان ها در تعریف ساختارهای پیچشی عمیق امکان تعریف تعداد لایه های بیشتر با روش های و ساختارهای مرسوم وجود ندارد. بیشترین تعداد لایه قابل تعریف با این روش ها حداکثر حدود ۲۰ تا ۲۵ لایه است. اما همان طور که در بررسی ساختار VGG اشاره کردیم بالا بردن تعداد لایه ها باعث افزایش نگاشت غیرخطی و بهبود عملکرد ساختار می شود. شبکه ResNet با استفاده از اتصالات جهش[1] از بروز پدیده محو شدگی گرادیان ها جلوگیری کرده و امکان تعریف لایه های بیشتر در ساختار پیچشی را فراهم کرده است. مانند ساختار Inception این ساختار نیز از بلوک های تکرار شونده مطابق شکل ۲۵-۱۲ تشکیل شده است. در معرفی بلوک های ساختار Inception اشاره کردیم که در مرحله آخر تانسور مربوط به هر انشعاب در کنار یکدیگر قرار گرفته و در نهایت اندازه عمق، بعد سوم، تانسور نهایی برابر مجموع عمق، بعد سوم، تانسورهای هر انشعاب و طول و عرض، ابعاد اول و دوم، همه تانسورها با هم برابر است. در حالی که در ساختار ResNet تانسور خروجی دو انشعاب که دارای ابعاد برابری هستند با هم جمع می شوند، مقادیر درایه ها نظیر به نظیر با هم جمع می شوند، بنابراین روند پیشرو در بلوک های این ساختار مطابق رابطه ۸۷-۱۲ است. این ساختار در سال ۲۰۱۵ میلادی معرفی شده است.

$$O = F_{(X)} + X \tag{۱۲-۸۷}$$



شکل ۲۵-۱۲: بلوک ساختار ResNet [۱۶۷]

[1] Skip connections

در رابطه ۸۷-۱۲ منظور از $F_{(X)}$ مجموعه از عملیات های پیچشی مختلف سه بعدی شامل مطابق شکل ۲۵-۱۲ است که به صورت متوالی بر روی تانسور ورودی اعمال می شوند هر یک از این عملیات ها به صورت یک لایه تعریف می شوند. اعمال هر یک از این عملیات ها به تانسور ورودی ممکن است به توجه به روابط ۴-۱۲ و ۵-۱۲ ابعاد آن را تغییر دهد در صورتی که در طبق رابطه ۸۷-۱۲ در روند پیشرو بلوک های ساختار باید ابعاد دو تانسور ورودی و خروجی عملیات $F_{(X)}$ با هم برابر باشند تا بتوان آن ها را با هم جمع کرد. با توجه به تعداد کرنل های سه بعدی در هر مرحله از عملیات ممکن است عمق تانسور ورودی و خروجی عملیات نیز برابر نباشند. برای رفع این مشکل از دو روش در این ساختار استفاده شده است، مشابه روش های بررسی شده در بخش ۱۲-۱۴:

۱. **استفاده از بعدگذاری با اندازه بالاتر برای تانسور ورودی در هنگام اعمال عملیات :** در این روش لندازه بعدگذاری را قبل از اعمال عملیات به تانسور ورودی به گونه ای طبق روابط ۴-۱۲ و ۵-۱۲ تنظیم می کنند که ابعاد خروجی، طول و عرض، عملیات با ابعاد تانسور ورودی اصلی مطابق رابطه با هم برابر باشند.

۲. **استفاده از کرنل های پیچشی سه بعدی با طول و عرض یک در مواردی که اندازه عمق تانسور خروجی عملیات کوچکتر از عمق تانسور ورودی است:** این روش در مواردی که عمق تانسور ورودی و خروجی عملیات متفاوت است، مشابه کرنل های ۱×۱ لایه تنگنا در ساختار Inception که عمق یک تانسور ورودی را با حفظ اندازه طول و عرض تغییر می داد استفاده می شود. همچنین برای ثابت ماندن طول و عرض تانسور ورودی و خروجی درکنار این روش از روش اول نیز در استفاده شده و به تانسور ورودی قبل از عملیات بعدگذاری مطابق روابط ۴-۱۲ و ۵-۱۲ اعمال می شود.

با توجه به تغییر طول و عرض تانسور در عملیات هایی مانند ضرب پیچشی از روش اول همواره در تعریف بلوک های این ساختار استفاده می شود اما استفاده از روش دوم فقط به مواردی که علاوه بر طول و عرض با توجه به تعداد کرنل های تعریف شده در هر عملیات $F_{(X)}$ عمق تانسور نیز تغییر می کند محدود می شود، البته می توان روش دوم را در مواردی که عمق تانسور تغییر نمی کند نیز استفاده که کرد که در این صورت با توجه به اضافه شدن یک عملیات ضرب پیچشی در اتصال جهش حجم محاسبات افزایش یافته احتمال بروز پدیده محو شدگی گرادیان ها افزایش می یابد، اما طبق نتایج تجربی استفاده از هر دو روش در همه بلوک ها باعث بهبود نسبی عملکرد می شود.

مشتق ورودی به خروجی هر بلوک که برای آموزش لایه های پیش از آن استفاده می شود

مطابق رابطه ۸۸-۱۲، مشابه رابطه ۸۶-۱۲، تعریف می شود، مطابق این رابطه اضافه شدن جمله ثابت یک به رابطه مشتقات زنجیره ای باعث بزرگ تر شدن کلی مقدار این مشتق های زنجیره ای و جلوگیری از پدیده محو شدگی گرادیان می شود.

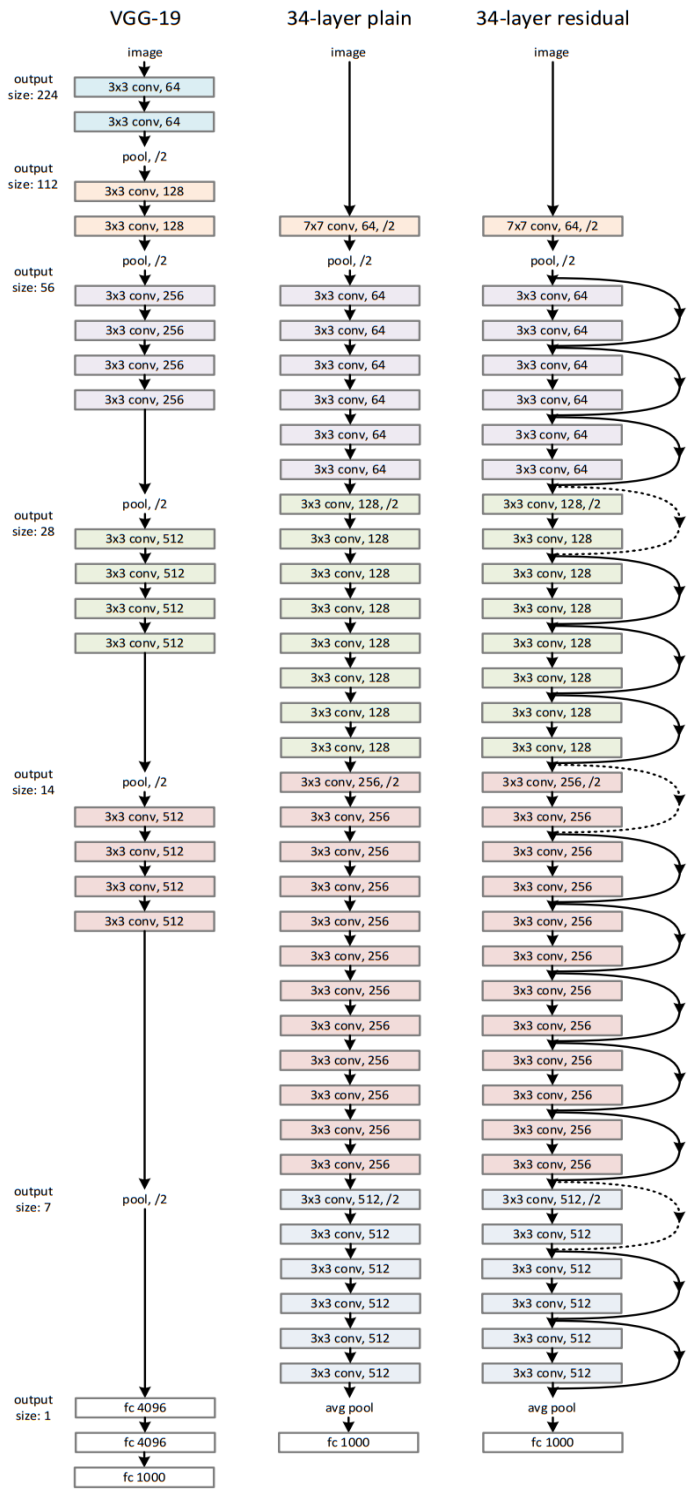
$$\frac{\partial E}{\dots}\dots\frac{\partial O}{\partial X}\frac{\partial X}{\dots}\qquad \frac{\partial O}{\partial F_{(X)}}\frac{\partial F_{(X)}}{\partial X}+1 \tag{۱۲-۸۸}$$

رابطه ۸۹-۱۲ مربوط به حالتی است که تنها از روش اول ذکر شده در تعریف بلوک های ساختار استفاده شده باشد، در صورت استفاده از هر دو روش فوق الذکر رابطه ۸۹-۱۲ به صورت رابطه ۹۰-۱۲ خواهد بود.

$$\frac{\partial E}{\dots}\dots\frac{\partial O}{\partial X}\frac{\partial X}{\dots}\qquad \frac{\partial O}{\partial F_{(X)}}\frac{\partial F_{(X)}}{\partial X}+\frac{\partial O}{\partial Conv_{\cdot(X)}}\frac{\partial Conv_{\cdot(X)}}{\partial X} \tag{۱۲-۸۹}$$

در رابطه ۹۰-۱۲ منظور از .Conv همان لایه پیچشی سه بعدی ۱×۱ است. با توجه به تغییرات رابطه ۹۰-۱۲ نسبت به رابطه ۸۹-۱۲ ممکن است مقادیر مشتق زنجیره ای نسبت به رابطه ۸۹-۱۲ کوچک تر بوده و بدین ترتیب احتمال پدیده محوشدگی گرادیان ها تا حدی افزایش یابد.

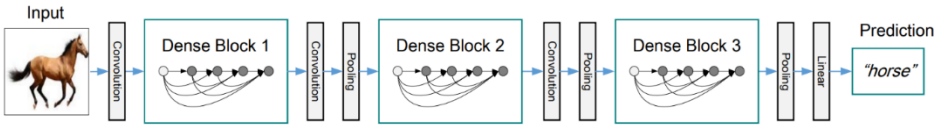
در این ساختار از لایه های تجمیع، به جز در لایه دوم مطابق شکل ۲۶-۱۲، استفاده نشده و کاهش بعد با افزایش اندازه گام حرکتی در لایه پیچشی، دو گام حرکتی، انجام می شود. همچنین با توجه به اینکه ساختار به صورت بلوک های تکرار شونده تعریف می شود برای ثابت نگاه داشتن محاسبات در همه بلوک ها، در بلوک هایی که ابعاد تانسور خروجی نسبت به ورودی، ورودی که به آن بعدگذاری روش اول اعمال شده باشد، کاهش می یابد، با توجه به اندازه گام حرکتی دو، نسبت این کاهش $\frac{1}{2}$ است، عمق تانسور خروجی بلوک دو برابر ورودی در نظر گرفته می شود. این ساختار با توجه حل مشکل مربوط به بروز محوشدگی گرادیان ها با تعداد لایه های مختلف به صورت ساختار های ۱۸، ۳۴، ۵۰، ۱۰۱، ۱۵۲ لایه تعریف شده است. در شکل ۲۶-۱۲ ساختار ۳۴ لایه ResNet، ساختار ۳۴ لایه بدون در نظر گرفتن اتصالات جهش و ساختار VGG-19 با هم مقایسه شده اند. در شکل ۲۶-۱۲ در اتصالات جهشی که به صورت خط ممتد مشخص شده اند تغییر ابعادی در ورودی و خروجی بلوک صورت نمی گیرد ولی در اتصالات جهشی که به صورت خط چین مشخص شده اند ابعاد تانسور ورودی و خروجی با هم برابر نبوده و لذا از یک یا هر دو روش فوق الذکر برای رفع این معضل استفاده می شود.



شکل ۲۶-۱۲: مقایسه ساختارهای پیچشی ResNet و VGG [۱۶۷]

۱۲.۱۵.۱.۶ شبکه DenseNet

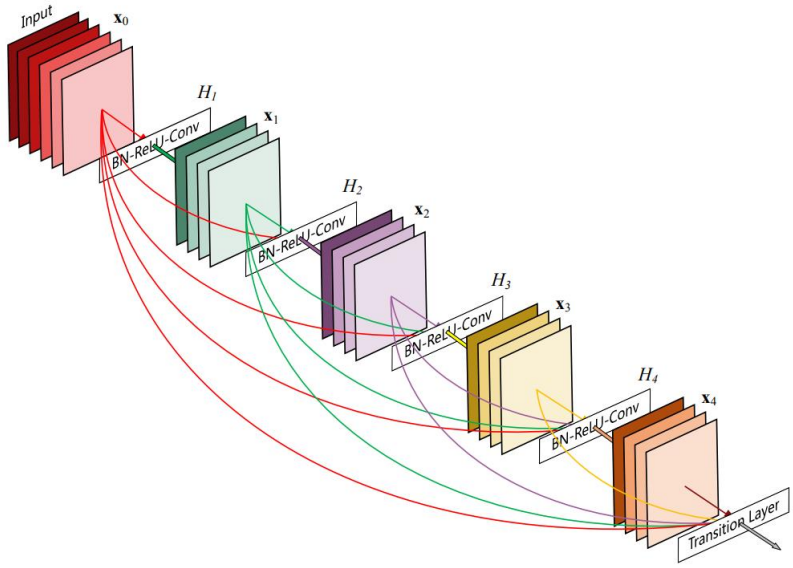
تاثیر استفاده از اتصالات جهش در ساختار ResNet باعث شد ساختارهای دیگری با ایده مشابه تعریف شوند، یکی از آن ها ساختار DenseNet است در این ساختار برخلاف ساختار ResNet که در اتصالات جهش آن دو تانسور با هم جمع می شدند تانسور ها در اتصالات جهش در راستای بعد سوم در کنار یک قرار می گیرند[1]. این شبکه در که در سال ۲۰۱۷ میلادی معرفی شده است. این ساختار نیز از بلوک های تکرار شونده تشکیل شده است. بین هر دو بلوک متصل یک لایه پیچشی سه بعدی با کرنل هایی با طول و عرض یک برای کاهش عمق تانسور و یک لایه تجمیع میانگین برای کاهش طول و عرض تانسور مطابق شکل ۱۲-۲۷ تعریف شده است.



شکل ۱۲-۲۷: نحوه اتصال بلوک ها در ساختار DenseNet [۱۶۸]

در هر بلوک این ساختار چهار عملیات مطابق شکل ۱۲-۲۸ تعریف می شود که در هر یک از این عملیات ها سه لایه به ترتیب شامل یک لایه نرمال سازی دسته ای[2]، یک تابع فعال ساز ReLU و یک لایه پیچشی سه بعدی با کرنل هایی با اندازه طول و عرض سه، 3×3 BN+ReLU+Conv.، است. اتصالات جهش در داخل هر بلوک برای اتصال نگاشت های ویژگی حاصل از هر یک از این چهار عملیات به ورودی عملیات های بعد از خود مطابق شکل ۱۲-۲۸ استفاده می شود. با اعمال اتصالات جهش نگاشت های ویژگی همه عملیات های پیشین در بلوک با اتصال جهش در کنار تانسور ورودی هر یک از عملیات های داخل بلوک قرار می گیرند. در این ساختار بعدگذاری لایه های داخل هر بلوک به گونه انتخاب می شود، مطابق روابط ۱۲-۱۶ و ۱۲-۱۷، تا اندازه تانسورها در ورودی هر لایه که با اتصالات جهش کنار یکدیگر قرار می گیرند در راستای دو بعد طول و عرض با هم برابر باشند.

[1] Concatenate
[2] Batch normalization



شکل ۲۸-۱۲: ساختار یک بلوک DenseNet [۱۶۸]

استفاده از نگاشت های ویژگی همه عملیات های قبل در بلوک های این ساختار به جای استفاده از نگاشت ویژگی حاصل از تنها یک لایه قبل تر که در سایر ساختارها مرسوم است، باعث می شود تا ویژگی های سطح پایین تر[1]، نگاشت ویژگی لایه های ابتدایی، در کنار ویژگی های سطح بالا[2]، نگاشت ویژگی لایه های متأخر، به عنوان ورودی هر عملیات در نظر گرفته شده و با توجه به این گستره وسیع تانسور های ورودی شامل ویژگی های سطح بالا و پایین در لایه های مختلف، عملکرد کلی ساختار ارتقا یابد. از این رو در این ساختار با استفاده از تعداد کرنل های کمتری نسبت به سایر ساختارها در لایه های پیچشی می توان به عملکرد مطلوبی دست یافته و حجم محاسبات ساختار را کاهش داد. اما در کنار حجم محاسبات بهینه در این ساختار استفاده مکرر از اتصالات جهش در بلوک ها و تعریف تانسور جدید برای ورودی هریک از چهار عملیات در بلوک متشکل از همه نگاشت های ویژگی عملیات های قبلی حجم زیادی از حافظه را اشغال می کند، برای رفع این معضل در لایه های پیچشی ۱×۱ بین هر دو بلوک عمق تانسور نصف می شود تا حجم حافظه مورد نیاز کنترل شود. جدول تعداد ساختار کلی شبکه را برای حالاتی ساختار شامل DenseNet-121، ۱۲۱، DenseNet-169، ۱۶۹، DenseNet-201، ۲۰۱، و DenseNet-264، ۲۶۴، کرنل پیچشی است نشان می دهد. با توجه به تعداد کرنل ها و پارامترهای آموزشی کمتر در این ساختار نسبت به ساختارهای پیچشی دیگر در این ساختار احتمال بیش برازش کمتر است ولی از طرف دیگر این ساختار نسبت به سایر ساختارها نسبتا دیرتر همگرا شده و مقدار تابع هزینه آن به ازای گام های آموزشی بیشتری به مقدار بهینه می رسد.

[1] Low-level features
[2] High-level features

۱-۶-۱-۱۵-۱۲ نرخ رشد[1]

در ساختار پیچشی DenseNet یک پارامتر کنترلی به نام نرخ رشد، k، تعریف می شود که هدف آن تعریف تعداد نگاشت های ویژگی در بخش های مختلف ساختار است. پارامتر نرخ رشد، k، در واقع همان تعداد نگاشت های ویژگی، عمق تانسور خروجی، هر یک از عملیات ها، لایه نرمال سازی دسته ای + تابع ReLU + لایه پیچشی، در بلوک های ساختار است. اشاره کردیم که هر یک از تانسور های خروجی عملیات های بلوک های ساختار با استفاده از اتصالات جهش در کنار تانسور ورودی هر عملیات بعدی مطابق شکل ۲۸-۱۲ قرار می گیرد. عملیات l-ام در یکی از بلوک های ساختار را در نظر بگیرید، اگر فرض کنیم تعداد نگاشت ویژگی ورودی به این بلوک از ساختار k_0 باشد، عمق ورودی بلوک = k_0، و هر یک از عملیات های داخل این بلوک از ساختار دارای تعداد k نگاشت ویژگی خروجی، عمق تانسور خروجی همه عملیات های پیشین = نرخ رشد، باشند در این صورت تعداد نگاشت ویژگی ورودی به عملیات l-ام در این بلوک برابر رابطه ۹۰-۱۲ است.

$$k_0 + k(l-1) \qquad (۹۰-۱۲)$$

با توجه به رابطه ۹۰-۱۲ نقش نرخ رشد در این ساختار تنظیم یک مقدار ثلبت برای تعداد نگاشت های خروجی ویژگی عملیات های بلوک های ساختار است، از طرفی لایه های تعریف شده در هر یک از عملیات ها شامل لایه نرمال سازی دسته ای، لایه فعال ساز و لایه پیچشی سه بعدی است، که با توجه به آن ها برای تنظیم تعداد نگاشت های ویژگی در خروجی به اندازه نرخ رشد باید اندازه عمق کرنل های سه بعدی لایه پیچشی ۳×۳ برابر با عمق تانسور ورودی، رابطه ۹۰-۱۲، و تعداد آن برابر نرخ رشد، k، باشد. همان طور که در بررسی ساختار شبکه InceptionNet نیز اشاره کردیم این کرنل های ضرب پیچشی سه بعدی در صورتی که عمق تانسور ورودی بزرگ باشد مستلزم محاسبات زیادی هستند. برای رفع این معضل در این ساختار با توجه به مشترک بودن مسئله با شبکه InceptionNet از راهکاری مشابه بلوک های شبکه Inception استفاده شده است؛ بدین صورت که به عملیات های تعریف شده در بلوک های ساختار علاوه بر لایه های ذکر شده یک لایه پیچشی سه بعدی با کرنل هایی با طول و عرض یک، لایه پیچشی تنگنا[2]، به تعداد در ابتدا اضافه می کنند تا عمق تانسور ورودی حاصل از رابطه کاهش یافته و سپس در ادامه عملیات حجم محاسبات در لایه پیچشی ۳×۳ کاهش یابد.

بدین ترتیب لایه های تعریف شده در هر یک از عملیات های بلوک های ساختار در نهایت شامل لایه نرمال سازی دسته ای، لایه فعال ساز ReLU، لایه تنگنا، لایه پیچشی ۱×۱، لایه نرمال سازی دسته ای، لایه فعال ساز ReLU، لایه پیچشی ۳×۳، BN+ReLU+Conv.1×1+BN+ReLU+Conv.3×3، است. در این ساختار همواره نرخ رشد به صورت مضربی از چهار تعریف می شود. جدول ۲-۱۲ ساختارهای مختلف این شبکه را به ازای

[1] Growth rate
[2] Bottleneck layer

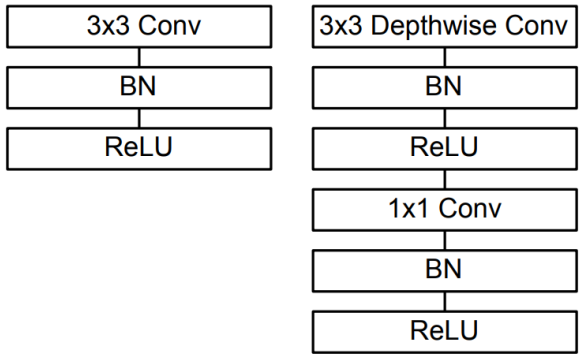
تعداد لایه های مختلف ۱۲۱، ۱۶۹، ۲۰۱ و ۲۶۴ با نرخ رشد ۳۲ نشان می دهد.

جدول ۲-۱۲: ساختار پیچشی DenseNet [۱۶۸]

Layers	Output Size	DenseNet-121	DenseNet-169	DenseNet-201	DenseNet-264
Convolution	112 × 112	7 × 7 conv, stride 2			
Pooling	56 × 56	3 × 3 max pool, stride 2			
Dense Block (1)	56 × 56	[1 × 1 conv; 3 × 3 conv] × 6	[1 × 1 conv; 3 × 3 conv] × 6	[1 × 1 conv; 3 × 3 conv] × 6	[1 × 1 conv; 3 × 3 conv] × 6
Transition Layer (1)	56 × 56	1 × 1 conv			
	28 × 28	2 × 2 average pool, stride 2			
Dense Block (2)	28 × 28	[1 × 1 conv; 3 × 3 conv] × 12	[1 × 1 conv; 3 × 3 conv] × 12	[1 × 1 conv; 3 × 3 conv] × 12	[1 × 1 conv; 3 × 3 conv] × 12
Transition Layer (2)	28 × 28	1 × 1 conv			
	14 × 14	2 × 2 average pool, stride 2			
Dense Block (3)	14 × 14	[1 × 1 conv; 3 × 3 conv] × 24	[1 × 1 conv; 3 × 3 conv] × 32	[1 × 1 conv; 3 × 3 conv] × 48	[1 × 1 conv; 3 × 3 conv] × 64
Transition Layer (3)	14 × 14	1 × 1 conv			
	7 × 7	2 × 2 average pool, stride 2			
Dense Block (4)	7 × 7	[1 × 1 conv; 3 × 3 conv] × 16	[1 × 1 conv; 3 × 3 conv] × 32	[1 × 1 conv; 3 × 3 conv] × 32	[1 × 1 conv; 3 × 3 conv] × 48
Classification Layer	1 × 1	7 × 7 global average pool			
		1000D fully-connected, softmax			

۱۲.۱۵.۱.۷ شبکه MobileNet

این شبکه که در سال ۲۰۱۷ میلادی معرفی شده است. هدف از توسعه این شبکه کاهش حجم محاسبات لازم در یک ساختار پیچشی بوده است. بدین منظور برای اولین بار در این ساختار از ضرب پیچشی جدائی پذیر مبتنی بر عمق در کرنل های سه بعدی با طول و عرض سه مطابق شکل استفاده شده است.



الف) بلوک چدائی پذیر **ب) بلوک مرسوم**

شکل ۲۹-۱۲: مقایسه بلوک پیچشی مرسوم و جدائی پذیر مبتنی بر عمق در ساختار MobileNet [۱۷۰]

شکل ۲۹-۱۲ مقایسه بلوک استفاده شده در ساختارهای پیچشی مرسوم و بلوک شبکه

MobileNet است. با استفاده از ضرب پیچشی جدائی پذیر در این ساختار حجم محاسباتی لازم در مقایسه با یک ساختار مرسوم متشکل از بلوک هایی مطابق شکل ۲۹-۱۲ الف، تعداد بلوک برابر در هر دو ساختار، تا حد زیادی کاهش می یابد، البته این کاهش حجم محاسبات در حد بسیار کمی منجر به افت عملکرد ساختار می شود. جدول ۳-۱۲ مقایسه تعداد پارامترهای ساختار و دقت حاصل از آموزش دو ساختار متشکل از بلوک های مرسوم مطابق شکل ۲۹-۱۲ الف و بلوک های شبکه MobileNet مطابق شکل ۲۹-۱۲ ب است. همان طور از جدول ۳-۱۲ نیز برداشت می شود افت عملکرد این ساختار در برابر کاهش محسوس حجم محاسبات آن قابل چشم پوشی است. در جدول ۳-۱۲ دقت بر اساس مجموعه دادگان آموزشی ImageNet، تعداد عملیات های ضرب یا جمع اسکالر در اسکالر در هر بار اجرای روند پیشرو و تعداد پارامترهای کل دو ساختار مقایسه شده است. تابع فعال ساز استفاده شده در این ساختار ReLU6 است.

جدول ۳-۱۲: مقایسه حجم محاسبات و دقت دو ساختار، با و بدون استفاده از ضرب جدائی پذیر[۱۷۰]

Model	ImageNet Accuracy	Million Mult-Adds	Million Parameters
Conv MobileNet	71.7%	4866	29.3
MobileNet	70.6%	569	4.2

در ادامه توسعه این ساختار در نسخه دوم آن اتصالات جهش مشابه شبکه ResNet نیز به آن اضافه شد که با عنوان MobileNet V.2 شناخته می شود، همچنین در این نسخه به انتهای بلوک یک لایه تنگنا، لایه پیچشی ۱×۱، اضافه شد تا عمق تانسور ورودی و خروجی در هر بلوک ثابت باشد. هدف از اضافه کردن این لایه تنگنا به بلوک های این ساختار کاهش عمق تانسور خروجی و در نتیجه حجم محاسبات و همچنین استخراج ویژگی های بهتر با نگاشت یک تانسور با عمق بالاتر به عمق کمتر، مشابه ایده ای که در خودرمزگذارها[1] با کاهش بعد ورودی به بردار پنهان استفاده می شود است. در ادامه در نسخه سوم از این ساختار که به نام MobileNet V.3 شناخته می شود، تعداد کرنل و سایر پارامترهای ساختار با استفاده از روش های جستجوی ساختار نورونی[2] که در فصل بررسی خواهیم کرد بهینه شده و به تانسور های بلوک آن مکانیزم توجه[3] اضافه شد که در ادامه در بخش مربوط ساختارهای پیچشی دارای مکانیزم توجه آن را معرفی خواهیم کرد.

[1] Autoencoders
[2] Neural architecture search (NAS)
[3] Attention mechanism

۱۲.۱۵.۲ ساختارهای پیچشی با خروجی متغیر (ساختارهای پیچشی مبتنی بر ناحیه[1])

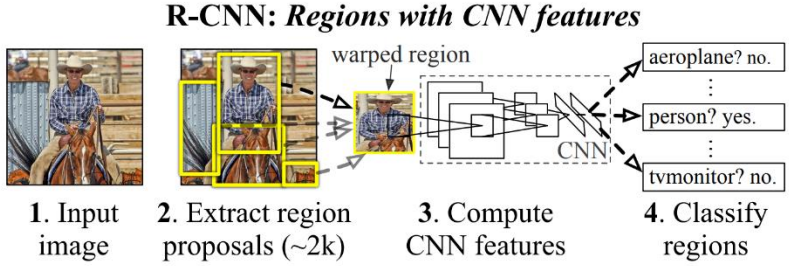
در ساختارهای پیچشی که تا کنون بررسی کردیم برای کل تصویر ورودی، یک مقدار مطلوب در مجموعه دادگان آموزشی تعریف می شود که همان خروجی نهائی شبکه بوده و ابعاد آن به ازای همه نمونه های مجموعه دادگان آموزشی برابر است، از این رو از این ساختارها بیشتر در طبقه بندی کلی یک تصویر استفاده می شود در صورتی که ممکن است یک تصویر دارای چند کلاس مختلف، دارای چند جسم مختلف، باشد و هر یک از آن ها قسمتی از فضای تصویر را اشغال کرده باشند همچنین ممکن است تعداد این کلاس ها در هر یک از تصاویر مجموعه دادگان آموزشی متفاوت باشد. در این صورت اگر فرض کنیم مقادیر مطلوب مورد نظر شامل کلاس و موقعیت هر جسم، کلاس مشخص، شده در تصویر باشد، بردار مقادیر مطلوب به ازای هر تصویر از مجموعه دادگان آموزشی دارای بعد متفاوتی خواهد بود، از طرفی تعداد نورون ها در آخرین لایه تماما متصل باید با بعد بردار مقادیر مطلوب برابر باشد. با این تفاسیر اگر از یکی از ساختارهای دارای لایه های تماما متصل برای چنین مجموعه دادگان آموزشی استفاده کنیم لایه انتهایی ساختار به ازای هر نمونه متفاوت خواهد بود، این امر متناقض با روند کلی تعریف شده برای شبکه های عمیق است که در آن ساختار شبکه در طول روند آموزش، تست و ارزیابی باید ثابت بوده و تنها مقادیر پارامترهای آموزشی آن در مرحله آموزش تغییر می کند. برای مرتفع کردن این معضل ساختارهای پیچشی با خروجی متغیر معرفی شده اند، در این ساختار ها کلاس[2] و موقعیت اجسام[3] موجود در هر تصویر به عنوان خروجی ساختار در نظر گرفته می شود. همچنین اگر حجم محاسبات این ساختارها بهینه بوده و سخت افزار مورد استفاده توان پردازشی مناسبی داشته باشد می توان این ساختارها را به ازای توالی تصاویر، ویدئو، اجرا کرده و از آن به عنوان مدل ردیابی کننده اجسام[4] در ویدئوها استفاده کنیم. در ادامه به بررسی چند مورد از این ساختارها می پردازیم.

۱۲.۱۵.۲.۱ شبکه R-CNN

این شبکه یکی از اولین ساختارهای پیچشی با خروجی متغیر است که در سال ۲۰۱۴ میلادی معرفی شده است. در این ساختار ابتدا با استفاده از یک روش غیر آموزشی[5] موسوم به جستجوی انتخابی[6]، نواحی[7] مربوط به اجسام مختلف در تصویر مشخص می شوند سپس این نواحی با یکی از الگوریتم های تغییر سایز که مانند Bilinear، به یک سایز استاندارد مربعی با ابعاد ۲۲۷×۲۲۷ تبدیل شده و مطابق شکل ۳۰-۱۲ وارد یک ساختار پیچشی مشابه ساختارهای دارای لایه های

[1] Region based CNNs
[2] Classification
[3] Object detection
[4] Object tracker
[5] Non-trainable
[6] Selective search
[7] Region of interest (ROI)

تماما متصل می شود، بردار حاصل از لایه های تماما متصل وارد یک ماشین بردار پشتیبان شده و کلاس مربوط به آن ناحیه با به عنوان خروجی ماشین بردار پشتیبان حاصل می شود. در این ساختار پارامترهای مربوط به ساختار پیچشی شامل کرنل های پیچشی و وزن های لایه های به صورت از پیش آموزش داده شده[1] با استفاده از یک مجموعه دادگان مناسب برای ساختارهای دارای لایه های تماما متصل در دسترس بوده و در طی روند آموزش ساختار فقط ماشین بردار پشتیبان آموزش داده می شود.

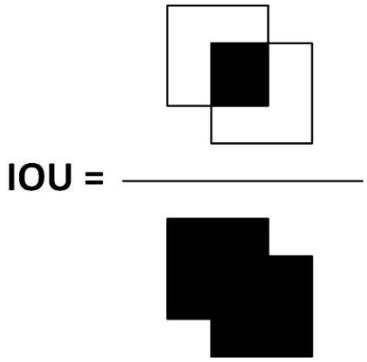


شکل ۳۰-۱۲: ساختار پیچشی RCNN [۱۷۲]

در مجموعه دادگان آموزشی این ساختار مقادیر مطلوب به ازای نواحی مختلفی که از قبل مشخص شده در هر تصویر تعریف می شوند. از طرفی تعدادی ناحیه نیز در روند اجرای ساختار توسط الگوریتم جستجوی انتخابی مشخص می شوند، ممکن است با نواحی که مقادیر مطلوب مجموعه دادگان آموزشی به ازای آن ها تعریف شده است متفاوت باشند. برای حل این مشکل پارامتر اشتراک بر اجتماع، IOU[2]، برای دو ناحیه از یک تصویر مطابق رابطه ۹۱-۱۲، نسبت مساحت اشتراک به اجتماع دو ناحیه مستطیلی[3] از تصویر تعریف می شود. شکل ۳۱-۱۲ نمایش پارامتر اشتراک بر اجتماع رابطه برای دو ناحیه مختلف از یک تصویر است.

$$IOU = \frac{Area(\text{Intersection})}{Area(\text{Union})} \qquad (۹۱-۱۲)$$

[1] Pre-train
[2] Intersection over union
[3] Bounding box



شکل ۳۱-۱۲: IOU

در صورتی که مقدار اشتراک بر اجتماع محاسبه شده به ازای ناحیه مشخص شده در مجموعه دادگان و ناحیه حاصل از الگوریتم جستجوی انتخابی بیشتر از یک حد آستانه[1] مشخص باشد مقدار مطلوب، کلاس، ناحیه مشخص شده در مجموعه دادگان به آن ناحیه که توسط الگوریتم جستجوی انتخابی استخراج و سپس وارد ساختار پیچشی شده است نسبت داده می شود. در صورتی که مقدار پارامتر بین ناحیه الگوریتم جستجوی انتخابی با چند ناحیه مشخص شده در مجموعه دادگان از حد آستانه بیشتر باشد مقدار مطلوب، کلاس، ناحیه ای که بیشترین مقدار اشتراک به اجتماع را دارد به آن ناحیه نسبت داده می شود.

در مجموعه تصاویر آموزشی این ساختار علاوه بر خروجی مربوط به ماشین بردار پشتیبان، کلاس، موقعیت هر ناحیه نیز به عنوان بخشی از مقادیر مطلوب مشخص شده است. اما به با توجه به عدم امکان آموزش الگوریتم جستجوی انتخابی امکان استفاده از مقادیر مطلوب مربوط به موقعیت نواحی برای بهبود نتایج نیست. برای استفاده از این مقادیر در کنار ماشین بردار پشتیبان یک مدل رگرسیون خطی مشابه رابطه ۴-۶ فصل۶ نیز تعریف شده و به همراه ماشین بردار پشتیبان آموزش داده می شود. مقادیر مطلوب برای موقعیت یک ناحیه مستطیلی شامل یک بردار با بعد چهار مختصات مرکز x, y، سطر و ستون مربوط به پیکسل مرکزی هر ناحیه، و همچنین ارتفاع و عرض h, w ناحیه مستطیلی می شود. اما مقدار مطلوب تعریف شده برای مدل رگرسور خطی اختلاف، $\Delta x, \Delta y, \Delta h, \Delta w$، این مقادیر مطلوب مربوط به موقعیت ناحیه با مقادیر تعیین شده برای ناحیه حاصل از الگوریتم جستجوی انتخابی است. به عبارت دیگر با آموزش مدل رگرسیون در این ساختار میزان انحراف[2] ناحیه الگوریتم جستجوی انتخابی محاسبه شده و با اعمال آن به موقعیت حاصل از الگوریتم جستجوی انتخابی موقعیت مطلوب که در مجموعه دادگان آموزشی مشخص شده است محاسبه می شود. بدین ترتیب برای مقادیر مربوط موقعیت نواحی نیز الگوریتم آموزشی تعریف می شود.

این ساختار با توجه به عدم آموزش پارامترهای ساختار پیچشی آن عملکرد چندان مطلوبی

[1] Threshold
[2] Offset

ندارد اما بررسی آن به لحاظ اینکه یکی از اولین ساختارهای پیچشی مبتنی بر ناحیه حائز اهمیت است. از طرفی با توجه به روند کند اجرای الگوریتم جستجوی انتخابی، زمان اجرای کل ساختار بالا بوده و امکان استفاده از آن به عنوان یک ردیاب جسم[1] برای دادگان ویدئویی نیست.

۱-۱-۲-۱۵-۱۲ جستجوی انتخابی

همان طور که اشاره کردیم این روش یک روش غیر آموزشی است که برای استخراج نواحی مختلف یک تصویر، نواحی مربوط به اجسام مختلف در تصویر، استفاده می شود. روند اجرای الگوریتم به ترتیب ذیل است:

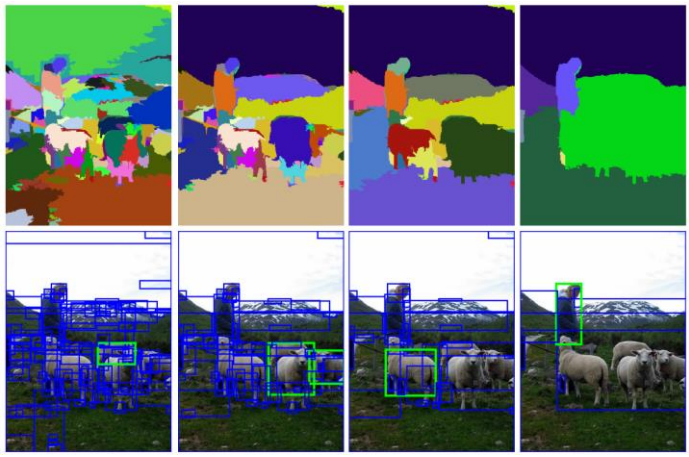
۱. ابتدا با استفاده از روش قطعه بندی معنایی مبتنی بر گراف[2] نواحی اولیه استخراج می شوند؛ روند قطعه بندی معنایی مبتنی بر گراف بدین صورت است که ابتدا برای یک تصویر، گراف متناظری تعریف می شود؛ در این گراف هر پیکسل از تصویر به عنوان یک گره[3] در نظر گرفته شده سپس بین گره هایی که در مرتبط به پیکسل های کنار در هم تصویر هستند یال[4] های بدون جهتی تعریف می شود. هر یال دارای یک مقدار عددی مثبت، وزن است، مقدار وزن بین هر دو گره با میزان تفاوت ویژگی های دو گره، شدت نور[5]، رنگ و...، متناسب است؛ بدین ترتیب هر چقدر تفاوت بین دو پیکسل بیشتر باشد احتمالا آن دو پیکسل مربوط به دو جسم مجزا بوده و مقدار وزن یال بین آن ها در گراف عدد بزرگ تری خواهد بود.

۲. پس از تعریف گراف و محاسبه وزن های مربوط به یال های آن، گره هایی از گراف که با مقدار وزن کمتر به هم متصل شده اند به عنوان پیکسل های مربوط به یک جسم مشخص شده و ناحیه اشغال شده توسط آن در تصویر اصلی به صورت یک ناحیه مستطیلی[6] مشخص می شود. این نواحی حاصل از گراف به عنوان نواحی اولیه در نظر گرفته می شوند. در مواردی ممکن است با توجه به محاسبات مربوط به وزن یال های گراف تعدادی از گره ها در چند زیر مجموعه قرار گرفته و در نتیجه نواحی مربوط به آن ها در تصویر هم پوشانی داشته باشند.

۳. پس از استخراج نواحی اولیه، شباهت مجموعه پیکسل های مربوط به نواحی که در کنار هم قرار دارند و یا با هم، هم پوشانی دارند از نظر رنگ، بافت[7]، مقایسه ویژگی

[1] Object tracker
[2] Graph-based segmentation
[3] Node
[4] Edge
[5] Intensity
[6] Bounding box
[7] Texture

هایی مانند SIFT [1] بین دو ناحیه، و... با هم مقایسه شده و اگر این شباهت بیش از مقدار حد آستانه در نظر گرفته شده باشد دو ناحیه با هم ادغام شده و به صورت یک ناحیه مربوط به یک کلاس در نظر گرفته می شوند. در نهایت تعداد کمتری ناحیه نسبت به نواحی اولیه از تصویر استخراج شده و به عنوان خروجی روش جستجوی انتخابی در نظر گرفته می شوند. شکل ۳۲-۱۲ از چپ به راست روند ادغام نواحی کوچکتر و تشکیل نواحی بزرگتر را که خروجی نهائی الگوریتم جستجوی انتخابی است را نشان می دهد.



شکل ۳۲-۱۲: روند ادغام نواحی در الگوریتم جستجوی انتخابی [۱۷۵]

۱۲.۱۵.۲.۲ شبکه Fast R-CNN

این شبکه در سال ۲۰۱۵ برای رفع برخی از نقص های شبکه R-CNN معرفی شده است. در این ساختار نیز ابتدا موقعیت نواحی مختلف با استفاده از روش جستجوی انتخابی استخراج می شوند، اما این نواحی بر خلاف شبکه R-CNN از تصویر اصلی جدا نشده و فقط موقعیت آن ها به صورت نواحی مستطیل شکل [2] در تصویر اصلی مشخص می شود، سپس یک ساختار پیچشی از پیش آموزش داده شده که لایه های تماما متصل و لایه مسطح ساز آن حذف شده باشد، VGG-16، به کل تصویر ورودی اعمال می شود، فرض کنیم اندازه تصویر ورودی ۳×۵۱۲×۵۱۲ باشد و در صورت اعمال شبکه VGG-16 به آن تبدیل به یک تانسور سه بعدی با ابعاد ۵۱۲×۱۶×۱۶ می شود. در این صورت طول و عرض تصویر با مقیاس $\frac{512}{16} = 32$ کاهش پیدا کرده است. فرض کنیم موقعیت یک ناحیه مستطیلی مستخرج از روش جستجوی انتخابی نیز دارای مختصات راس سمت چپ بالایی x, y و ارتفاع و عرض w, h باشد. در این صورت این ابعاد نیز با مقیاس برابری

[1] Scale-invariant feature transform
[2] Region of interest (ROI)

کاهش می یابند. بدین ترتیب مختصات نظیر ناحیه مشخص شده روی تصویر ورودی بر روی نگاشت ویژگی حاصل از شبکه VGG-16 به مطابق روابط ۹۲-۱۲ تا ۹۵-۱۲ است.

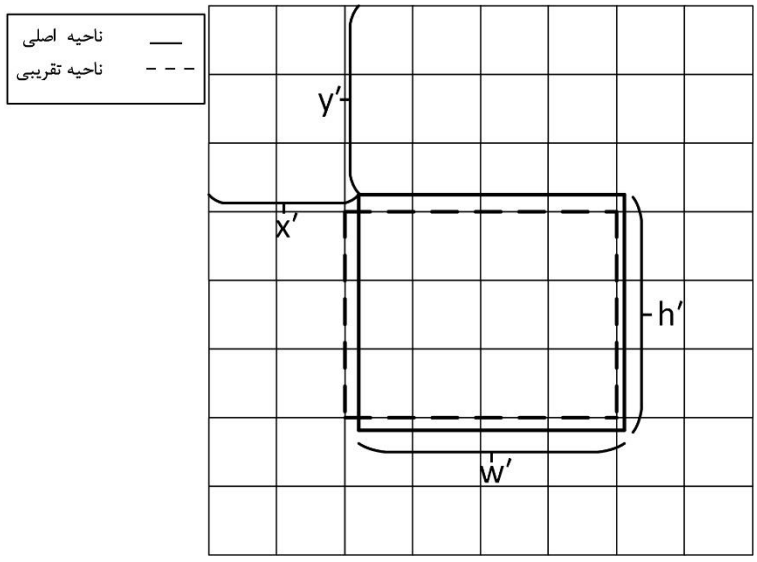
$$x' = \frac{x}{32} \quad (۹۲-۱۲)$$

$$y' = \frac{y}{32} \quad (۹۳-۱۲)$$

$$h' = \frac{h}{32} \quad (۹۴-۱۲)$$

$$w' = \frac{w}{32} \quad (۹۵-۱۲)$$

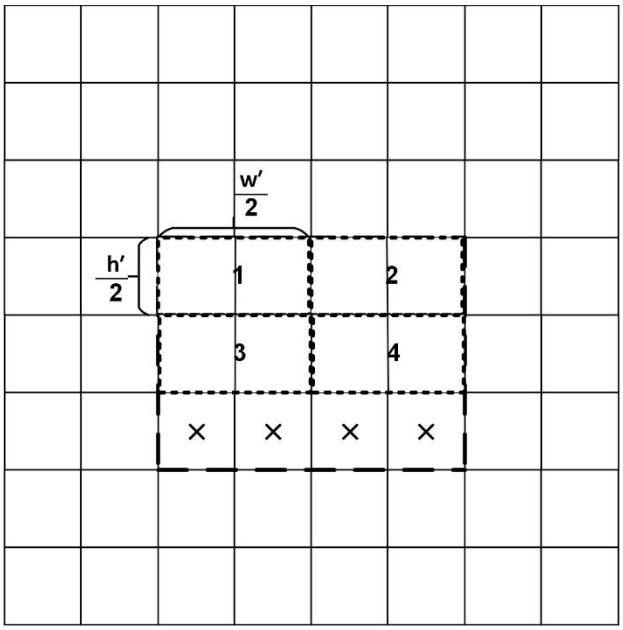
می توانیم ناحیه دارای مختصاتی مطابق روابط ۹۲-۱۲ تا ۹۵-۱۲ را به صورت مستطیلی[1] در راستای طول و عرض نگاشت ویژگی حاصل از ساختار پیچشی رسم کنیم. در صورتی که هر یک از اضلاع این مستطیل دقیقا به مرز جدا کننده بین دو درایه در نگاشت ویژگی منطبق نشوند موقعیت این اضلاع را برابر با نزدیک ترین مرز بین دو درایه به آن ها مطابق شکل ۳۳-۱۲ به صورت تقریبی در نظر می گیریم.



شکل ۳۳-۱۲: محل ناحیه بر روی نگاشت ویژگی

[1] Bounding box

در نهایت با جداسازی ناحیه مذکور از همه نگاشت های ویژگی، در راستای عمق تانسور خروجی ساختار پیچشی، یک تانسور با ابعاد $\acute{w}, h, 512$ که درایه $1,1,n$ آن درایه x, y, n تانسور نگاشت ویژگی اصلی است، با فرض انطباق مقادیر به مرز بین دو درایه و برابر بودن دو ناحیه مشخص شده در شکل ۳۳-۱۲. سپس یک لایه تجمیع حداکثری دو بعدی برای مثال با ابعاد ۲×۲، در ساختار اصلی این ابعاد ۳×۳ تعریف شده است) تعریف می کنیم. تانسور را در راستای ارتفاع و عرض به دو، ۲×۲، تانسور کوچکتر با ابعاد $512 \times \frac{\acute{w}}{2} \times \frac{h}{2}$ مطابق شکل ۳۴-۱۲ تقسیم می کنیم.



شکل ۳۴-۱۲: تقسیم ناحیه به تانسورهای کوچکتر

مطابق شکل ۳۴-۱۲ با توجه به ابعاد تانسور ممکن است برخی از این نواحی در تقسیم تانسور دخیل نشوند از این ابعاد در محاسبات صرف نظر می کنیم، مانند درایه های ستون انتهائی در شکل ۳۴-۱۲.

این روند تقسیم را به کل عمق های تانسور نگاشت ویژگی ادامه می دهیم، بدین ترتیب در کل ۵۱۲×۲×۲ پنجره با اندازه خواهیم داشت، در ادامه درایه با مقدار حداکثری از هر کدام از این پنجره ها جدا شده و همانند لایه مسطح ساز به صورت یک بردار زیر هم قرار گرفته و وارد لایه تماما متصل می شود. به این روند که در آن مقادیر حداکثری صرفا از محدوده درایه های مربوط به ناحیه جسم استخراج می شوند لایه تجمیع ناحیه[1] گفته می شود.

در نهایت خروجی لایه تماما متصل وارد یک لایه SoftMax جهت تشخیص کلاس جسم داخل

[1] ROI pooling layer

ناحیه و یک لایه تماما متصل رگرسور برای محاسبه انحراف بین مختصات ناحیه مشخص شده توسط روش جستجوی انتخابی و مختصات مشخص شده در مجموعه دادگان آموزشی می شود. تابع هزینه ساختار نیز به صورت مجموع دو تابع هزینه مربوط به لایه SoftMax و رگرسور محاسبه می شود. روند توضیح داده شده برای حالتی است که تنها یک ناحیه توسط روش جستجوی انتخابی مشخص شده است، در صورت وجود چند ناحیه، لایه تجمیع ناحیه برای هر کدام از آن ها جداگانه اعمال شده و هر بار خروجی آن که همواره یک بردار با اندازه ثابت است وارد لایه تماما متصل و در ادامه لایه SoftMax و رگرسور شده و مقدار تابع هزینه برای هر کدام از آن ها جداگانه محاسبه می شود. مقدار تابع هزینه مربوط به هر تصویر برابر میانگین مقادیر تابع هزینه به ازای ناحیه های داخل آن است.

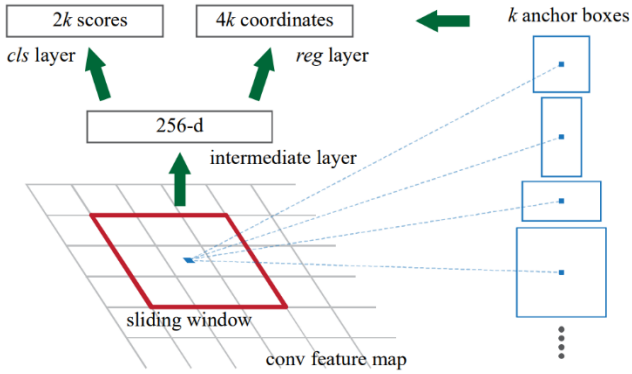
در این ساختار برخلاف ساختار R-CNN، امکان آموزش همه وزن های ساختار با توجه به جایگزینی ماشین بردار پشتیبان وجود داشته و از این نظر برتری محسوسی در عملکرد نسبت به ساختار R-CNN دارد. علاوه بر این با توجه به اینکه فقط یک بار محاسبات مربوط به لایه های پیچشی برای هر تصویر ورودی اجرا می شود نسبت به شبکه R-CNN حجم محاسبات کمتر و در نتیجه سرعت اجرای بالاتری، زمان اجرای[1] کمتری، دارد اما این بهبود سرعت با توجه به اینکه همچنان از روش جستجوی انتخابی استفاده می شود چندان محسوس نیست.

۱۲.۱۵.۲.۳ شبکه Faster R-CNN

این شبکه در سال ۲۰۱۶ برای بهبود عملکرد شبکه Fast R-CNN معرفی شده است. در این شبکه بر خلاف دو شبکه پیشین روش جستجوی انتخابی حذف شده و از یک شبکه برای محاسبه ناحیه اجسام استفاده می شود. روند اجرای این شبکه بدین صورت است که ابتدا تصویر ورودی وارد یک ساختار پیچشی شده و تبدیل به نگاشت های ویژگی به صورت یک تانسور سه بعدی می شود. در راستای طول و عرض این تانسور یک پنجره لغزان[2] مطابق شکل ۳۵-۱۲ تعریف می شود. این پنجره لغزان در طول همه عمق های تانسور نگاشت های ویژگی مشابه یک کرنل پیچشی سه بعدی با عمقی برابر با عمق تانسور ورودی گسترش یافته و در هر گام حرکتی تعدادی از درایه های تانسور نگاشت های ویژگی را در راستای طول و عرض و همه عمق های آن جدا کرده و مشابه لایه مسطح ساز درایه های آن را کنار هم قرار داده و به یک بردار تبدیل می کند. این بردار در هر گام حرکتی وارد یک شبکه متشکل از لایه های تماما متصل شده و مطابق شکل ۳۵-۱۲ k ناحیه پیشنهادی به صورت بردار مختصات $4k$ بعدی به عنوان خروجی شبکه محاسبه می شود. این شبکه با عنوان شبکه پیشنهاد ناحیه[3] شناخته می شود. به هر کدام از k ناحیه ای که مختصات آن ها توسط شبکه

[1] Inference time
[2] Sliding window
[3] Region proposal network (RPN)

پیشنهاد ناحیه محاسبه می شود جعبه لنگر[1] گفته می شود.



شکل ۳۵-۱۲: پنجره لغزان در ساختار Faster R-CNN [۱۷۴]

در ادامه به هر یک از نواحی که جعبه های لنگر در تانسور نگاشت های ویژگی مشخص می کند یک لایه تجمیع ناحیه اعمال شده و در نهایت خروجی لایه تجمیع هر ناحیه به صورت یک بردار وارد لایه های تماما متصل شده و انحراف هر ناحیه پیشنهادی از مقدار مطلوب به همراه احتمال تعلق آن به یک کلاس به صورت بردار دو بعدی به عنوان خروجی نهائی شبکه مشابه شبکه Fast R-CNN محاسبه می شود. بردار های خروجی نهائی ساختار در بالای شکل ۳۵-۱۲ نشان داده شده است، با توجه به شکل ۳۵-۱۲ در خروجی نهائی این شبکه یک بردار دو بعدی برای تشخیص کلاس مربوط به هر کدام از k ناحیه حاصل از یک گام حرکتی پنجره لغزان تعریف شده است که یک بعد آن مربوط به وجود جسم و بعد دوم مربوط به عدم وجود جسم داخل ناحیه است، در حالی که می دانیم تنها با یک بعد و تعریف مقادیر مطلوبی مانند صفر و یک برای آن می توان وجود یا عدم وجود جسم را تعیین کرد. دلیل استفاده از دو بعد در این ساختار تعریف تئوریکال امکان تعمیم خروجی به تعداد کلاس های بیشتر، زیر کلاس[2] های مربوط به وجود جسم که کلاس جسم را مشخص می کند، و متعاقبا تعریف بردارهای مقادیر مطلوب با بعد دلخواه به صورت بردارهای One hot است؛ بدین صورت که برای تشخیص کلاس های مختلف، در صورت وجود جسم، می توان بعد این بردار را به صورت $(n+1)k$ در نظر گرفت که n بعد اول مربوط به کلاس ها که به صورت بردار One hot تعریف می شود و یک بعد اضافه شده مربوط به کلاس عدم وجود جسم در ناحیه است.

برای نواحی که مختصات آن ها با ناحیه مشخص شده به عنوان مقادیر مطلوب در مجموعه دادگان دارای مقدار IOU بیش از حد آستانه ۰.۷ باشد بعد مربوط به وجود جسم برابر با یک و برای نواحی که IOU آن کمتر از ۰.۳ باشد بعد مربوط به عدم وجود جسم برابر با یک، و بعد وجود جسم

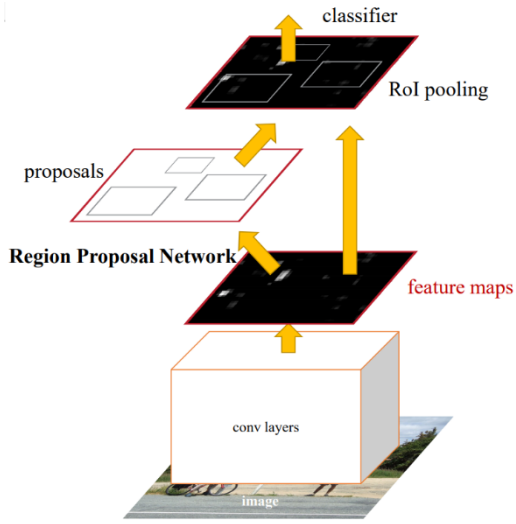
[1] Anchor box
[2] Sub-classes

برابر با صفر، در بردار One hot مقادیر مطلوب تعریف می شود، نواحی با IOU بین ۰.۳ و ۰.۷ در محاسبه تابع هزینه دخیل نیستند. در نهایت تابع هزینه شبکه به صورت رابطه ۹۶-۱۲ تعریف می شود که مطابق آن محاسبه ای برای انحراف مختصات نواحی کمتر از حد آستانه ۰.۳ تعریف نمی شود و جمله دوم صرفا در صورت وجود جسم در مقدار تابع هزینه دخیل می شود. این شبکه به نسبت دو ساختار پیشین دارای سرعت اجرای بهتری است اما همچنان سرعت آن به حدی نیست که به عنوان یک روش ردیابی اجسام به صورت بلادرنگ[1] برای پردازش توالی تصاویر، ویدئو، استفاده شود. در رابطه ۹۶-۱۲، $N_{cls}, N_{reg}, \lambda$ به ترتیب یک اسکالر برای کنترل تاثیر جمله دوم، مجموع تعداد ابعاد مربوط به کلاس ها به ازای همه نمونه ها، $(n+1)$ ×اندازه دسته[2]، و تعداد ابعاد مربوط به مختصات، 4 ×اندازه دسته، است. همچنین t_i مجموعه مقادیر مطلوب شامل ۴ بعد مربوط به مختصات نواحی و p_i مقدار احتمال مربوط به کلاس وجود جسم، یکی از زیر کلاس های وجود جسم، است، $\hat{p}_i, \hat{t}_i$ خروجی های شبکه متناظر با مقادیر مطلوب و i زیروند شاخص مربوط به هر نمونه ورودی است.

$$E_{([\hat{p}_i],[\hat{t}_i])} = \frac{1}{N_{cls}} \sum_i L_{cls}(p_i, \hat{p}_i) + \lambda \frac{1}{N_{reg}} \sum_i p_i L_{cls}(t_i, \hat{t}_i) \quad (۱۲-۹۶)$$

در این ساختار با توجه به اینکه موقعیت اولیه نواحی اجسام داخل تصویر با استفاده از یک شبکه محاسبه می شوند، در صورتی که مقیاس و ابعاد اجسام داخل تصاویر با تغییر مجموعه دادگان آموزشی به مجموعه ای دیگر تغییر کند. با آموزش شبکه مذکور موقعیت نواحی نیز نسبت به دادگان تغییر می کند. این ویژگی با عنوان Translation-Invariance شناخته می شود. این ویژگی در ساختارهایی که در آن ها موقعیت این نواحی با روش های غیر آموزشی محاسبه می شوند لحاظ نشده و مقیاس و ابعاد نواحی صرفا تابع قیدها و پارامترهای تعریف شده برای الگوریتم غیرآموزشی تشخیص نواحی بوده و فاقد انعطاف پذیری مناسب است. شکل ۳۶-۱۲ شمای کلی ساختار Faster R-CNN را نشان می دهد.

[1] Real time
[2] Batch size



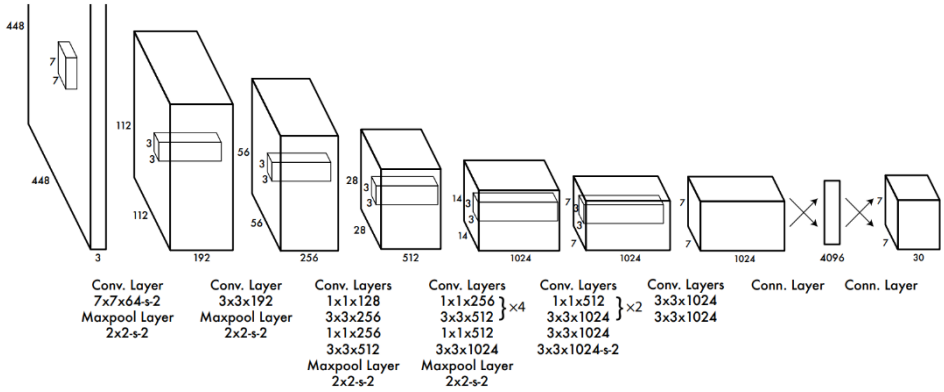
شکل ۳۶-۱۲: شمای کلی ساختار شبکه Faster R-CNN [۱۷۴]

۱۲.۱۵.۲.۴ شبکه YOLO[1]

در همه ساختارهایی که در این بخش معرفی کردیم، ناحیه ای از تصویر اصلی[2] جدا شده و کلاس مربوط به جسم داخل آن ناحیه به همراه انحراف مختصات ناحیه از مقدار مطلوب مشخص می شد. اما در ساختار YOLO برای کل نواحی یک تصویر ورودی احتمال وجود جسم، موقعیت و کلاس مربوط به آن ها محاسبه می شود؛ در ساختارهای پیشین نواحی استخراج شده به ترتیب وارد لایه بعدی ساختار شده و در نهایت کلاس و انحراف مختصات هر کدام از آن ها به عنوان خروجی نهائی شبکه محاسبه می شد این روند به ویژه در صورتی که برای یک تصویر نواحی زیادی به عنوان نواحی اولیه دارای جسم تشخیص داده شود باعث کندی اجرای شبکه می شود. در حالی که در ساختار YOLO این روند به ازای همه نواحی در یک مرحله اجرا می شود. ساختار این شبکه مطابق شکل ۳۷-۱۲ است. این ساختار شامل ۲۴ لایه پیچشی سه بعدی و دو لایه تماما متصل است. پارامترهای مربوط به قسمت پیچشی ساختار ابتدا با استفاده از مجموعه دادگان ImageNet با اندازه ورودی ۳×۲۲۴×۲۲۴ به صورت یک شبکه طبقه بند آموزش داده شده و پارامترهای حاصل به عنوان وزن های اولیه در نظر گرفته شده است، سپس برای ساختار اصلی YOLO سایز ورودی دو برابر تعریف شده، ۳×۴۴۸×۴۴۸، و مجددا بر روی پارامترهای اولیه یک آموزش با سرپرست با استفاده از مجموعه دادگان اصلی شامل تصاویر دارای مقادیر مطلوب به صورت کلاس و موقعیت آن ها در تصویر، اعمال شده است. در این ساختار به بردار خروجی آخرین لایه تماما متصل تغییر ابعاد[3] اعمال شده و خروجی آن به یک تانسور سه بعدی با ابعاد ۳۰×۷×۷ تبدیل می شود؛ روندی مشابه

[1] You only look once
[2] ROI
[3] Reshape

معکوس روند لایه مسطح ساز. این تانسور که از تغییر ابعاد بردار خروجی لایه تماما متصل حاصل می شود همان خروجی نهائی ساختار است. از این رو مقادیر مطلوب ساختار نیز به صورت یک تانسور ۷×۷×۳۰ تعریف می شود که در ادامه آن را بررسی خواهیم کرد.

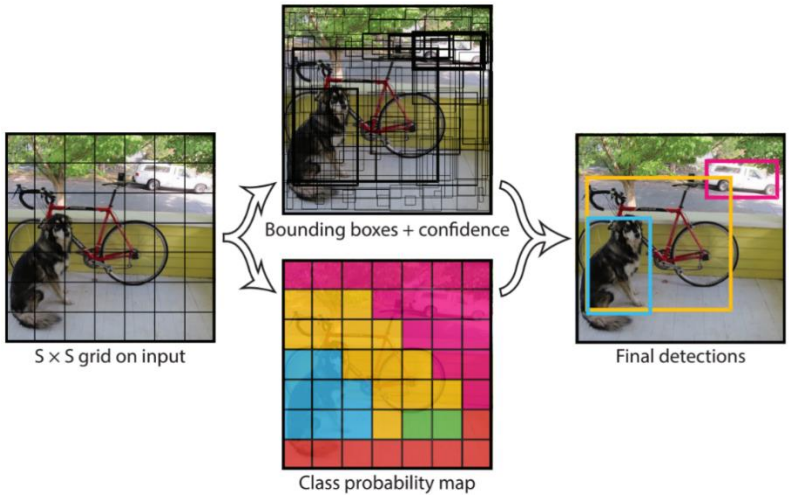


شکل ۳۷-۱۲: ساختار شبکه YOLO [۱۰۳]

تصویر ورودی در ساختار شبکه YOLO مطابق شکل ۳۸-۱۲ سمت چپ به بخش[1] های فرضی مساوی تقسیم شده و در نهایت به صورت یک تصویر $s \times s$، $s = 7$، تعریف می شود. برای هر یک از این بخش ها یک بردار با اندازه $B \times 5 + C$ به عنوان خروجی مطلوب متناظر با آن بخش تعریف می شود با کنار هم قرار دادن این بردارهای مطلوب با اندازه $B \times 5 + C$، یک تانسور $(B \times 5 + C) \times 7 \times 7$ به وجود می آید. در این ساختار همان طور که در ساختار Faster R-CNN برای هر گام حرکتی تعدادی محدوده پیشنهادی، k، تعریف می شد، برای هر یک از بخش های تصویر ورودی، s، تعداد B ناحیه مستطیلی[2] تعریف می شود؛ برای تعریف مختصات، نقطه مرکز ناحیه، عرض و ارتفاع آن، هر یک از این محدوده ها به یک بردار با اندازه چهار، x, y, w, h، نیاز داریم، همچنین برای محاسبه مقدار پارامتر IOU بین هر یک از نواحی پیشنهادی با ناحیه مشخص شده در دادگان آموزشی یک مقدار اسکالر به عنوان ضریب اطمینان[3] تعریف می شود که در کنار چهار بعد مربوط به مختصات در مجموع پنج پارامتر به هر ناحیه مستطیلی نسبت داده می شود، بدین ترتیب به هر یک از ناحیه های مستطیلی ضریب پنج در بردار مقادیر مطلوب اعمال می شود، $B \times 5$)، علاوه بر مختصات ناحیه و احتمال تعلق جسم به یک کلاس، C، نیز تعریف شده و به بردار مقادیر مطلوب بعد اضافه می شود تا مقدار مطلوب کلاس مربوط به جسم داخل ناحیه مستطیلی به صورت بردار One hot با اندازه C تعریف شود. در این ساختار عدم وجود جسم با اعمال حد آستانه به مقدار بعد مربوط به IOU تعریف می شود. با این توضیحات با فرض تعداد ۲

[1] Grid
[2] Bounding box
[3] Confidence score

ناحیه مستطیلی برای هر یک از بخش های تصویر ورودی، $B = 2$، و فرض ۲۰ کلاس تعریف شده برای اجسام داخل مجموعه تصاویر آموزشی، $C = 20$، ابعاد تانسور مقادیر مطلوب برابر ۳۰×۷×۷ خواهد بود که دارای ابعادی برابر با تانسور خروجی شبکه است که پیش تر به آن اشاره کردیم. شکل ۳۸-۱۲ تصویر ورودی در ساختار YOLO، شکل ۳۸-۱۲ چپ، تعیین مختصات نواحی مربوط به هر بخش، شکل ۳۸-۱۲ وسط بالا، کلاس هر بخش، شکل ۳۸-۱۲ وسط پایین، و حذف نواحی زائد به روش محو غیر حداکثری[1] که در ادامه معرفی خواهیم کرد، شکل ۳۸-۱۲ وسط راست، است.



شکل ۳۸-۱۲: تصویر ورودی و خروجی در ساختار YOLO [۱۰۳]

این ساختار نسبت به ساختارهای پیش تر معرفی شده سرعت اجرای بسیار بالایی داشته و از این رو به عنوان ردیاب اجسام می توان از آن استفاده کرد، در مقابل عدم تشخیص اجسام کوچک در تصویر و دقت پایین تر در مقایسه با ساختارهای پیچشی دارای لایه های تماما متصل که در بخش معرفی کردیم از معایب آن است. در ادامه توسعه این ساختار در نسخه دوم آن که با عنوان YOLO V.2، تغییراتی شامل استفاده از نرمال سازی دسته ای[2]، آموزش ساختار با تصاویر با ابعاد مختلف، کوچک تر کردن بخش های تصویر ورودی، حذف لایه های تماما متصل و تنظیم ابعاد خروجی با استفاده از لایه های تجمیع و پیچشی و استفاده از جعبه های لنگر[3] برای پیشنهاد ناحیه جسم و محاسبه انحراف ناحیه از جعبه لنگر به عنوان خروجی، مشابه Faster R-CNN، به جای موقعیت آن و تعریف ساختار با استفاده از ۱۹ لایه پیچشی، اعمال شد. در نسخه سوم این ساختار، YOLO V.3، از یک ساختار پیچشی ۵۳ لایه[4] و یک شبکه هرم ویژگی که در ادامه بررسی خواهیم

[1] Non-maximal suppression (NMS)
[2] Batch normalization
[3] Anchor boxes
[4] DarkNet-53

کرد نیز استفاده شد.

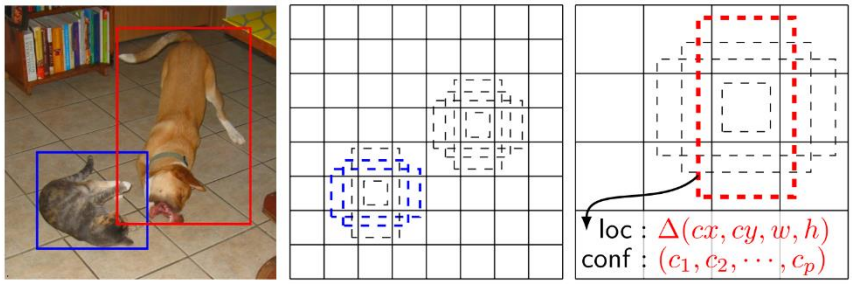
در نسخه دوم برای آموزش ساختار با تصاویری با ابعاد مختلف، ابتدا تصویر ورودی را ۳۲۰×۳۲۰ در نظر گرفته و آموزش ساختار را با آن اجرا می کنند، پس از اجرای روند آموزش به ازای ۱۰ دسته[1] ابعاد تصویر به ۳۵۲×۳۵۲ تغییر کرده[2]، متعاقبا پارامترهای وابسته در ساختار نیز تغییر می کنند، و روند آموزش برای ۱۰ دسته دیگر با ابعاد ورودی جدید ادامه پیدا می کند سپس به ابعاد ورودی ۳۲ پیکسل دوباره اضافه می شود، این روند ادامه می یابد تا تصویر ورودی به ابعاد ۶۰۸×۶۰۸، این عامل با تغییر مقیاس اجسام داخل تصویر مشکل مربوط به عدم تشخیص اجسام کوچک را تا حد زیادی برطرف می کند. همچنین در نسخه دوم برای پیشنهاد نواحی اولیه، جعبه های لنگر، از روش خوشه بندی K-Means استفاده شده است؛ بدین صورت که ابتدا روش خوشه بندی را با تعداد خوشه تعیین شده، $k = 5$، بر روی نواحی مشخص شده به عنوان نواحی مطلوب، مختصات مطلوب، در مجموعه تصاویر آموزشی اجرا کرده و سپس مراکز دسته ها را به عنوان جعبه های لنگر پیشنهادی اولیه برای هر بخش[3] از تصویر ورودی به طور مشترک در نظر می گیرند.

۱۲.۱۵.۲.۵ شبکه SSD[4]

شبکه SSD نیز همچون شبکه YOLO به ازای هر تصویر ورودی کلاس و ناحیه اجسام داخل آن را تشخیص می دهد. این شبکه در سال ۲۰۱۶ میلادی معرفی شده است. شکل ۳۹-۱۲ مقایسه این شبکه با ساختار نسخه اول شبکه YOLO است. مطابق شکل ۳۹-۱۲ شبکه SSD در ابتدا از یک شبکه VGG-16 که وزن های آن از پیش آموزش داده شده و به عنوان وزن اولیه در این ساختار در نظر گرفته شده است برای استخراج ویژگی استفاده می کند. همچنین مطابق شکل ۳۹-۱۲ در این ساختار خروجی لایه های پیچشی مختلف بخشی از نواحی، مختصات و کلاس نواحی، را به عنوان خروجی نهائی ساختار تشکیل می دهند. نحوه تبدیل این خروجی این لایه ها به قسمتی از خروجی نهائی تقریبا مشابه ساختار Faster R-CNN و ساختار YOLO است؛ بدین صورت که در ساختار Faster R-CNN شبکه پیشنهاد ناحیه یک پنجره لغزان بر روی ورودی تعریف کرده و به ازای هر گام حرکتی تعدادی ناحیه پیشنهادی ارائه می شد در حالی که در شبکه SSD پنجره لغزان با یک لایه پیچشی سه بعدی با کرنل های ۳×۳ جایگزین شده است. تعداد این کرنل های سه بعدی $k \times ((C + 1) + 4)$ است. منظور از k تعداد ناحیه های مستطیلی پیش فرض به عنوان نواحی پیشنهادی اولیه برای هر یک از درایه ها در راستای طول و عرض تانسور، و همه عمق های تانسور؛ این روند مشابه تقسیم تصویر ورودی به بخش های $s \times s$ در ساختار YOLO است با این تفاوت که هر بخش یک تانسور با ابعاد $1 \times 1 \times n$، n = تعداد کرنل های ۳×۳ عمق تانسور خروجی لایه پیچشی سه بعدی با تانسور های ۳×۳، است، C تعداد کلاس ها به اضافه یک کلاس

[1] Batch
[2] Resize
[3] Grid
[4] Single shot multibox detector (SSD)

مربوط به عدم وجود جسم در هر ناحیه و چهار ابعاد مربوط به انحراف مختصات هر یک از این ناحیه ها از مختصات نواحی پیش فرض است. برای مثال تانسور خروجی یکی از لایه های پیچشی ساختار را که به آن یک لایه ضرب پیچشی با کرنل هایی با ابعاد $3 \times 3 \times k \times ((C+1)+4)$ اعمال شده و به عنوان بخشی از ناحیه های خروجی تعریف می شود را در نظر بگیرید، این تانسور را می توانیم در راستای طول و عرض مطابق شکل ۴۰-۱۲ b یا c نشان دهیم. مطابق شکل ۴۰-۱۲ b و c برای هر یک از درایه های این تانسور در راستای طول و عرض، $n \times 1 \times 1$، k ناحیه پیش فرض در نظر می گیریم (مشابه ساختار YOLO که به هر بخش از تصویر ورودی B ناحیه نسبت داده می شد)، هر یک از نواحی دارای $C+4$ پارامتر مطابق شکل ۴۰-۱۲ c است که انحراف مختصات آن نسبت به ناحیه مقدار مطلوب، ۴۰-۱۲ a، و کلاس آن ناحیه را مشخص می کند. در ادامه این نواحی به همراه پارامترهای مربوط به انحراف مختصات و کلاس با مقادیر مطلوب موجود در مجموعه دادگان آموزشی مقایسه می شده و مقدار تابع هزینه ساختار محاسبه می شود، در ساختار Fast R-CNN اشاره کردیم که نواحی مربوط به اجسام مختلف در یک تصویر با همان مقیاسی که با اعمال لایه های پیچشی به ابعاد تصویر ورودی اعمال می شود تغییر می کنند. در این ساختار نیز که نواحی خروجی از لایه های پیچشی مختلف محاسبه می شوند با توجه به موقعیت لایه در ساختار، مقیاس ناحیه نسبت به نواحی تصویر اصلی محاسبه می شود. برای محاسبه انحراف مختصات هر یک از نواحی حاصل از تانسور های نگاشت ویژگی، ۴۰-۱۲ b و c، با مختصات مطلوب مجموعه دادگان، ۴۰-۱۲ a، باید هر دو ناحیه دارای مقیاس یکسان باشند، بدین منظور نسبت مقیاس مربوط به لایه ای که نواحی از آن استخراج شده است به مقیاس تصویر ورودی، به آن نواحی مستخرج از آن لایه اعمال می شوند تا بدین ترتیب مقیاس نواحی به مقیاس نواحی مقادیر مطلوب تبدیل شوند تا میزان انحراف مختصات، $\Delta x, \Delta y, \Delta h, \Delta w$، هر کدام از آن ها محاسبه شود. این تفاوت مقیاس نواحی لایه های مختلف با مقایسه سه تصویر شکل ۴۰-۱۲ نیز استنباط می شود.



(a) Image with GT boxes (b) 8×8 feature map (c) 4×4 feature map

شکل ۴۰-۱۲: محاسبه نواحی حاصل از نگاشت های ویژگی در ساختار SSD [۱۷۷]

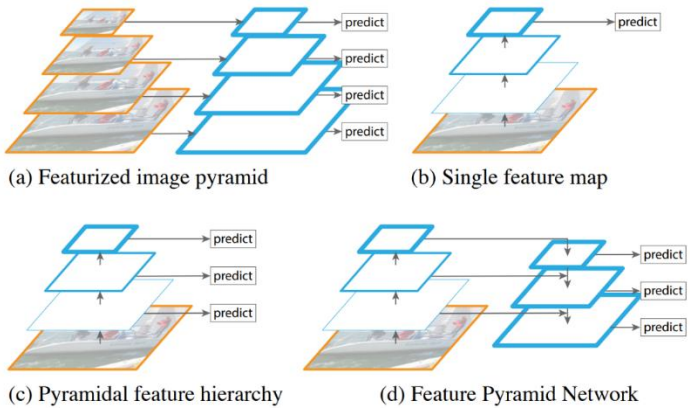
۱۲.۱۵.۲.۶ شبکه هرم ویژگی[1]

اشاره کردیم که خروجی لایه های ابتدایی در یک ساختار پیچشی دارای ویژگی های سطح پایین ولی با وضوح[2] و جزئیات بیشتر است، در حالی که لایه های نزدیک به انتهای ساختار ویژگی های سطح بالا را می سازند که حاوی اطلاعات کلی از تصویر با جزئیات نسبت کمتر هستند. در ساختارهایی که هدف کلی تشخیص کلاس کل یک تصویر است استفاده از ویژگی های سطح بالا می تواند عملکرد مطلوبی داشته باشد، تصویر a در شکل ۴۱-۱۲ شمای کلی چنین ساختاری است، حتی برخی از ساختارهایی که در آن ها محل جسم در تصویر نیز به عنوان خروجی تعریف می شود نیز دارای شمایی مشابه تصویر a شکل ۴۱-۱۲ هستند مانند نسخه اول ساختار YOLO. این در حالی است که در چنین ساختارهای پیچشی که محل جسم در تصویر نیز به عنوان خروجی تعریف می شود جزئیات تصویر در ورودی می تواند به بهبود عملکرد ساختار در تشخیص موقعیت اجسام[3] کمک کند، در حالی که ویژگی های سطح بالاتر در طبقه بندی عملکرد بهتری دارند. بنابراین در این ساختارها بهتر است ویژگی های سطح بالا و پایین در کنار هم در محاسبه خروجی دخیل شوند. یکی از راه های کنار هم قرار دادن هر ویژگی های سطح بالا و پایین تعریف اتصالات جهش است، اما استفاده از این اتصالات با توجه به تعریف لایه پیچشی برای تنظیم ابعاد در اتصال و استفاده از بعدگذاری با ابعاد بسیار بزرگ در صورتی که فاصله بین دو لایه متصل زیاد باشد از مواردی است که باعث می شود خروجی حاصل به ویژه در اتصال جهش با عملیات جمع حاوی اطلاعات مناسبی از هر دو نوع ویژگی های سطح بالا و پایین نباشد، از این رو بهبود حاصل از اتصالات جهش در این ساختارها محدود است.

راهکار دیگر بدین منظور استفاده از خروجی لایه های مختلف برای تعیین کلاس و موقعیت اجسام در یک تصویر است، این رویکرد در ساختار SSD به کار برده شده است، شمای کلی ساختارهای مبتنی بر این رویکرد مطابق تصویر c شکل ۴۱-۱۲ است. معضل مربوط به این روش عدم کنار هم قرار گرفتن همزمان ویژگی های سطح بالا و پایین است. راهکار نسبتا ساده دیگر، تبدیل تصویر به مقیاس های مختلف به صورت هرم مطابق تصویر a شکل ۴۱-۱۲ و استفاده از هر کدام از آن ها به عنوان ورودی یک ساختار متناسب با آن، ساختاری که لزوما پیچشی نیست، است، این راهکار تا حدی مشابه روش به کاربرده شده در روند آموزش نسخه دوم ساختار YOLO است اما به طور کلی با توجه به بهبود ناچیز در برابر افزایش حجم محاسبات و حافظه مورد نیاز، در مواردی که برای هر مقیاس ساختار متفاوتی تعریف می شود، در ساختارهای پیچشی مدرن کاربرد چندانی ندارد حتی در نسخه دوم ساختار YOLO نیز روش مذکور به طور محدود فقط در روند آموزش به کار برده شده است.

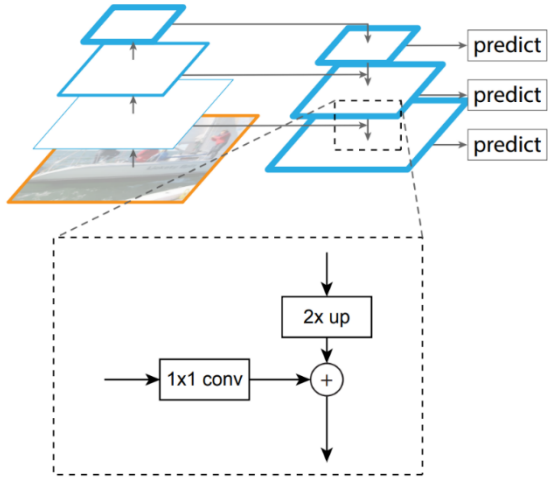
---

[1] Feature pyramid network (FPN)
[2] Resolution
[3] Object detection



شکل ۴۱-۱۲: مقایسه شماتیک ساختارهای مختلف با ساختار شبکه هرم ویژگی [۱۷۸]

با توجه به راهکارهای فوق الذکر بهترین راهکار از میان آن ها، روش به کاربرده شده در ساختارهایی مانند ساختار SSD است. شبکه هرم ویژگی برای رفع معضل این راهکار ارائه شده است. شمای این شبکه مطابق تصویر d شکل ۴۱-۱۲ است. بدین ترتیب در روند پیشرو این شبکه پس از اعمال یک ساختار پیچشی به تصویر خروجی بخش های مختلف ساختار حاوی سطوح مختلف ویژگی های تصویر ورودی مطابق قسمت سمت چپ تصویر d شکل ۴۱-۱۲ خواهند بود. ساختار پیچشی استفاده شده که چنین ویژگی هایی را می سازد ساختاری متشکل از پنج بلوک ResNet است. ابعاد طول و عرض تانسور خروجی هر بلوک نصف ورودی است. این مرحله از روند پیشرو ساختار روند پایین به بالا[1] است.



شکل ۴۲-۱۲: روند بالا به پایین واتصالات جانبی در شبکه هرم ویژگی [۱۷۸]

[1] Bottom-up pathway

در ادامه روند پیشرو ساختار روند بالا به پایین و اتصالات جانبی[1] آغاز می شود در این مرحله به تانسورهای خروجی هر بلوک یک لایه پیچشی سه بعدی با کرنل هایی با طول و عرض یک برای تنظیم عمق تانسور و یکسان سازی آن با عمق تانسور خروجی بلوک بعدی مطابق شکل ۴۲-۱۲ اعمال شده و تانسور مربوط به بلوک بالاتر نیز از آن جایی که دارای طول و عرضی به اندازه نصف تانسور فعلی، تانسور حاصل از لایه پیچشی ۱×۱، است، در راستای طول و عرض به روش نزدیک ترین همسایگی[2] ابعاد آن دو برابر شده و با تانسور فعلی مطابق شکل ۴۲-۱۲ جمع می شود بدین ترتیب اتصالات جانبی در این ساختار تعریف می شود که در پایین ۴۲-۱۲ به صورت شماتیک نشان داده شده است. بدیهی است که تانسور آخرین بلوک فقط با اعمال یک لایه پیچشی ۱×۱ حاصل شده و با تانسور دیگری جمع نمی شود. با توجه به امکان بروز پدیده الایزینگ[3] به دلیل استفاده از روش نزدیک ترین همسایگی به هر یک از تانسورهای حاصل جمع یک لایه پیچشی سه بعدی با کرنل های ۳×۳ اعمال می شود تا تاثیر آن کم شود. با اعمال این لایه پیچشی روند پیشرو ساختار به اتمام می رسد. بدین ترتیب تانسورهایی حاوی ویژگی های سطوح مختلف به عنوان خروجی ساختار تعریف می شوند.

شبکه هرم ویژگی معمولا به طور مستقل استفاده نشده و تانسورهای حاصل از آن در بخش های مختلف ساختارهای دیگر استفاده شده و بدین ترتیب این شبکه در ساختارهای دیگر ادغام می شود. برای مثال از این ساختار در شبکه Fast R-CNN به جای شبکه VGG-16 از شبکه هرم ویژگی استفاده کرده و بدین ترتیب به جای استفاده از یک تانسور در ورودی لایه تجمیع ناحیه از تانسورهای حاصل از هرم ویژگی استفاده کرد، رویکرد مشابهی را می توان در ساختار Faster R-CNN برای تعریف ورودی شبکه پیشنهاد ناحیه استفاده کرد. همچنین از این ساختار در نسخه سوم شبکه YOLO استفاده شده و بدین ترتیب مشکل مربوط به عدم پوشش مقیاس های مختلف تصویر که در نسخه اول ساختار وجود داشت برطرف شده است.

۱۲.۱۵.۲.۷ رویکردهای کلی استفاده شده در ساختارهای پیچشی با خروجی متغیر

در ساختارهای پیچشی که در این بخش معرفی کردیم علیرغم تفاوت ها و ویژگی های منحصر به فرد هر ساختار موارد مشترکی نیز بین همه آن ها وجود دارد. استفاده از چهار بعد جهت تعیین مختصات یک ناحیه که شامل یک نقطه، مرکز یا یکی از گوشه ها، از ناحیه، ارتفاع و عرض ناحیه است از جمله موارد مشترک در این ساختارها است همچنین در همه این ساختارها تعداد نواحی که به عنوان خروجی ساختار تعریف می شوند بسیار بیشتر از تعداد نواحی است که در مجموعه دادگان آموزشی به عنوان نواحی مطلوب مشخص شده اند. این وجه اشتراکات در این ساختارها باعث شده که مسائل و چالش های مشترکی ناشی از این موارد نیز در این ساختارها مشترک بوده و در نتیجه

[1] Top-down pathway and lateral connections
[2] Nearest neighbor upsampling
[3] Aliasing

راهکار مشترکی نیز برای رفع آن ارائه شود. در ادامه به معرفی چالش های ناشی از این موارد ذکر شده و راهکارهای ارائه شده برای رفع آن ها اشاره می پردازیم.

۱-۷-۲-۱۵-۱۲ نحوه تعریف مختصات مقادیر مربوط به مختصات

در ساختارهای بررسی شده اشاره کردیم که معمولا مقادیر مربوط به انحراف مختصات یک ناحیه، $\Delta x, \Delta y, \Delta h, \Delta w$، به عنوان خروجی ساختار تعریف شده و با اعمال آن به مختصات اولیه پیشنهادی، مختصات ناحیه نهائی محاسبه و با مقادیر مطلوب در تابع هزینه مقایسه می شود. در برخی ساختارهای نیز که نواحی اولیه پیشنهادی تعریف نمی شود، مانند نسخه اول ساختار YOLO، مقادیر مربوط به مختصات، x, y, h, w، نواحی مستقیما در خروجی ساختار محاسبه می شوند. چالش مربوط به هر دو حالت این است که مقادیر خروجی و پارامترهای مربوط به یک ساختار عمیق همواره نرمال سازی شده و معمولا در بازه اعداد بین صفر تا یک و یا منفی یک تا یک قرار می گیرند در حالی که مقادیر مربوط به مختصات هر ناحیه که معادل چهار مقدار x, y, w, h یا $\Delta x, \Delta y, \Delta h, \Delta w$ است، بسته به اندازه و محل جسم در در بازه اعداد گسترده ای قرار می گیرند که ممکن است مقیاس آن با بازه مقادیر نرمال خروجی ساختار متفاوت باشد. در این صورت اگر مستقیما مقادیر x, y, w, h یا $\Delta x, \Delta y, \Delta h, \Delta w$ هر ناحیه را به عنوان مقدار مطلوب بعد متناظر آن در تانسور خروجی در نظر بگیریم ممکن است در روند آموزش ساختار با توجه اختلاف مقیاس مقادیر خروجی و مطلوب، مقدار خطای حاصل بزرگ بوده و متعاقبا گرادیان های آموزشی که بر اساس خطا تعریف می شوند نیز مقادیر بزرگی داشته باشند که در نهایت سبب رشد بیش از حد وزن ها به ویژه در لایه های نزدیک به خروجی می شود. برای رفع این مشکل مقادیر x, y, w, h یا $\Delta x, \Delta y, \Delta h, \Delta w$ معمولا به صورت مستقیم به عنوان مقدار مطلوب تعریف نمی شوند، در عوض یک پارامتر واسط t_i که در بازه عددی برابر با بازه پارامترهای نرمال سازی شده ساختار قرار دارد، به ازای هر یک از چهار پارامتر مربوط به مختصات تعریف شده و به عنوان خروجی ساختار در نظر گرفته می شود، سپس پارامترهای مربوط به مختصات با استفاده از این پارامتر واسط مطابق روابطی مانند رابطه ۹۷-۱۲ محاسبه می شوند. این روابط می توانند به صورت خطی یا غیرخطی تعریف شوند. رابطه ۹۷-۱۲ نحوه محاسبه پارامتر x با استفاده از پارامتر واسط t_x است که در آن x^*, w^* مقادیر مربوط به مختصات جعبه لنگر اولیه است.. از این رابطه در ساختار Faster R-CNN استفاده شده است.

$$\hat{x} = w^* t_x + x^* \tag{۹۷-۱۲}$$

همچنین در مواردی نیز مقادیر مطلوب برای خروجی یک ساختار به صورت نسبتی از مقادیر x, y, w, h یا $\Delta x, \Delta y, \Delta h, \Delta w$ از ابعاد تصویر ورودی تعریف می شوند که در این صورت

همواره مقادیر مطلوب مقداری در بازه صفر تا یک خواهند داشت. در برخی ساختارها با توجه به تعاریف آن ساختار از پارامترهای دیگری نیز بدین منظور استفاده می شود برای مثال در ساختار YOLO که نواحی به هر یک از بخش های تصویر ورودی نسبت داده می شوند، موقعیت هر بخش از تصویر نیز در محاسبات دخیل بوده و مختصات هر ناحیه به صورت نسبی، نسبت به مختصات بخش مربوط به خود تعریف می شود.

۲-۷-۲-۱۵-۱۲ محو غیر حداکثری[1]

در همه ساختارهایی که در این بخش بررسی کردیم، تعداد نواحی که به عنوان خروجی ساختار تعریف می شوند بسیار بیشتر از تعداد نواحی مطلوبی است که در مجموعه دادگان آموزشی تعریف شده است؛ مانند تصویر بالای وسط شکل ۳۸-۱۲ که شامل تعداد زیادی نواحی به عنوان خروجی ساختار است در حالی که نواحی مطلوب مطابق تصویر سمت راست شکل ۳۸-۱۲ است. بنابراین توضیحات باید پس از اجرای ساختار و محاسبه خروجی آن روشی را تعریف کنیم که این نواحی زائد را حذف کند، روش محو غیر حداکثری یکی از روش هایی است که بدین منظور تعریف می شود. این روش مطابق ذیل تعریف می شود، این روش را برای حالتی که فقط یک کلاس در مجموعه دادگان تعریف شده باشد بررسی می کرده و می توانیم برای تعداد کلاس بالاتر تعمیم دهیم:

۱- ابتدا یک مجموعه، B، شامل همه نواحی به همواره مختصات و احتمال درستی، نسبت به تعریف ساختار احتمال وجود جسم و یا مقدار ضریب اطمینان ناحیه، متناظر با هر کدام از آن ها تعریف می کنیم. یک مقدار حدآستانه نیز به عنوان پارامتر کنترلی معیار IOU تعریف می کنیم.

۲- یک مجموعه تهی، D، تعریف می کنیم، نواحی که به عنوان نواحی نهائی انتخاب می شوند در ادامه به این مجموعه اضافه خواهند شد.

۳- از مجموعه B ناحیه دارای بیشترین احتمال درستی حذف و به مجموعه D افزوده می شود.

۴- مقدار پارامتر IOU ناحیه اضافه شده به مجموعه D با همه نواحی که در مجموعه B باقی مانده اند مقایسه شده و در صورتی که مقدار IOU هر یک از نواحی مجموعه B از حد آستانه تعریف شده بیشتر باشد آن ناحیه را به عنوانی یک ناحیه زائد تشخیص داده و آن را از مجموعه B حذف می کنیم.

[1] Non-maximal suppression (NMS)

۵- سپس از نواحی باقی مانده در مجموعه B یک ناحیه دیگر با بیشترین احتمال درستی انتخاب کرده و آن را از مجموعه B حذف و به مجموعه D اضافه می کنیم. سپس مرحله ۴ را برای آن اجرا می کنیم.

۶- این روند تا زمانی ادامه پیدا می کند که مجموعه B تهی شود. در این صورت نواحی موجود در مجموعه D، نواحی نهائی خواهند بود.

در یک تصویر ممکن است چند جسم مربوط به یک کلاس به گونه ای در کنار هم قرار گرفته باشند که نواحی محاط به هر یک از آن ها با هم، هم پوشانی داشته باشند. در این صورت در روند اجرای روش محو غیرحداکثری ممکن است به دلیل این هم پوشانی مقدار IOU دو ناحیه بیشتر از حد آستانه بوده و ناحیه ای که در مجموعه B قرار دارد حذف شود در حالی که این ناحیه حذف شده با توجه به جسم مربوط به آن دارای احتمال درستی بسیار بالاتری نسبت به نواحی است که مقدار IOU آن ها کمتر از حد آستانه تعریف شده است. در این صورت این ناحیه به اشتباه به عنوان ناحیه زائد تشخیص داده شده است. برای رفع این معضل یک روش دیگر با عنوان حذف غیر حداکثری هموار[1] تعریف می شود.

تفاوت روش حذف غیر حداکثری هموار با روش حذف غیر حداکثری که نحوه اجرای آن را بررسی کردیم در مرحله ۴ است. در روش حذف غیر حداکثری هموار نواحی که مقدار IOU آن ها بیشتر از حد آستانه باشد از مجموعه B حذف نمی شوند بلکه احتمال درستی آن ها مطابق رابطه ۹۸-۱۲ تغییر می کند. در این صورت مقدار احتمال درستی ناحیه نسبت به مقدار آن IOU کاهش پیدا کرده و احتمال انتخاب آن در مراحل بعدی به عنوان ناحیه ای با بیشترین احتمال درستی که وارد مجموعه D می شود کاهش می یابد. این روش با توجه به روند کند حذف نواحی از مجموعه B نسبت به روش محو حداکثری سرعت اجرای کمتری دارد. در رابطه ۹۸-۱۲، s_i احتمال درستی یک ناحیه و t مقدار حد آستانه تعریف شده برای IOU است.

$$s_i := \begin{cases} s_i, & if \quad IOU < t \\ s_i(1-IOU), & if \quad IOU \geq t \end{cases} \tag{۹۸-۱۲}$$

۱۲.۱۵.۳ ساختارهای پیچشی با خروجی تانسور (تصویر)

در ساختارهای پیچشی که تا کنون بررسی کردیم، خروجی ساختار به صورت یک بردار مقادیر مطلوب تعریف می شد، حتی خروجی ساختارهایی مانند YOLO که به صورت تانسور تعریف می شد در واقع تغییر شکلی از یک بردار بود. با گسترش روش های مبتنی بر یادگیری عمق و پیشرفت ساختارهای پیچشی کاربردهای دیگری برای این ساختارهای تعریف شد، استفاده از ساختارهای

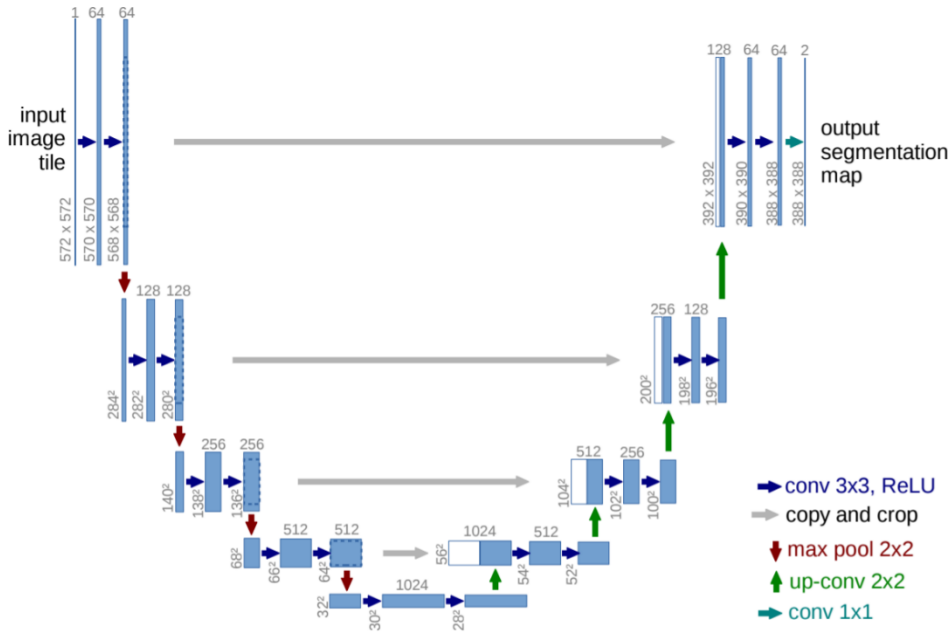
[1] Soft NMS

پیچشی برای افزایش وضوح تصاویر[1]، ترمیم تصاویر[2]، تولید تصویر رنگی از تصاویر سیاه و سفید و قطعه بندی معنائی[3] تصاویر به صورت طبقه بندی هر یک از پیکسل های تصویر از جمله این کاربرد ها است. در همه کاربردهای ذکر شده خروجی یک تانسور، تصویر، است، بنابراین ساختارهای پیچشی مختص این موارد باید به گونه ای تعریف شوند که خروجی آن ها به صورت یک تانسور باشد. در ادامه به معرفی چند مورد از این ساختارها می پردازیم، ضرب پیچشی ترانهاده از مهم ترین عملیاتی است که در این نوع ساختارها به طور گسترده به کار برده می شود.

۱۲.۱۵.۳.۱ شبکه UNet

این شبکه در سال ۲۰۱۵ میلادی در معرفی شده است. ساختار این شبکه در ابتدا برای قطعه بندی معنائی تصاویر بایومدیکال معرفی شده است ولی با توجه به انعطاف پذیری مناسب آن می توان با تعریف خروجی های مختلف برای کاربردهایی مانند افزایش وضوح، ترمیم تصویر و تبدیل تصویر سیاه سفید به رنگی نیز از آن استفاده کرد. ساختار اولیه این شبکه مطابق شکل ۴۳-۱۲ است. این شبکه مطابق شکل ۴۳-۱۲ از لایه های پیچشی و لایه های پیچشی ترانهاده که با عنوان Up-Convolution در این ساختار معرفی شده است، تشکیل شده و بین خروجی لایه های پیچشی و ورودی لایه های پیچشی ترانهادی که ابعاد تانسورهای آن ها در راستای طول و عرض برابر است اتصالات جهشی که دو تانسور را کنار هم قرار می دهند[4] تعریف شده است.

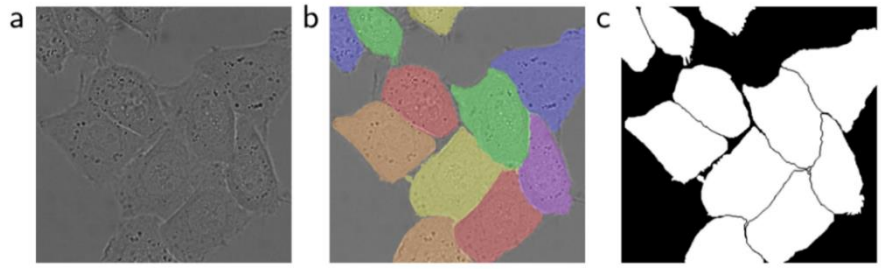
[1] Super resolution
[2] Inpainting
[3] Semantic segmentation
[4] Concatenation



شکل ۴۳-۱۲: ساختار شبکه UNet [۱۰۹]

شـــکل ۴۴-۱۲ ورودی، a، خروجی، c که به صـــورت یک کانال تعریف شـــده اســـت، و انطباق خروجی بر روی ورودی، b، این شـــبکه UNet را نشـــان می دهد. برای وظیفه قطعه بندی معنائی تصویر ورودی این شبکه یک تصویر رنگی با سه کانال رنگی RGB یا سیاه سفید با یک کانال بوده و خروجی و مقادیر مطلوب آن آن یک ماسـک، تصـویر، به صـورت یک تانسـور سـه بعدی تعریف می شـــود که اندازه عمق، تعداد کانال ها، آن به تعداد کلاس های تعریف شـــده برای مجموعه دادگان اســت، قطعات، مجموعه پیکســل ها، مربوط به جســم هر کلاس در کانال متناظر با آن در ماســک مقادیر مطلوب دارای عدد مقدار یک و بقیه پیکسـل ها دارای مقدار صـفر اسـت؛ به عبارت دیگر در هر کانال از ماسک مقادیر مطلوب دو کلاس پیکسل تعریف می شود، پیکسل های متعلق به کلاس متناظر با آن کانال، مقادیر یک، و پیکسـل هایی که متعلق به کلاس خاصـی نیسـتند، پیکسـل های پس زمینه با مقادیر صـــفر، بدین ترتیب ایده مربوط به بردارهای One hot که برای طبقه بندی استفاده می شود، برای تصاویر، ماسک، تعمیم داده می شود. برای وظیفه قطعه بندی معنایی تصویر معمولا اندازه تانسور خروجی و ورودی در راستای طول و عرض برابر تعریف می شود تا بتوان مانند شـکل ۴۴-۱۲ خروجی و ورودی را با هم منطبق کرده و محل پیکسـل های مربوط به هر جسـم را مشخص کنیم، در صورتی که این ابعاد با هم برابر نباشند پس از اجرای ساختار و محاسبه خروجی، به یکی از تانسورهای ورودی یا خروجی تغییر شکل اعمال شده تا ابعاد آن با تانسور دیگر برابر شود. در وظایفی مانند ترمیم تصـاویر، تبدیل تصـاویر سـیاه و سـفید به رنگی و یا بهبود وضـوح تصـویر، تانسور خروجی و مطلوی به صورت یک تانسور سه بعدی با عمق سه، سه کانال، تعریف می شود که

مقادیر درایه ها هر کانال از تانسور مقادیر مطلوب، همان مقادیر پیکسل های تصویر مطلوب متناظر با تصویر ورودی است.



شکل ۴۴-۱۲: تصویر ورودی، انطباق ورودی و خروجی در شبکه UNet [۱۰۹]

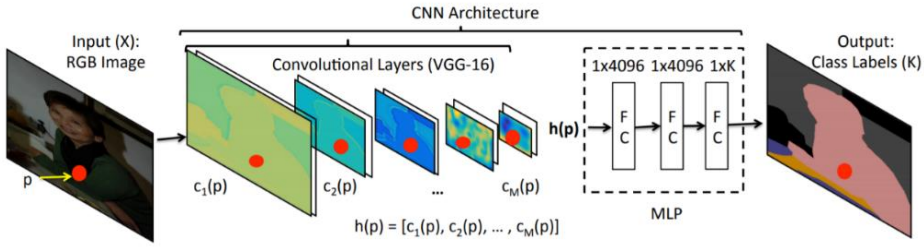
در قطعه بندی معنائی ممکن است بعضی از تصاویر ورودی مانند تصاویر شکل ۱۲-۴۴ شامل چند نمونه[1] از یک کلاس باشند، در این صورت اگر مجموعه پیکسل های مربوط به هر نمونه را از کل پیکسل های مربوط به آن کلاس تفکیک کنیم به آن قطعه بندی نمونه[2] اطلاق می شود. قطعه بندی نمونه در این ساختار پس از تشکیل تانسور ماسک به صورت دستی با استفاده از روش های پردازش تصویر مانند اعمال اتساع[3] و فرسایش[4]، تشخیص بدنه محدب[5] و... استفاده می شود می شود که دقت قطعه بندی نمونه ها در این روش وابسته به نحوه قرارگیری نمونه ها در کنار هم و پارامترهای روش های استفاده شده بستگی داشته و در مواردی عملکرد چندان مناسبی ندارد.

۱۲.۱۵.۳.۲ شبکه Pixel-Net

شبکه Pixel-Net در سال ۲۰۱۶ میلادی معرفی شده است. از شبکه Pixel-Net نیز برای وظایفی که برای شبکه UNet تعریف می شود استفاده می شود. با توجه به شکل ۱۲-۳۴ شبکه UNet دارای لایه های پیچشی و پیچشی ترانهاده است که اجرای آن مستلزم حجم محاسبات بالائی نسبت به ساختاری مانند شبکه های پرسپترون چند لایه است. از این رو شبکه Pixel-Net برای مرتفع کردن مشکل مربوط به حجم محاسبات شبکه UNet معرفی شده است. ساختار این شبکه مطابق شکل ۱۲-۴۵ ابتدا از یک ساختار پیچشی VGG-16 تشکیل شده است. طبق مطالبی که پیشتر بررسی کردیم محل متناظر با هر یک از پیکسل های تصویر ورودی با را اعمال مقیاس مربوط هر لایه ساختار می توان در آن لایه مشخص کرد. روند اجرای این شبکه بدین صورت است که به ازای تصویر ورودی مطابق شکل ۱۲-۴۵ موقعیت هر یک از پیکسل های تصویر ورودی در

[1] Instance
[2] Instance segmentation
[3] Dilation
[4] Erosion
[5] Convex hull

همه لایه های پیچشی مشخص می شود، از آنجایی که ممکن است این موقعیت مشخص شده در لایه های مختلف بسته به مقیاس لایه موقعیت دقیق یک درایه از تانسور نباشد (موقعیت بین چند درایه از تانسور در راستای طول و عرض آن تعریف شود)، مقدار متناظر با محل متناظر با پیکسل ورودی در هر لایه برابر با درون یابی دو خطی[1] از چهار درایه نزدیک به آن در راستای طول و عرض، و همه عمق های تانسور، در نظر گرفته می شود. بدین ترتیب برای هر پیکسل ورودی در هر لایه پیچشی یک بردار با اندازه برابر با عمق تانسور خروجی آن لایه تعریف می شود، مجموعه این بردارها به ازای همه لایه ها به مطابق رابطه ۹۹-۱۲ تعریف می شود.



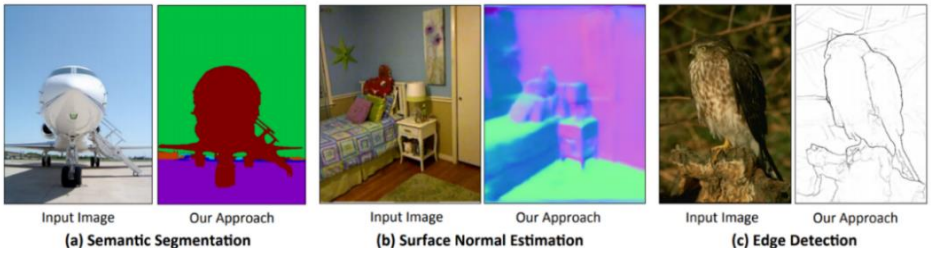
شکل ۴۵-۱۲: ساختار شبکه Pixel-Net [۱۱۰]

$$h_p(\mathbf{X}) = [c_{1(p)}, c_{2(p)}, ..., c_{n(p)}] \qquad (۹۹-۱۲)$$

در رابطه ۹۹-۱۲، $\boldsymbol{X}$ تصویر ورودی و $c_{i(p)}$ بردار متناظر با پیکسل p در لایه i است. بردار های مجموعه رابطه ۹۹-۱۲ در جهت طول کنار هم قرار گرفته و تشکیل یک بردار بزرگ تر را می دهند که به عنوان ورودی یک شبکه پرسپترون چند لایه تعریف می شود، لایه خروجی این شبکه که خروجی نهائی کل ساختار است برداری با اندازه K که همان تعداد کلاس های تعریف شده به اضافه یک کلاس برای کلاس عدم تعلق به کلاس های تعریف شده، کلاس پس زمنیه[2]، برای نمونه های موجود در مجموعه تصاویر آموزشی است. با این توصیف بردار مقدار مطلوب نیز به صورت یک بردار One Hot با اندازه K تعریف می شود. این بردار خروجی کلاس هر پیکسل موجود در تصویر ورودی را که برداری متناظر با بردار رابطه دارد مشخص می کند، با تکرار روند تولید بردار رابطه ۹۹-۱۲ برای همه پیکسل های ورودی، ماسک کلی تصویر ورودی تشکیل شده و بدین ترتیب روند قطعه بندی معنائی تصویر کامل می شود. در این ساختار علاوه بر کاهش نسبی حجم محاسبات نسبت به شبکه UNet، با توجه به عدم تعریف لایه های پیچشی ترانهاده که مقادیر درایه های خروجی با استفاده از کرنل های پیچشی، مقادیر مربوط به هر پیکسل جداگانه به طور مستقل محاسبه می شود و از این جهت ساختار دارای دقت بیشتری در نسبت به شبکه UNet است، علاوه بر این با

[1] Bi-linear interpolation
[2] Back ground

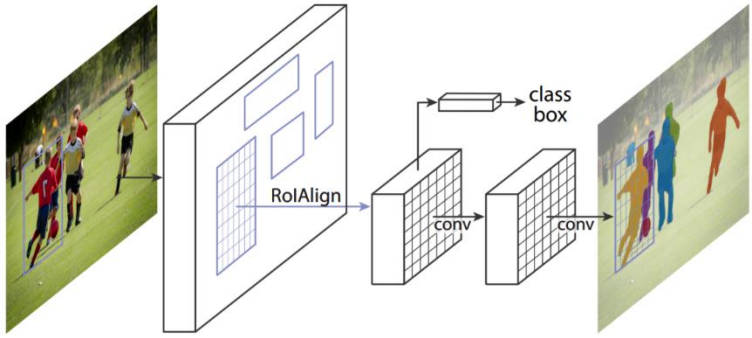
توجه به ویژگی می توان وظایف دیگری نیز مانند تشخیص لبه اجسام برای این ساختار تعریف کرد. در صورت استفاده از ساختار برای وظایفی مانند ترمیم تصویر، شبکه پرسپترون به صورت یک مدل رگرسور با خروجی به صورت یک بردار با اندازه شده و به ازای هر پیکسل ورودی مقدار یک پیکسل از خروجی را محاسبه می کند. شکل ۴۶-۱۲ شامل تصویر ورودی و خروجی متناظر آن برای سه نمونه وظیفه تعریف شده برای این ساختار شامل قطعه بندی معنائی، تشخیص سطوح، راستای سطوح، و تشخیص لبه تصویر است.



شکل ۴۶-۱۲: تصاویر ورودی و خروجی متناظر برای شبکه Pixel-Net [۱۱۰]

۱۲.۱۵.۳.۳ شبکه Mask R-CNN

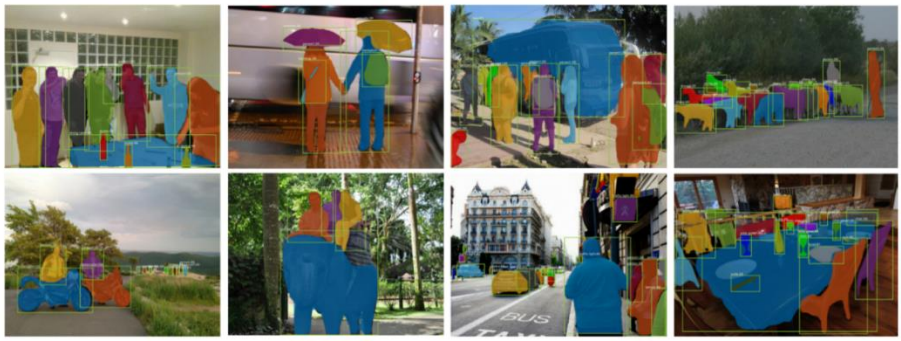
در معرفی ساختار UNet اشاره کردیم که برای قطعه بندی نمونه های مختلف یک کلاس در تصویر پس از محاسبه خروجی و تشکیل ماسک متناظر با ورودی از روش های کلاسیک پردازش تصویر استفاده می کنیم تا مجموعه پیکسل های هر نمونه را از کل پیکسل های مربوط به یک کلاس جدا کنیم. در این روند ساختار عمیق شبکه UNet نقشی نداشته و از این نظر عملکرد شبکه در این وظیفه دارای محدودیت هایی است. برای رفع این معضل و ارائه راهکاری مبتنی بر یادگیری عمیق برای قطعه بندی نمونه شبکه Mask R-CNN معرفی شده است. در این شبکه دو وظیفه تشخیص موقعیت و قطعه بندی معنائی به صورت سری تعریف شده است. این شبکه در سال ۲۰۱۸ میلادی معرفی شده است در واقع یک نسخه از شبکه Faster R-CNN است که در آن علاوه بر کلاس هر ناحیه، ماسک متناظر با آن ناحیه پیشنهادی نیز در خروجی ساختار محاسبه می شود. شمای کلی این شبکه مطابق شکل ۴۷-۱۲ است. ساختار پیچشی ابتدای برای استخراج ویژگی، ورودی شبکه پیشنهاد ناحیه، استفاده می شود به صورت هرم ویژگی تعریف شده است.



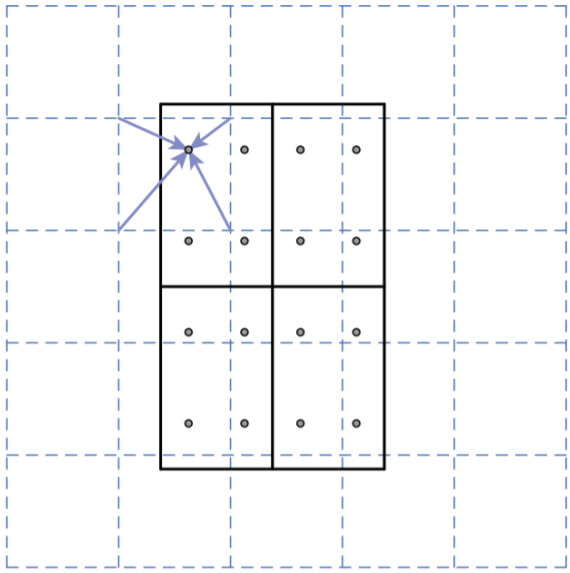
شکل ۴۷-۱۲: شمای ساختار شبکه Mask R-CNN [۱۰۲]

در ساختار Faster R-CNN اشاره کردیم که پس از محاسبه نواحی پیشنهادی به بخش های مربوط به این نواحی در تانسور خروجی بخش پیچشی یک لایه تجمیع ناحیه اعمال می شود، در صورت استفاده از لایه تجمیع ناحیه ممکن است با توجه به ابعاد و مختصات نواحی، پنجره هایی که مقادیر حداکثری از آن ها استخراج می شود به طور تقریبی، مانند شکل ۳۳-۱۲، تعریف شوند که این موضوع در وظیفه ای مانند قطعه بندی معنایی که مستلزم ویژگی های دقیق از تصویر است ممکن است باعث افت عملکرد ساختار شود. برای رفع مشکل مربوط به نواحی پنجره های تقریبی در لایه تجمیع ناحیه رویکرد دیگری با عنوان لایه تراز ناحیه[1] تعریف می شود. در این روش موقعیت ناحیه تقریب زده نشده و موقعیت محاسبه شده نسبت به مقیاس ساختار پیچشی عینا در تانسور اعمال می شود، سپس این ناحیه به پنجره های کوچکتر تقسیم می شود، برای استخراج مقدار حداکثری از هر پنجره مطابق شکل ۴۹-۱۲، چهار نقطه در هر پنجره تعریف می شود، هر کدام از این نقاط یک مقدار اسکالر دارد که از طریق درون یابی دو خطی نسبت به چهار درلیه نزدیک آن محاسبه می شود. پس از آن مقدار حداکثری هر پنجره از میان مقادیر این چهار نقطه انتخاب می شود. این مقادیر حداکثری برای تشخیص کلاس جسم داخل ناحیه و انحراف موقعیت و همچنین تشکیل ماسک مربوط به آن از طریق لایه های پیچشی ترانهاده استفاده می شود. شکل ۴۸-۱۲ تعدادی از خروجی های این شبکه را که بر روی تصویر ورودی منطبق شده اند نشان می دهد.

[1] ROI Align



شکل ۴۸-۱۲: خروجی های منطبق به شده به ورودی در ساختار Mask R-CNN [۱۰۲]



شکل ۴۹-۱۲: لایه تراز ناحیه [۱۰۲]

۱۲.۱۵.۴ ساختارهای پیچشی-بازگشتی

یکی از کاربردهای مهم پردازش تصویر مبتنی بر یادگیری عمیق استفاده از آن ها در نویسه خوان های نوری[1] برای تشخیص و استخراج کاراکترها و حروف موجود در تصویر یک نوشته است. در توسعه یک نویسه خوان نوری اگر بخواهیم از بین ساختارهایی که تا کنون معرفی کردیم انتخاب کنیم مناسب ترین ساختار، یکی از ساختارهای پیچشی با خروجی متغیر خواهد بود، در این صورت برای هر یک از حروف و کاراکترهای داخل متن تصویر یک کلاس تعریف می کنیم، ساختار موقعیت و کلاس مربوط به هر کاراکتر را تشخیص داده و با کنار هم قرار دادن آن ها با توجه به موقعیت آن ها که در خروجی ساختار متن داخل تصویر استخراج خواهد شد.

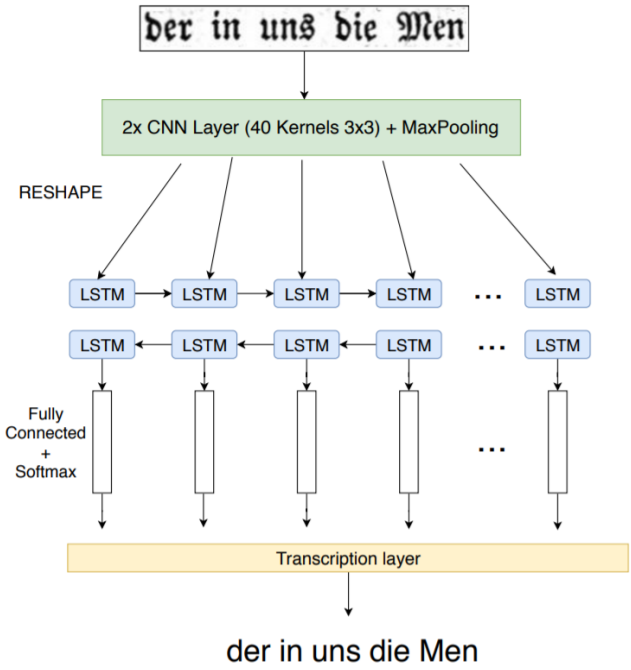
[1] Optical character recognition (OCR)

رویکرد فوق الذکر مبتنی بر ساختار پیچشی با خروجی متغیر یک راهکار مبتدی بوده و استفاده از آن چالش های بسیاری به همراه خواهد داشت. یکی از این چالش ها مربوط به تهیه مجموعه دادگان آموزشی مناسب برای آموزش ساختار است؛ با توجه به گستره وسیع فونت ها و ابعاد کاراکترها در نوشته های مختلف تهیه یک مجموعه دادگان آموزشی به طوری که به ازای هر کاراکتر یک کلاس مجزا تعریف شده و هر کلاس شامل دامنه وسیعی از تصاویر مربوط به آن کاراکتر باشد بسیار دشوار است و در بسیاری از موارد مجموعه دادگان فقط برای گستره محدودی از فونت ها و ابعاد مربوط به کاراکترها تهیه می شود، این عامل سبب می شود که مدل آموزش داده شده با استفاده از این دادگان یک مدل عمومی نبوده و فقط قادر به تشخیص فونت های موجود در مجموعه دادگان باشد. چالش دیگری که در این رویکرد مطرح می شود تعریف تعداد زیاد کلاس برای آموزش مدل با توجه به تعداد بالای کاراکترهای موجود در هر زبان است که ممکن است باعث افت عملکرد ساختار پیچشی شود. چالش بعدی مربوط به نحوه استفاده از این مدل در دنیای واقعی است؛ با توجه به موقعیت نسبی دوربین و محل متن ممکن است تصویر حاوی متن دارای درجه ای از پرسپکتیو[1] باشد، در این صورت حتی اگر فونت و اندازه کاراکترهای متن در تصاویر آموزشی تعریف شده باشد مدل قادر به تشخیص کاراکتر نخواهد بود. اما در کنار این موارد می دانیم که یک کاراکتر همواره الگوی ثابتی داشته و بخش های مختلف آن از این جهت به هم وابسته هستند، برای مثال کاراکتر A را در نظر بگیرید فرض کنیم تصویر حاوی این کاراکتر را حول محور تقارن کاراکتر به دو بخش تقسیم کنیم در این صورت الگوی کلی هر یک از این دو بخش همواره فارغ از فونت و اندازه کاراکتر قرینه بخش دیگر بوده و بدین ترتیب الگوی این دو بخش به هم وابسته است. این وابستگی منحصر به کاراکتر های متقارن مانند A نبوده و برای همه حروف و کاراکترها روند مشابهی تعریف می شود، از این رو همواره می توان با کنار هم قرار دادن الگوی کلی بخش های یک کاراکتر و مدل سازی وابستگی بین آن ها، کلاس، آن کاراکتر را تشخیص داد.

ایده تعریف وابستگی بین بخش های یک کاراکتر باعث شد تا برای رفع معضلات و چالش های ذکر شده برای این وظیفه به جای استفاده از مدل های پیچشی با خروجی متغیر از ساختار های پیچشی-بازگشتی استفاده شود. همان طور که در فصل بررسی کردیم مدل های بازگشتی و حافظه دار می توانند در مواردی که هر نمونه های ورودی به هم وابستگی دارند عملکرد مناسبی داشته باشند. بدین ترتیب با ترکیب آن ها با ساختارهای پیچشی می توان این یک مدل مناسب تعریف کرد که ورودی آن به بخش های مختلف یک متن است. ساختار کلی این شبکه ها مطابق شکل ۱۲-۵۰ است. برخلاف سری های زمانی که در آن ها هر نمونه فقط به نمونه های پیش از خود وابسته است در اینجا الگوی یک کاراکتر به بخش های قبل و بعد از خود وابسته است، از این رو در این ساختارها معمولا از شبکه بازگشتی LSTM دو جهته[2] استفاده می شود. وابستگی اشاره شده

[1] Perspective
[2] Bi-directional LSTM

محدود به الگوی بین بخش های یک کاراکتر محدود نبوده و چیدمان کاراکترهای مختلف در کنار هم که باعث ایجاد کلمات و جملات در یک متن می شود نیز در تعریف و مدل سازی این ساختارها حائز اهمیت است، به همین دلیل بخش بازگشتی این ساختارها معمولا مشابه مدل های زبانی که در فصل معرفی شده است تعریف می شود.



شکل ۵۰-۱۲: ساختار شبکه پیچشی-بازگشتی تعریف شده برای وظیفه نویسه خوان نوری [۱۱۵]

با توجه به توضیحات فوق روند پیشرو در این ساختارهای بدین شرح است:

۱. ابتدا با استفاده از روش های کلاسیکی مانند تشخیص لبه و یا با استفاده از یک مدل آموزش داده شده قسمت های حاوی متن از تصویر استخراج می شوند.

۲. قسمت استخراج شده در مرحله ۱ در راستای افقی به بخش های کوچک مساوی تقسیم می شود.

۳. به هر یک از بخش های مرحله ۲ یک ساختار پیچشی شامل تعدادی لایه پیچشی و تجمیع اعمال شده و در نهایت به ازای هر بخش یک تانسور نگاشت های ویژگی تشکیل می شود.

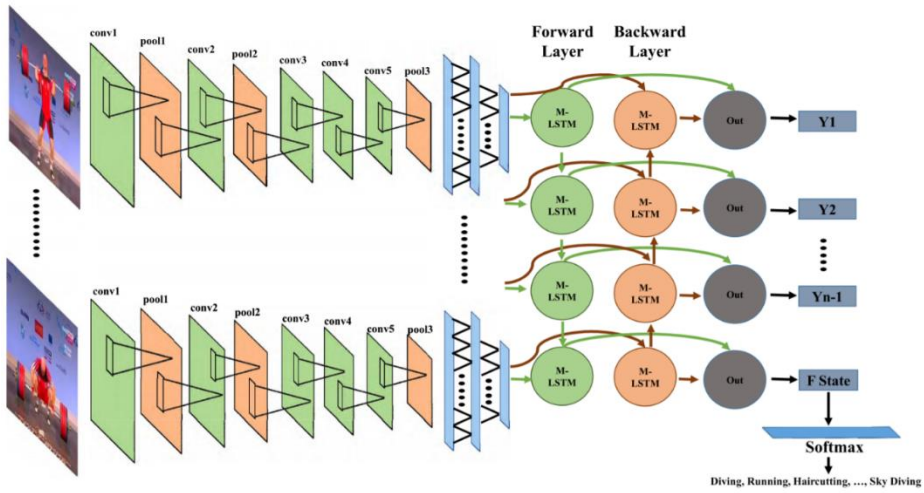
۴. تانسورهای مرحله ۴ به صورت یک سری وارد ساختار پیچشی شده و سپس خروجی نهائی ساختار که خروجی بخش بازگشتی است محاسبه می شود.

۵. پس از تشکیل خروجی ساختار با توجه به ماهیت خروجی، یک سری عملیات پس پردازش مانند تنظیم فاصله بین کاراکترها و... بر روی خروجی مرحله ۴ در بخش Transcription layer اعمال می شود.

مقادیر مطلوب برای ساختارهایی مطابق ساختار شکل ۵۰-۱۲ متن متناظر با متن داخل تصویر است. با توجه به استفاده از مدل های زبانی با تعریف مقادیر مطلوب مناسب از این ساختار می توان برای وظایفی مانند ترجمه متن یک تصویر نیز استفاده کرد. روند آموزش این ساختارها ممکن است شامل هر دو بخش پیچشی و بازگشتی باشد و یا فقط محدود به بخش بازگشتی بوده و پارامترهای بخش پیچشی به صورت از پیش آموزش داده شده تعریف شوند.

از ساختارهای پیچشی-بازگشتی می توان در پردازش توالی تصاویر، ویدئو، نیز استفاده کرد. پردازش ویدئو در وظایفی مانند ردیابی اجسام با استفاده از ساختارهایی مانند YOLO با تفکیک فریم های ویدئو به تصاویر مجزا انجام می شود، اما در برخی وظایف مانند طبقه بندی کلی حرکات[1]، تشخیص احساسات[2] با حالت چهره و... اطلاعات یک فریم حاوی اطلاعات مناسبی نبوده و با پردازش فریم های مختلف در کنار عملکرد مدل ارتقا پیدا می کند. در این موارد ساختارهای پیچشی-بازگشتی برای پردازش ویدئو به کار برده می شوند. روند کلی این ساختارها مشابه موارد ذکر شده برای ساختار پیچشی-بازگشتی است که برای نویسه خوان نوری مطابق شکل ۵۰-۱۲ تعریف می شود با این تفاوت که در ساختاری که برای پردازش ویدئو استفاده می شود لایه های پیچشی هر بار به یک فریم، تصویر، از ویدئو اعمال می شوند. شکل ۵۱-۱۲ ساختار کلی یک شبکه پیچشی-بازگشتی که برای پردازش ویدئو استفاده می شود را نشان می دهد. ساختار شکل ۵۱-۱۲ برای وظیفه طبقه بندی حرکات ویدئو یک وزنه بردار تعریف شده است.

[1] Action classification
[2] Emotion detection with facial expression



شکل ۵۱-۱۲: ساختار پیچشی-بازگشتی با ورودی ویدئو [۱۵۳]

۱۲.۱۵.۵ مکانیزم توجه[1] در ساختارهای پیچشی

از مکانیزم توجه که در ابتدا برای مدل های زبانی ارائه شده است می توان در ساختارهای پیچشی نیز استفاده کرد. روند اجرای این مکانیزم در مدل های زبانی در فصل ۱۴ بررسی شده است. این مکانیزم به دو صورت کلی در ساختارهای پیچشی تعریف می شود؛ حالت اول روش مبتنی بر کانال[2] که برای وزن دهی به کانال های، عمق ها، یک تانسور و حالت دوم روش مبتنی بر مکان[3] که برای ایجاد تمرکز و وزن دهی به بخشی از تانسور در فضای تعریف شده در راستای طول و عرض آن به کار برده می شود. هر دو مکانیزم توجه مبتنی بر کانال و مکان با توجه به این که از دارای ورودی مشترک با بخشی که به آن مکانیزم اعمال می شود دارند مکانیزم توجه خود محور[4] محسوب می شوند، تفاوت مکانیزم توجه و مکانیزم توجه خود محور در فصل بررسی شده است. در ادامه به معرفی مهم ترین ساختارهای پیچشی که این مکانیزم در آن ها استفاده شده است می پردازیم. با تعریف یک ساختار پیچشی دارای چند خروجی می توانیم برای این ساختارها مکانیزم توجه چندگانه[5] نیز تعریف کنیم که در ادامه به بررسی آن نیز خواهیم پرداخت.

۱۲.۱۵.۵.۱ شبکه های SE-Net[6]

در شبکه های SE-Net از رویکرد مکانیزم توجه مبتنی بر کانال استفاده می شود. این شبکه ها با اضافه کردن عملیات های مربوط به مکانیزم توجه به ساختارهای مرسوم تعریف می شوند. این

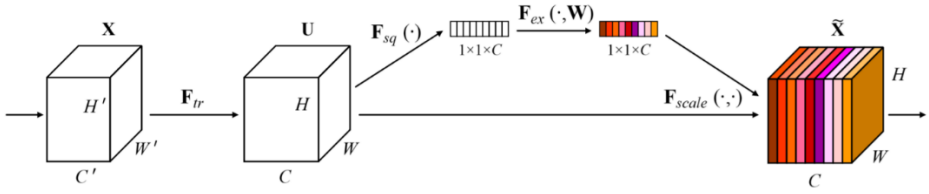
[1] Attention mechanism (AT)
[2] Channel-wise attention mechanism
[3] Position-wise attention mechanism
[4] Self-attention mechanism (SA)
[5] Muti-head attention mechanism
[6] Squeeze-and-Excitation Networks

ساختار در سال ۲۰۱۸ میلادی معرفی شده است. عملیات تعریف شده در این شبکه ها اگرچه اولین ساختاری نیست که در آن مکانیزم توجه مبتنی بر کانال تعریف شده است ولی عملکرد آن باعث شده است که ساختار بلوک های آن ها به عنوان تعریفی کلی از ساختار مکانیزم توجه مبتنی بر کانال در شبکه های پیچشی شناخته شود، از این جهت بررسی آن این ساختار حائز اهمیت است. ساختار کلی عملیات مکانیزم توجه تعریف شده در این شبکه مطابق شکل ۱۲-۵۲ است. این عملیات را می توان در بلوک سایر ساختارهای پیچشی با تعریف آن به صورت سری با عملیات های بلوک تعریف کرد، برای مثال شکل ۵۳-۱۲ ادغام آن در بلوک شبکه ResNet و Inception و مقایسه آن ها با بلوک های اولیه را نمایش می دهد. عملیات های تعریف شده شامل سه عملیات به شرح زیر است که به صورت سری در کنار هم اجرا می شوند:

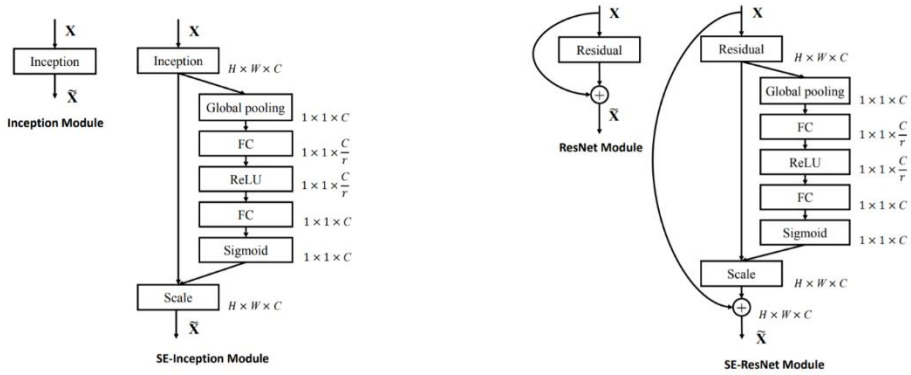
۱. **Squeeze:** این عملیات در واقع یک لایه تجمیع میانگین سراسری است که به تانسور ورودی به عملیات اعمال می شود. با اعمال این عملیات تانسور ورودی به بلوک با ابعاد $m \times n \times c$ تبدیل به یک تانسور با ابعاد $1 \times 1 \times c$ ، بردار، خواهد بود.

۲. **Excitation:** این عملیات شامل یک شبکه پرسپترون دو لایه شامل لایه پنهان و خروجی بوده و ابعاد خروجی و ورودی آن با هم برابر هستند. توابع فعال ساز لایه اول ReLU و لایه دوم سیگموئید است. تعداد نورون های لایه پنهان کمتر از بعد ورودی است، $\frac{c}{r}, r \in \mathbb{R}$.

۳. **Scale:** در این عملیات بردار خروجی حاصل از مرحله Excitation در تانسور ورودی ضرب می شود، ضرب بردار در تانسور؛ با توجه به ابعاد به ازای هر یک از کانال های تانسور ورودی، یک مقدار اسکالر درخروجی مرحله Excitation تولید می شود، این مقدار اسکالر در همه درایه های آن کانال ضرب شده و در نهایت یک تانسور با ابعاد تانسور اولیه به عنوان خروجی نهائی ایجاد می شود. تفاوت این تانسور خروجی با تانسور ورودی اولیه، اعمال وزن متناسب به مقادیر هر کانال است.



شکل ۵۲-۱۲: ساختار عملیات تعریف شده در شبکه SE-Net [۹۶]



شکل ۵۳-۱۲: مقایسه بلوک های شامل و فاقد عملیات SE-Net [۹۶]

با اعمال مکانیزم توجه مبتنی بر کانال بر روی یک تانسور نگاشت های ویژگی، با توجه به بازه سیگموئید لایه خروجی مرحله Excitation ، بازه صفر تا یک، به کانال هایی که حاوی اطلاعات بهتری هستند، با ضرب مقادیر نزدیک به یک وزن بیشتری داده شده و از این رو با اعمال تابع فعال ساز بر این کانال ها مقدار خروجی تابع به نسبت کانال هایی که وزن کمتری دارند بزرگ تر بوده و در نتیجه تاثیر آن ها در روند کلی ساختار بیشتر خواهد بود.

در عملیات های فوق الذکر، مرحله Scale فاقد پارامتر آموزشی است ولی برای آموزش پارامترهای مرحله Exictation لازم است مشتق خروجی نسبت به ورودی آن را محاسبه کنیم، این مشتق مطابق رابطه ۱۰۰-۱۲ است. در رابطه ۱۰۰-۱۲، $\boldsymbol{U}$ همان تانسور ورودی به عملیات های این مکانیزم است که در شکل ۵۲-۱۲ نیز نشان داده شده است.

$$\cdots \underbrace{\frac{\partial \mathbf{O}_{scale}}{\partial \mathbf{O}_{ex}}}_{\mathbf{U}} \frac{\partial \mathbf{O}_{ex}}{\cdots} \qquad (۱۲-۱۰۰)$$

رابطه مشتق خروجی نسبت به ورودی کل عملیات های مکانیزم نیز مطابق رابطه ۱۰۱-۱۲ تعریف می شود. طبق رابطه ۱۰۱-۱۲ این جمله شامل ضرب مشتقات خروجی نسبت به ورودی سه عملیات به اضافه یک است. این جمله یک به دلیل وجود تانسور ورودی در عملیات

$Scale$ است.

$$\cdots \quad \underbrace{\frac{\partial \mathbf{O}_{scale}}{\partial \mathbf{U}}}_{f'_{scale(.)} \cdot f'_{ex(.)} \cdot f'_{sq(.)} + 1} \quad \frac{\partial \mathbf{U}}{\cdots} \tag{۱۲-۱۰۱}$$

۱۲.۱۵.۵.۲ شبکه های دارای مکانیزم توجه مبتنی بر مکان

همان طور که در مطالب قسمت قبل اشاره کردیم، مکانیزم توجه مبتنی بر کانال به تعریف و اعمال عملیات های مربوط به صورت یک ماژولی که به ساختار اضافه می شود، اعمال می شود. رویکرد مشابهی نیز در مکانیزم توجه مبتنی بر مکان در ساختارهای پیچشی وجود دارد. همان طور که پیشتر اشاره کردیم، مکانیزم توجه مبتنی بر مکان در واقع یک مکانیزم توجه خودمحور است، که به درایه های یک تانسور در ساختار پیچشی اعمال می شود. روند عملیات های این مکانیزم مطابق مراحل ذیل است:

۱. در ابتدا به تانسور ورودی که این مکانیزم قرار است به آن اعمال شود سه لایه مسطح ساز اعمال می شود. بدین ترتیب تانسور تبدیل به سه بردار مشابه A, B, C می شود. قبل از اعمال لایه مسطح ساز با توجه به حجم محاسبات مکانیزم توجه معمولا با اعمال یک لایه پیچشی سه بعدی با کرنل های ۱×۱ عمق تانسور را کاهش دهیم.

۲. به هر یک از درایه های یکی از بردار های مرحله قبل، B، تابع رابطه ۱۲-۱۰۲ اعمال می شود. در رابطه ۱۲-۱۰۲، $\bar{x}$ میانگین مربوط به آن درایه به ازای همه دادگان دسته[1] و k یک پارامتر اسکالر کنترلی است.

$$\phi_{(x)} = \frac{1}{k}(x - \bar{x}) \tag{۱۲-۱۰۲}$$

۳. بردار حاصل از رابطه ۱۲-۱۰۲ و یکی دیگر از بردارهای مرحله ۱ مطابق رابطه ۱۲-۱۰۳ در هم ضرب، ضرب ماتریسی، می شوند، حاصل این ضرب یک ماتریس مربعی، D، خواهد بود. این ماتریس مربعی همان ماتریس کواریانس بردار حاصل از تانسور ورودی است.

$$D_{n \times n} = A_{n \times 1} \times B^T_{1 \times n} \tag{۱۲-۱۰۳}$$

۴. به درایه های ماتریس مربعی حاصل از مرحله قبل یک تابع بیشینه هموار، Soft max، مطابق رابطه ۱۲-۱۰۴ اعمال می شود. بدین ترتیب درایه های حاصل مقادیری در

[1] Batch

بازه صفر تا یک خواهند داشت این مقادیر همان وزن مقادیر وزن ها هستند. مقادیر بزرگ تر به دو دسته از درایه های نسبت داده می شوند؛ نخست به درایه هایی با مقدار واریانس بالا که روی قطر اصلی ماتریس کواریانس قرار دارند، این نوع درایه ها از آن جهت اهمیت دارند که دارای دامنه تغییرات بزرگ تری بوده و در نتیجه در تفکیک پذیری ساختار نقش بیشتری دارند بنابراین باید دارای وزن بیشتری باشند. دسته دوم درایه هایی که در خروجی تابع بیشینه هموار مقدار بزرگ تری دارند درایه هایی هستند که خارج از قطر اصلی قرار داشته و دارای وابستگی بالایی با سایر درایه ها دارند؛ از آن جایی که در یک تصویر بر خلاف ورودی های برداری که استقلال ابعاد در آن ها مورد تاکید است درایه ها، پیکسل ها، در کنار هم اطلاعات مربوط به الگوی تصویر ورودی را ارائه کرده و دارای وابستگی محلی به هم هستند، درایه هایی که وابستگی زیادی به درایه های دیگر داشته باشند به احتمال زیاد حاوی اطلاعات بخشی از الگوی تصویر هستند در حالی که درایه های دارای وابستگی کمتر اطلاعات چندان مفیدی از الگوی تصویر ارائه نمی کنند و حتی شاید یک نویز باشند.

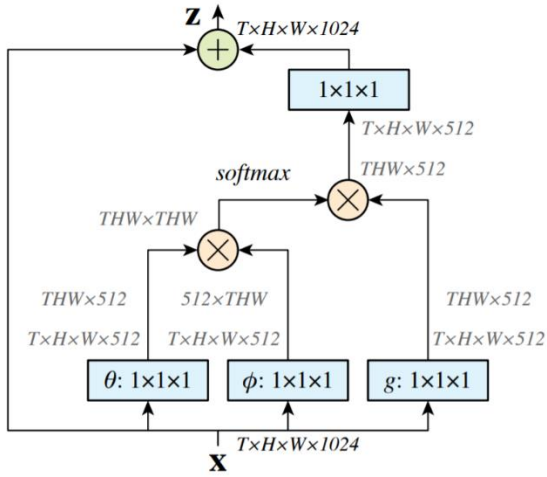
$$\text{Softmax}(d_{i,j}) = \frac{e^{d_{i,j}}}{\sum_{k=1}^{n}\sum_{l=1}^{n} e^{d_{k,l}}} \qquad (۱۲-۱۰۴)$$

۵. ماتریس حاصل از اعمال تابع بیشینه هموار، $\acute{D}$، در بردار سوم مرحله اول، C، مطابق رابطه ۱۲-۱۰۵ ضرب شده و تبدیل به یک با اندازه برابر با بردارهای حاصل در مرحله اول می شود. این بردار همان بردار اولیه ای است که به آن وزن های مکانیزم توجه اعمال شده است.

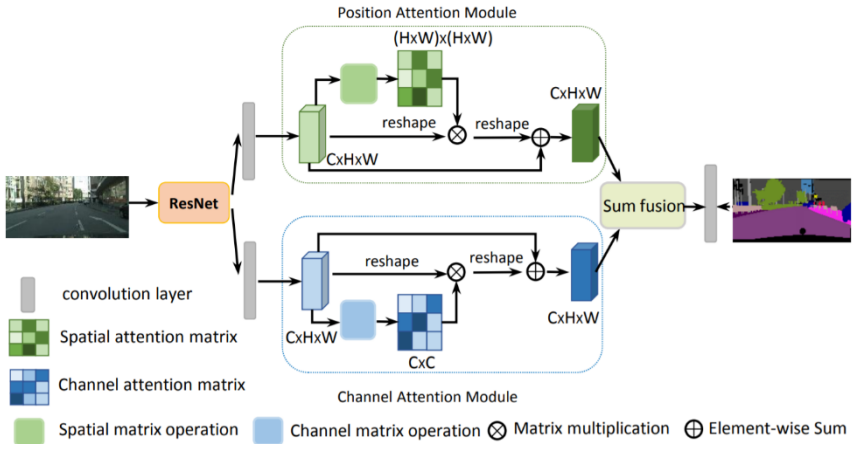
$$O_{n\times 1} = D'_{n\times n} \times C_{n\times 1} \qquad (۱۲-۱۰۵)$$

۶. در ادامه بردار حاصل از مرحله ۵ تغییر شکل داده و به صورت یک تانسور با ابعادی برابر با تانسور های حاصل از اعمال لایه های پیچشی سه بعدی ۱×۱ در مرحله ۱ می شود. سپس به این تانسور یک لایه پیچشی سه بعدی ۱×۱ اعمال می شود تا عمق آن برابر با تانسور ورودی اولیه شود.

شکل ۱۲-۵۴ شمای کلی این مکانیزم را نشان می دهد. در برخی ساختار های پیچشی می توان از هر دو مکانیزم توجه مبتنی بر کانال و مکان همزمان استفاده کرد. شبکه Dual attention network یکی از ساختار های دارای هر دو مکانیزم توجه است که برای قطعه بندی معنائی تصویر به کار برده شده و شمای ساختار آن مطابق شکل ۱۲-۵۵ است.



شکل ۵۴-۱۲: روند مکانیزم توجه مبتنی بر مکان[۱۵۲]



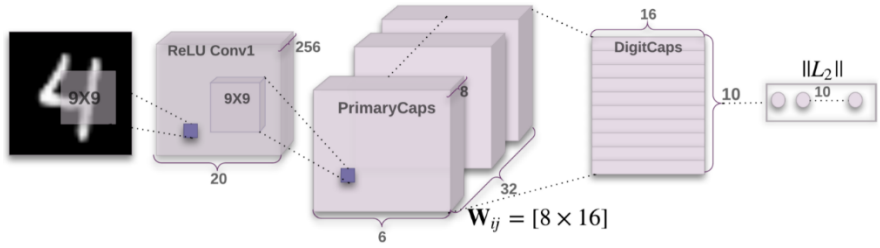
شکل ۵۵-۱۲: ساختار شبکه Dual Attention Network [۱۱۹]

۱۲.۱۵.۵.۳ مکانیزم توجه چندگانه در ساختارهای پیچشی

برخی از ساختارهای پیچشی دارای چند تانسور خروجی موازی مانند ساختار شکل ۵۶-۱۲ هستند. در این ساختارها به هر کدام از این تانسورها عملیات کاهش بعدی مانند تجمیع سراسری و... اعمال شده و در نهایت همه آن ها در کنار هم قرار گرفته و یا با هم جمع شده، میانگین، و تشکیل یک بردار را می دهند که در ادامه ساختار، لایه های تماما متصل و...، از آن استفاده می شود. این نوع از ساختارها محدود به ساختارهای پیچشی نبوده و می توان خروجی سایر ساختارهای مرسوم را نیز به صورت چندگانه تعریف کرد. این ساختارها با عنوان شبکه کپسولی[1] شناخته شده و

[1] Capsule networks

به هر یک از خروجی های آن یک کپسول گفته می شود.



شکل ۵۶-۱۲: ساختار شبکه پیچشی CapsNet [۱۱۶]

در صورتی که در یک ساختار شبکه کپسولی از میانگین کپسول ها برای ایجاد خروجی نهائی استفاده شده باشد می توانیم با اعمال مکانیزم توجه چندگانه، این میانگین را تبدیل به میانگین وزن دار کنیم. در این صورت هر کپسول نسبت به مقادیر وزن خود در ایجاد تانسور خروجی نهائی نقش خواهد داشت. روند اجرای این مکانیزم در ساختار شبک های کپسولی به شرح زیر است:

۱. برای هر کپسول یک پارامتر اسکالر در نظر گرفته می شود.

۲. به مجموعه پارامترهای مرحله ۱، یک تابع بیشینه هموار اعمال می شود تا به اعدادی در بازه صفر تا یک به طوری که مجموع آن ها یک باشد تبدیل شوند.

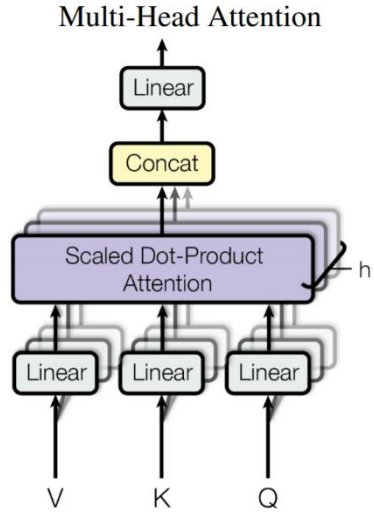
۳. مقدار خروجی تابع بیشینه هموار منتناظر با هر کپسول با در مقادیر همه درایه های کپسول ضرب می شود[1].

۴. سپس درایه های تانسور های همه کپسول ها نظیر به نظیر با هم جمع شده و حاصل یک تانسور با ابعادی برابر با آن می شود. این تانسور حاصل از مجموع کپسول ها همان تانسور خروجی نهائی است که به آن مکانیزم توجه چندگانه اعمال شده است.

شکل ۵۷-۱۲ شمای کلی مکانیزم توجه چندگانه را نشان می دهد. می توانیم با دخیل کردن مقادیر کپسول ها در کنار پارامترهای اسکالر مرحله ۱، به صورت ضرب یخشی از مقادیر یا همه مقادیر کپسول در پارامتر مرحله ۱ و...، پیش از اعمال تابع بیشینه هموار مرحله ۲ این مکانیزم را به یک مکانیزم توجه خود محور تبدیل کنیم، در این صورت ابعاد پارامترهای مرحله اول با توجه به عملیات تعریف شده ممکن است برای هر کپسول به صورت اسکالر تعریف نشود ولی خروجی ضرب مقادیر کپسول در آن پارامتر باید همواره اسکالر باشد و تابع بیشینه هموار به بردار اسکالرهایی با

[1] Dot product

اندازه ای برابر با تعداد کپسول ها اعمال شود.



شکل ۵۷-۱۲: مکانیزم توجه چندگانه[۵]

۱۲.۱۶ داده افزائی[1] در ساختارهای پیچشی

تهیه یک مجموعه دادگان آموزشی مناسب برای آموزش یک ساختار پیچشی مستلزم صرف وقت و هزینه بسیار زیادی است، این معضل در مواردی که مقادیر مطلوب، برچسب، تعریف شده برای ساختار دارای قالب خاصی هستند؛ مانند ساختارهای دارای خروجی متغیر که در آن ها برچسب هر تصویر شامل کلاس و مختصات اجسام آن است، نمود بیشتری پیدا می کند. بنابراین ممکن است در مواردی برای آموزش ساختار پیچشی مجموعه دادگانی در دسترس باشد که تعداد نمونه های آن کمتر از حد مطلوب باشد، در این صورت ممکن احتمال بروز پدیده بیش برازش برای ساختار به ویژه اگر تعداد گام های آموزشی زیاد باشد افزایش می یابد، علاوه بر این ساختار حاصل دامنه گسترده مناسبی را پوشش نداده و فقط برای تصاویری که به تصاویر آموزشی شباهت زیادی دارند کارایی دارد. برای رفع این معضلات در ساختارهای پیچشی معمولا یک عملیات پیش پردازش افزایش دادگان تعریف می شود. در عملیات افزایش دادگان با اعمال تبدیل هایی به مجموعه تصاویر دادگان آموزشی تصاویر جدیدی ایجاد و به مجموعه اضافه می شود. این تبدیل ها به دو دسته کلی تعریف می شوند:

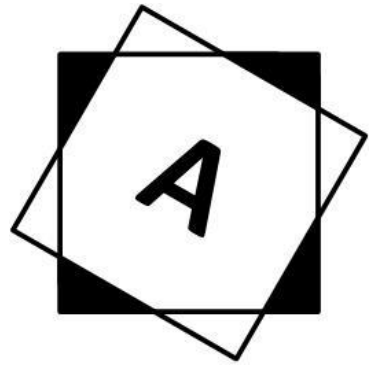
۱. دسته اول تبدیل هایی که بر ویژگی های هندسی[2] تصویر اعمال می شوند مانند

[1] Data augmentation
[2] Geometric transforms

دوران[1]، تغییر اندازه[2]، برش[3] و حذف تصادفی بخش کوچکی از تصویر[4] و...

۲. دسته دوم تبدیل هایی که به ویژگی های ظاهری[5] اعمال می شوند مانند فیلتر بلور[6]، اعوجاج رنگ[7]، افزودن نویز به تصویر و...

با اعمال هر یک از تبدیل ها به هر یک از تصاویر مجموعه دادگان یک تصویر جدید ایجاد می شود، برخی با تغییر پارامترهای تبدیل هایی که دارای چنین پارامترهایی هستند می توان یک تبدیل را به چند حالت تعریف کرده و تعداد تصاویر بیشتری تولید کرد؛ مانند تنظیم درجه دوران در تبدیل دوران. با اعمال هر یک از تبدیل ها به مجموعه دادگان مقادیر مطلوب متناظر با آن تبدیل نیز می بایست در صورت نیاز تغییر کنند، برای مثال با اعمال تبدیل تغییر اندازه محل جسم در تصویر تغییر می کند و اگر مقادیر مطلوب شامل مختصات اجسام تصویر نیز باشند باید برای تصویر حاصل از تبدیل اصلاح شوند. باید در نظر داشته باشیم تبدیل های هندسی به محتویات تصویر اعمال می شوند و ابعاد کلی تصویر حاصل، تانسور، همواره با تصاویر اولیه برابر است برای مثال تبدیل دوران با زاویه ای بین صفر تا ۹۰ درجه ساعتگرد را در نظر بگیرید که به یک تصویر اعمال می شود مطابق شکل ۵۸-۱۲، تصویر حاصل شامل بخش هایی است که فراتر از ابعاد تصویر اولیه هستند، قسمت های مثلثی سفید، همچنین برخی قسمت های محدوده تصویر اولیه نیز فاقد مقدار هستند، قسمت های سیاه رنگ؛ در این حالت برای حفظ ابعاد تصویر اولیه قسمت های خارج از محدوده در نظر گرفته نشده و برای مقادیر پیکسل های بخش های تعریف نشده داخل محدوده، قسمت های سیاه رنگ، مقدار صفر، مشابه روش بعد گذاری، در نظر گرفته می شود.



شکل ۵۸-۱۲: اعمال تبدیل دوران به یک تصویر

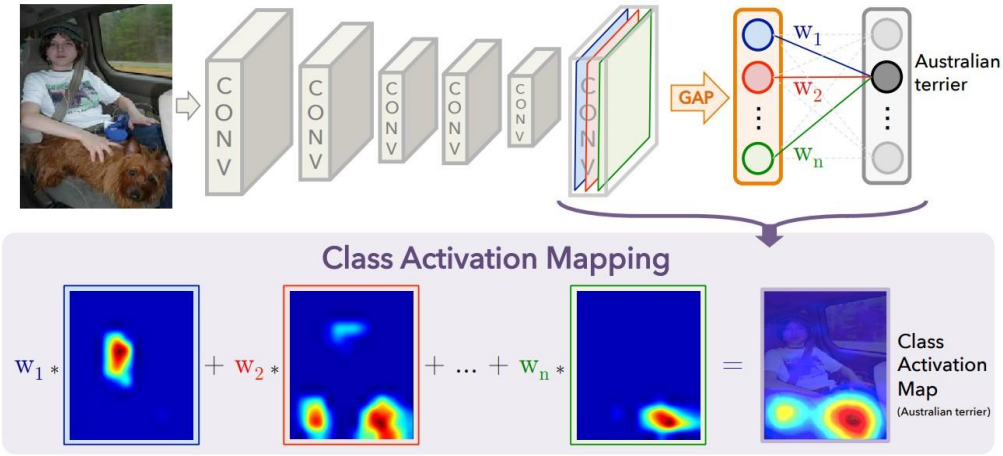
[1] Rotation
[2] Resize
[3] Crop
[4] Random erasing
[5] Appearance transforms
[6] Blur filter
[7] Color distortion

از روش افزایش دادگان می توان در مواردی که تعداد نمونه های کلاس های مختلف در مجموعه دادگان آموزشی با هم برابر نبوده و استفاده از آن مجموعه باعث بایاس شدن مدل به کلاس هایی با تعداد نمونه بیشتر می شود نیز برای افزایش نمونه های یک کلاس استفاده کرد. حتی در مواردی که تعداد تصاویر مجموعه دادگان آموزشی مطلوب است نیز معمولا از این روش به منظور تعمیم دامنه ورودی مدل استفاده می شود. این روش منحصر به دادگان تصویری نبوده و می توان با تعریف تبدیل های مناسب برای سایر دادگان نیز از آن بهره گرفت.

۱۲.۱۷ نگاشت فعال سازی کلاس[1]

در روند توسعه یک ساختار پیچشی، برای مثال یک ساختار پیچشی طبقه بند، پیدا کردن نواحی از تصویر ورودی که تاثیر بیشتری در تشخیص کلاس تصویر ورودی دارند نقش موثری در توسعه ساختار، نحوه آموزش و بهبود عملکرد آن دارد. از این رو روش نگاشت فعال سازی کلاس یکی از رویکردهایی است که بدین منظور توسعه یافته است. این رویکرد تنها در ساختارهای پیچشی که در انتهای لایه های پیچشی از لایه تجمیع سراسری استفاده شده است کاربرد داشته و در مواردی که از لایه مسطح ساز در ساختار استفاده شده باشد کاربردی ندارد. شکل ۵۹-۱۲ روند کلی این رویکرد را نشان می دهد. طبق شکل ۵۹-۱۲ برای تعیین نواحی موثر تصویر ورودی مقادیر وزن های مربوط به هر یک از عمق های تانسور ورودی که در تعیین خروجی مطلوب، نورون کلاس مدنظر، مربوط به تصویر ورودی دخیل هستند، به لایه تجمیع سراسری که به صورت یک بعد از بردار تغییر پیدا کرده اند در مقادیر آن عمق ضرب می شود. با توجه به استفاده از توابع فعال سازی مانند ReLU در این ساختارها بخش قابل توجهی از این عمق ها تنک بوده و دارای درایه های صفر است. پس از ضرب وزن مربوط به هر عمق در مقادیر درایه های آن، ضرب اسکالر در ماتریس، عمق های حاصل با هم جمع شده و تبدیل به یک ماتریس می شود. ابعاد ماتریس حاصل را افزایش، معمولا به روش دوخطی[2]، و با طول و عرض تصویر ورودی برابر می کنند. در این صورت ماتریس حاصل همانند ماسکی است که نواحی موثر تصویر ورودی را مشخص می کند.

[1] Class activaion mapping (CAM)
[2] Bilinear



شکل ۵۹-۱۲: رویکرد نگاشت فعال سازی کلاس [۱۸۹]

۱۲.۱۸ مسائل

۱. روابط آموزش مرتبط با مکانیزم توجه مبتنی بر مکان را تعریف کنید.
۲. در مورد روش های تغییر ابعاد تصاویر، مانند Bilinear، تحقیق کنید.
۳. تفاوت های ساختار پیچشی[1] و همبسته[2] را توضیح دهید.
۴. در روابط این فصل و فصل ۱۱ همه روابط مربوط به ضرب پیچشی و تجمیع برای گام حرکتی یک در نظر گرفته شـــده اند، این روابط را برای گام های حرکتی بزرگ تر توسعه دهید.
۵. در مورد رویکرد های توسـعه داده شـده بر اسـاس روش نگاشـت فعال سـازی کلاس تحقیق کنید.

[1] Convolution
[2] Correlation

فصل ۱۳
شبکه های مولد[1]

۱۳.۱ مقدمه

در فصـــل ۶ در مورد مدل های مولد[2] و متمایزکننده[3] بحث کرده و ویژگی های هر کدام را معرفی کردیم. همچنین تا کنون در بخش های مختلف نمونه های مختلفی از این مدل ها را بررسی کرده ایم. ماشـین های بولتزمن، خودرمزگذارهای متغیر و... نمونه هایی از مدل های مولد هسـتند. اکثر ساختارهای معرفی شده در حوزه یادگیری ژرف مانند شبکه های پرسپترون چندلایه، شبکه های بازگشتی و حافظه دار و شبکه های پیچشی دارای ماهیت متمایز کننده هستند، اما در کنار این ساختارهای متمایزکننده، مدل هایی نیز تعریف می شوند که از ساختاری مشابه با ساختارهای ژرف متمایزکننده که در فصل های پیشین معرفی کردیم بهره می برند، در این فصل به معرفی این مدل های مولد خواهیم پرداخت. در ادامه مطالب این فصل به ساختارهای مولد ترکیبی که از چند ساختار مولد مختلف تشکیل شده اند نیز پرداخته و مزایا و معایب آن ها را بررسی خواهیم کرد. در انتهای فصل نیز مسائل مربوط به آن برای آشنائی بیشتر با مطالب ارائه شده است.

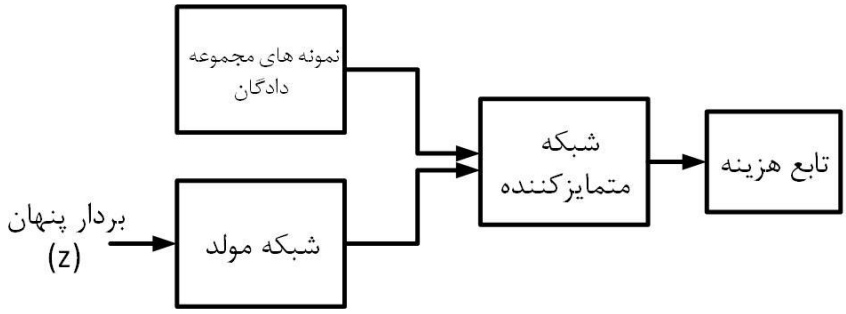
۱۳.۲ شبکه های متخاصم مولد[4]

ایده مربوط به شـبکه های متخاصـم مولد، قابلیت پیاده سـازی در همه سـاختارهای ژرفی که دارای آموزش بر اساس پس انتشار خطا هستند را دارد؛ بدین ترتیب می توان شبکه های متخاصم مولد را در قالب سـاختارهای پرسـپترون، مبتنی بر کرنل، بازگشـتی، پیچشـی و... تعریف کرد ولی معمولا این شـبکه ها دارای سـاختارهای پیچشـی هسـتند. سـاختار این نوع شـبکه ها از یک شـبکه متمایز کننده و یک شبکه مولد مطابق شکل ۱-۱۳ تشکیل شده است. آموزش این ساختارها مانند اکثر ساختارهای عمیق به صورت با سرپرست تعریف می شود ولی علیرغم این نوع آموزش در حالت کلی لزومی بر تعریف مقادیر مطلوب، برچسـب، برای نمونه های مجموعه دادگان آموزشـی این سـاختارها، مشـابه آنچه برای سـاختارهای تعریف شـده برای مسـائل رگرسـیون و طبقه بندی تعریف

---

[1] Generative networks
[2] Generative models
[3] Discriminative models
[4] Generative adversarial networks

می شود، وجود ندارد؛ تعلق یک نمونه به مجموعه دادگان آموزشی در این ساختارها، خود به عنوان مقدار مطلوب آن نمونه شناخته می شود.



شکل ۱-۱۳: ساختار کلی شبکه های متخاصم مولد

۱۳.۲.۱ روند پیشرو در شبکه های متخاصم مولد

با توجه به ساختار شکل ۱-۱۳ می توانیم روند پیشرو شبکه های مولد متخاصم را مطابق ذیل تعریف کنیم:

۱. در ابتدا تعدادی بردار ورودی تصادفی دلخواه، بردار نویز، به عنوان ورودی شبکه مولد در نظر می گیریم که به هر یک از آن ها بردار پنهان[1]، z، گفته می شود؛ می توان این بردارها را به صورت نمونه هایی از یک توزیع احتمالاتی تعریف کرد که براساس آن با توجه به ساختار شبکه مولد، خروجی این شبکه محاسبه می شود، شبکه مولد می تواند ساختارهای مختلفی مانند پیچشی، بازگشتی و... داشته باشد ولی باید همواره توجه داشته باشیم که ابعاد و ساختار خروجی شبکه مولد برابر با نمونه های مجموعه دادگان آموزشی باشد، همچنین بردارهای پنهانی که در ابتدا به عنوان ورودی شبکه مولد تعریف می شوند در طول روند آموزش ساختار ثابت بوده و تغییری نمی کنند.

۲. نمونه های مجموعه دادگان آموزشی و نیز نمونه های حاصل از خروجی شبکه مولد به عنوان ورودی شبکه متمایزکننده تعریف می شود، شبکه متمایزکننده نیز مانند شبکه مولد می تواند ساختارهای مختلفی داشته باشد، در حالت کلی این شبکه یک شبکه طبقه بند برای دو کلاس ناسازگار است که تعیین می کند نمونه ورودی مربوط به مجموعه دادگان آموزشی است، نمونه واقعی[2]، و یا نمونه ای است که شبکه مولد آن را ایجاد کرده است، نمونه مصنوعی[3]، بدین ترتیب خروجی این شبکه به صورت یک اسکالر تعریف می شود، مقدار مطلوب صفر در خروجی این

[1] Latent vector
[2] Real sample
[3] Fake sample

شـبکه برای کلاس نمونه های واقعی و مقدار یک برای کلاس نمونه های مصـنوعی در نظر گرفته می شود.

۱۳.۲.۲ تابع هزینه شبکه های متخاصم مولد

در ساختار شبکه های مولد متخاصم هدف از آموزش شبکه متمایزکننده تشخیص کلاس نمونه های ورودی، تمایز نمونه واقعی از مصنوعی، و هدف از آموزش شـبکه مولد تولید نمونه هایی مشابه نمونه های مجموعه دادگان اسـت به طوری که شـبکه متمایزکننده توانایی تشـخیص آن به عنوان نمونه مصـنوعی را نداشـته باشـد، بدین ترتیب در این سـاختار هدف این دو شـبکه تضـعیف عملکرد سـاختار دیگر است از این رو می توانیم عملکرد هر یک از این دو شبکه را با توجه به عملکرد شبکه دیگر بررسـی کنیم، بر این اسـاس تابع هزینه در سـاختار شـبکه های متخاصـم مولد نیز به صـورت کمینه-بیشـینه[1]، تعریف شـده و برای هر دو سـاختار مشـترک اسـت. بدین صـورت که اگر هدف از آموزش شـبکه متمایز کننده رسـاندن تابع هزینه به مقدار حداقل باشـد هدف از آموزش شـبکه مولد رساندن همان تابع هزینه به مقدار حداکثر است. با توجه به تعریف ساختار شبکه متمایز کننده می توانیم تابع هزینه سـاختار شـبکه مولد متخاصـم را به صـورت یک تابع آنتروپی متقابل تعریف کنیم که هدف شـبکه متمایزکننده به حداقل رسـاندن مقدار آن و هدف شـبکه مولد به حداکثر رسـاندن مقدار آن است، با این توضیحات تابع هزینه یک شبکه مولد متخاصم در حالت کلی به صورت رابطه ۱-۱۳ تعریف می شود. در رابطه ۱-۱۳، منظور از D, G به ترتیب شبکه مولد و متمایز کننده، $\mathrm{E}_i[.]$ امید ریاضـــی، میانگین، به ازای نمونه های i، x نمونه های مجموعه دادگان و z بردار پنهان ورودی شبکه مولد است.

$$\min_G \max_D V(D,G) = \mathrm{E}_x[\log D(x)] + \mathrm{E}_z[\log(1-D(G(z)))] \quad (۱۳-۱)$$

بدین ترتیب تابع هزینه شبکه متمایز کننده به صورت رابطه ۲-۱۳ تعریف می شود.

$$E_{D(x,G(z))} = \mathrm{E}_x[\log D(x)] + \mathrm{E}_z[\log(1-D(G(z)))] \quad (۱۳-۲)$$

هدف از آموزش شـــبکه متمایز کننده به حداکثر رســـاندن مقدار تابع ۲-۱۳ اســـت. با توجه به مســـتقل بودن جمله اول رابطه ۱-۱۳ تابع هزینه شـــبکه مولد نیز مطابق رابطه ۳-۱۳ تعریف می شود.

$$E_{G(z)} = \mathrm{E}_z[\log(1-D(G(z)))] \quad (۱۳-۳)$$

[1] Min-Max

هدف از آموزش شبکه مولد به حداقل رساندن مقدار تابع هزینه رابطه ۳-۱۳ است. در گام های اولیه آموزش ساختار شبکه که عملکرد شبکه مولد ضعیف است مقدار تابع هزینه ۳-۱۳ مقدار بسیار بزرگی دارد، تابع هزینه اشباع[1] شده است، مقادیر این تابع باعث ایجاد گردایان های بزرگ و افزایش احتمال ناپایداری روند آموزش شبکه مولد می شود، برای رفع این معضل می توانیم به جای استفاده از رابطه ۳-۱۳ تابع هزینه شبکه مولد را به صورت رابطه ۴-۱۳ تعریف کنیم. در صورت استفاده از رابطه ۴-۱۳ به عنوان تابع هزینه شبکه مولد، هدف از آموزش شبکه به حداکثر رساندن این مقدار خواهد بود. استفاده از تابع هزینه رابطه ۴-۱۳ باعث بهبود روند آموزش شبکه مولد در گام های آموزشی اول می شود. به تابع هزینه رابطه ۴-۱۳ که برای شبکه مولد تعریف می شود، تابع هزینه غیر اشباع کننده[2] اطلاق می شود.

$$E_{G(z)} = \mathrm{E}_z[\log(D(G(z))] \tag{۱۳-۴}$$

با توجه به استفاده از نمونه های خروجی شبکه مولد و مجموعه دادگان آموزشی در رابطه تابع هزینه ۱-۱۳ و ۲-۱۳ آموزش شبکه های مول متخاصم به صورت دسته ای[3] یا دسته ای کوچک[4] تعریف می شود.

۱۳.۲.۳ آموزش شبکه های متخاصم مولد

با توجه به تعریف تابع هزینه برای شبکه های مولد متخاصم، آموزش این شبکه ها نیز در دو مرحله انجام می شود، نخست مقدار تابع هزینه برای شبکه متمایز کننده، رابطه ۲-۱۳ با استفاده از نمونه های مجموعه دادگان آموزشی، x، و خروجی شبکه مولد، z، محاسبه می شود. بر اساس این مقدار مشتقات زنجیره ای برای پارامترهای آموزشی شبکه متمایزکننده محاسبه شده و مقادیر این پارامترها مطابق رابطه ۵-۱۳ به روش گرادیان افزایشی[5] به روز رسانی می شود؛ دلیل استفاده از گرادیان افزایشی به جای گرادیان نزولی برای به روزرسانی مقادیر پارامترها در رابطه ۵-۱۳ افزایش مقدار تابع هزینه است؛ رابطه گرادیان افزایشی مشابه گرادیان نزولی است با این تفاوت که برای به روزرسانی مقادیر یک پارامتر مقادیر گرادیان با مقدار فعلی پارامتر جمع می شوند.

$$\theta_{D(k+1)} = \theta_{D(k)} + \eta \frac{\partial E_D}{\partial \theta_D}(k) \tag{۱۳-۵}$$

[1] Saturated
[2] Non-Saturating loss function
[3] Batch
[4] Mini batch
[5] Gradient ascent

پس از به روزرسانی مقادیر مربوط به شبکه متمایزکننده، مقدار تابع هزینه شبکه مولد محاسبه و سپس وزن های آن به روش گرادیان نزولی در صورت استفاده از تابع هزینه رابطه ۳-۱۳ و به روش گرادیان نزولی، مطابق رابطه ۶-۱۳، و در صورت استفاده از تابع هزینه رابطه ۴-۱۳ به رشو گرادیان افزایشی مطابق رابطه ۷-۱۳ به روز رسانی می شود. با توجه به شکل ۱-۱۳ شبکه مولد و متمایزکننده، مانند لایه های یک شبکه عصبی، به صورت سری در کنار هم قرار گرفته اند، بنابراین در تعریف مشتقات زنجیره ای برای به روزرسانی پارامترهای شبکه مولد باید مطابق رابطه ۸-۱۳ مشتق خروجی به ورودی شبکه متمایزکننده را نیز محاسبه کنیم، این مقادیر صرفا برای محاسبه مشتقات پارامترهای شبکه مولد بوده و پارامترهای شبکه متمایزکننده در این مرحله تغییری نمی کنند. در رابطه ۸-۱۳، $\hat{y}_D, \hat{y}_G$ به ترتیب خروجی شبکه مولد و متمایزکننده است.

$$\theta_{G(k+1)} = \theta_{G(k)} - \eta \frac{\partial E_G}{\partial \theta_G}(k) \tag{۱۳-۶}$$

$$\theta_{G(k+1)} = \theta_{G(k)} + \eta \frac{\partial E_G}{\partial \theta_G}(k) \tag{۱۳-۷}$$

$$\frac{\partial E_G}{\partial \theta_G} = \frac{\partial E_G}{\partial \hat{y}_D} \frac{\partial \hat{y}_D}{\cdots} \frac{\cdots}{\partial \hat{y}_G} \frac{\partial \hat{y}_G}{\cdots} \frac{\cdots}{\partial \theta_G} \tag{۱۳-۸}$$

با توجه به توضیحات در هر مرحله به روزرسانی پارامترهای شبکه مولد متخاصم، ابتدا یک بار پارامترهای شبکه متمایزکننده با توجه به مقدار تابع هزینه متناظر به روزرسانی می شوند؛ در این بخش پارامترهای شبکه مولد تغییری نمی کنند، سپس پارامترهای شبکه مولد با توجه به مقدار تابع هزینه متناظر به روزرسانی می شوند، در این بخش نیز پارامترهای شبکه متمایزکننده ثابت باقی می مانند ولی برای محاسبه مشتقات زنجیره ای مربوط به شبکه مولد از پارامترهای شبکه متمایزکننده مطابق رابطه ۸-۱۳ استفاده می شود. آموزش ساختار شبکه های مولد متخاصم معمولا با چالش هایی مانند پدیده محوشدگی گرادیان ها و... همراه است که برای رفع این معضلات ساختارهای متنوعی ارائه شده است که در ادامه به بررسی چند نمونه از آن ها می پردازیم.

۱۳.۲.۴ کاربرد شبکه های مولد متخاصم

پس از اتمام روند آموزش شبکه متخاصم مولد مانند ساختار خودرمزگذارها، شبکه متمایزکننده از ساختار حذف شده و از شبکه مولد برای نمونه هایی مشابه با نمونه های دادگان آموزشی استفاده می شود. از نمونه هایی که شبکه مولد تولید می کند می توان برای افزایش تعداد نمونه های یک مجموعه دادگان آموزشی که برای اهداف و مسائل دیگر به کاربرده می شود استفاده کرد.

۱۳.۲.۵ ساختارهای شبکه های مولد متخاصم

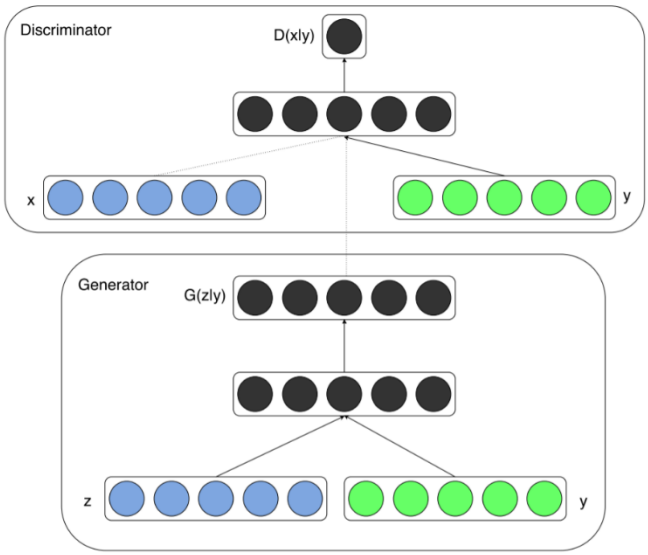
همانند ساختارهای پیچشی برای شبکه های مولد متخاصم نیز ساختارهای متنوعی با هدف بهبود جنبه های مختلف این شبکه ها ارائه شده است. با توجه به گستره وسیع ساختار های تعریف شده برای شبکه های مولد متخاصم، صرفا به بررسی ساختارهای مهم می پردازیم.

۱۳.۲.۵.۱ شبکه مولد متخاصم شرطی[1]

یکی از چالش های موجود در آموزش شبکه های مولد متخاصم تخصیص برچسب مشترک، واقعی یا مصنوعی، به کلاس های مختلف تصاویر تعریف شده در مجموعه دادگان است که باعث افزایش احتمال بیش برازش شبکه متمایزکننده و یا ناپایداری روند آموزش هر دو شبکه مولد و متمایزکننده می شود. برای رفع این معضل شبکه های مولد متخاصم شرطی معرفی شده اند. ساختار این شبکه مشابه ساختار شبکه های مولد متخاصم پایه است که پیشتر معرفی کردیم با این تفاوت که مجموعه دادگان آموزشی آن شامل برچسب های مناسب طبقه بندی هستند. همچنین در این ساختار برای هر یک از بردارهای پنهان نیز که به عنوان ورودی شبکه مولد استفاده می شوند برچسب کلاسی نسبت می داده می شود؛ در نهایت خروجی شبکه مولد متعلق به کلاس نسبت داده شده به بردار پنهان خواهد بود. از این برچسب ها به عنوان بردار ورودی هر دو شبکه مولد و متمایزکننده در کنار ورودی اصلی این شبکه ها استفاده می شود؛ بنابراین ساختار شبکه های مولد و متمایزکننده باید به گونه ای تعریف شود که از بردار برچسب نیز بتوان در کنار ورودی های اصلی این شبکه ها استفاده کرد. تابع هزینه این ساختار مطابق رابطه ۹-۱۳ تعریف می شود، استفاده از اطلاعات مربوط به کلاس نمونه ها در تابع هزینه و روند آموزش ساختار باعث کاهش احتمال بیش برازش و پایداری روند آموزش می شود. شکل کلی ساختار شبکه های مولد متخاصم شرطی مطابق شکل ۲-۱۳ است. در رابطه ۹-۱۳، y بردار برچسب ها است که معمولا به صورت بردار One-Hot تعریف می شود. ساختار شبکه های مولد متخاصم شرطی را می توان با اعمال تغییراتی در اکثر ساختارهای مرسوم تعریف شده اعمال کرد.

$$\min_{G}\max_{D} V(D,G) = \mathrm{E}_{x}[\log D(x \mid y)] + \mathrm{E}_{z}[\log(1 - D(G(z \mid y)))] \qquad (۱۳-۹)$$

[1] Conditional GAN (CGAN)

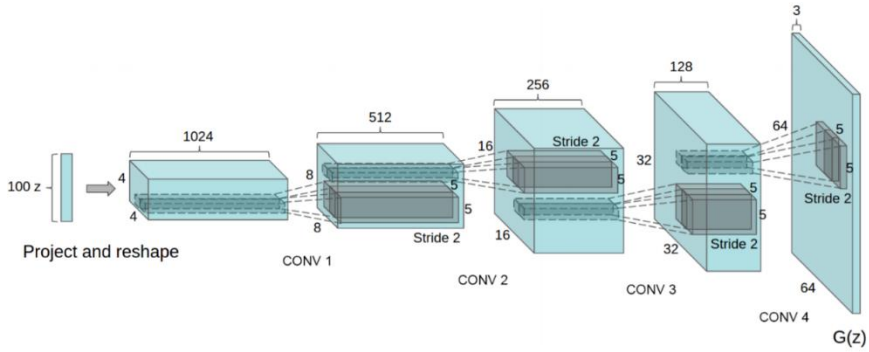


شکل ۲-۱۳: ساختار شبکه مولد متخاصم شرطی[۱۷۹]

۱۳.۲.۵.۲ DCGAN[1]

این شبکه یکی از مرسوم ترین و ساده ترین ساختارهای شبکه های مولد متخاصم مبتنی بر شبکه های پیچشی است. ساختار این شبکه در سال ۲۰۱۶ میلادی معرفی شده است. شبکه مولد این ساختار مطابق شکل ۳-۱۳ از لایه های پیچشی ترانهاده تشکیل شده است که ورودی آن یک بردار نویز تصادفی و خروجی یک تصویر، تانسور سه بعدی، است. برای شبکه متمایزکننده این ساختار نیز از شبکه های پیچشی دارای لایه های تماما متصل که در فصل ۱۲ معرفی کردیم استفاده می شود. با توجه به استفاده از لایه های پیچشی ترانهاده در این شبکه ها، همان طور که در مطالب فصل ۱۲ نیز اشاره کردیم تصاویر خروجی حالت شطرنجی[2] دارند، برای بهبود کیفیت این تصاویر پس از اتمام روند آموزش ساختار و حذف شبکه متمایزکننده، در مرحله تولید نمونه تصاویر نهائی توسط شبکه مولد به تصاویر خروجی این شبکه یک فیلتر بلور[3] یا بلور گوسی[4] اعمال می شود؛ استفاده از فیلتر بلور باعث کاهش جزئیات تصویر می شود بنابراین در مواردی که جزئیات تصاویر اهمیت بالائی دارد، مانند تصاویر پزشکی، استفاده از شبکه های مولد متخاصم برای افزایش تعداد نمونه های مجموعه دادگان آموزشی چندان مناسب نیست. تابع هزینه ارائه شده برای این ساختار مطابق تابع هزینه اصلی شبکه های متخاصم مولد، رابطه ۱-۱۳، تعریف می شود.

[1] Deep convolutional GAN
[2] Checkerboard artifacts
[3] Blur
[4] Gaussian blur



شکل ۳-۱۳: شبکه مولد ساختار DCGAN [۱۸۰]

۱۳.۲.۵.۳ WSGAN[1]

یکی دیگر از ساختارهای معرفی شده برای بهبود پایداری روند آموزش شبکه های مولد متخاصم شبکه های WSGAN است. در این ساختار برخلاف ساختارهای پیشین که در آن ها واقعی یا مصنوعی بودن تصاویر توسط شبکه متمایزکننده به صورت یک مسئله طبقه بندی تعریف می شد، فاصله توزیع احتمالاتی نمونه های مصنوعی از نمونه های مجموعه دادگان با استفاده از معیار فاصله واسااشتاین محاسبه می شود، بدین ترتیب در این ساختار شبکه متمایزکننده با یک شبکه منتقد جایگزین شده است، خروجی شبکه منتقد یک بردار است که از آن برای محاسبه فاصله واسااشتاین نمونه های مجموعه دادگان و نمونه های مصنوعی استفاده می شود. تابع هزینه این ساختار مطابق رابطه ۱۰-۱۳ تعریف می شود.

$$\min_{G}\max_{D} WS_1(P_{(\mathbf{x})}, P_{(\hat{\mathbf{x}})}) = \inf_{\Gamma\in p(\mathbf{x}\sim P_{(\mathbf{x})}, \hat{\mathbf{x}}\sim P_{(\hat{\mathbf{x}})})} \mathrm{E}_{(\mathbf{x},\hat{\mathbf{x}})\sim\Gamma}[d(\mathbf{x},\hat{\mathbf{x}})] \qquad (۱۳-۱۰)$$

در رابطه ۱۰-۱۳، $x, \hat{x}$ به ترتیب بردار خروجی حاصل از شبکه منتقد به ازای نمونه های مصنوعی و نمونه های مجموعه دادگان و $d(.)$ فاصله اقلیدسی است. رابطه ۱۰-۱۳ را می توانیم به صورت رابطه ۱۱-۱۳ که فرم دوگانه کانتورویچ-رابینشتاین[2] است، توسعه دهیم. در رابطه ۱۱-۱۳، F_L مجموعه توابع n-لیپشیتس[3] است. در اینجا از تابع ۱- لیپشیتس استفاده می کنیم، به صورت زیروند WS نشان داده شده است، که ساده ترین نوع آن یک تابع همانی $(y = x)$ است.

$$\min_{G}\max_{D} WS_1(P_{(\mathbf{x})}, P_{(\hat{\mathbf{x}})}) = \sup_{f\in F_L} \mathrm{E}_{\mathbf{x}\sim P_{(\mathbf{x})}}[f(\mathbf{x})] - \mathrm{E}_{\hat{\mathbf{x}}\sim P_{(\hat{\mathbf{x}})}}[f(\hat{\mathbf{x}})] \qquad (۱۳-۱۱)$$

[1] Wasserstein GAN
[2] Kantorvich-Rubistein's duality
[3] n-Lipshitz functions

۱۳.۲.۵.۴ Self-supervised GAN

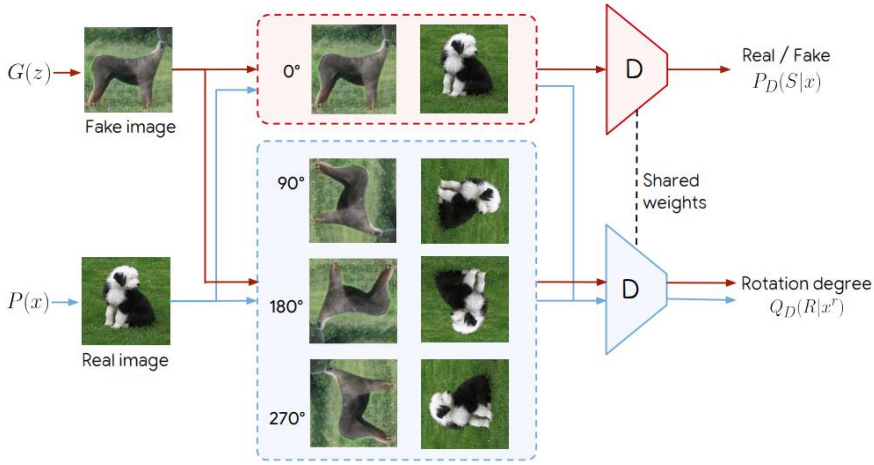
یکی از مهم ترین معضلات در روند آموزش ساختار شبکه های مولد متخاصم، آموزش تناوبی است، بدین صورت که تابع هزینه در طی چند گام آموزشی ابتدا کاهش و سپس دوباره به مقدار اولیه باز می گردد، این روند تا پایان آموزش ساختار ادامه یافته و در نهایت آموزش مناسبی برای آن صورت نمی گیرد دلیل این موضوع فراموشی شبکه متمایزکننده[1] است که متعاقبا باعث اختلال در روند آموزش شبکه مولد نیز می شود؛ منظور از فراموشی از بین رفتن تاثیر آموزش وزن های شبکه است. برای رفع این معضل در ساختار شبکه های متخاصم مولد مبتنی بر شبکه های پیچشی از یک رویکرد خود ناظر[2] استفاده شده است؛ بدین ترتیب که پس از تولید تصاویر توسط شبکه مولد، به تصاویر تولید شده توسط شبکه مولد و همچنین تصاویر مجموعه دادگان پیش از اعمال آن ها به عنوان ورودی به شبکه متمایز کننده یک نگاشت دوران تصادفی اعمال شده و درجه این دوران به عنوان برچسب هر تصویر در نظر گرفته شده و می توانند در هر گام آموزشی به طور تصادفی متفاوت از نگاشت های گام قبلی اعمال شوند. در ادامه شبکه متمایزکننده علاوه بر تشخیص مصنوعی یا واقعی بودن، جهت دوران را نیز تشخیص می دهد، بدین منظور شبکه متمایز کننده در این ساختار به صورت دو شبکه مجزا با وزن های مشترک مطابق شکل ۳-۱۳ تعریف می شود که تفاوت آن ها در لایه خروجی است. تابع هزینه این ساختار نیز به ترتیب برای شبکه متمایز کننده و مولد مطابق روابط ۱۳-۱۲ و ۱۳-۱۳ تعریف می شود، طبق این روابط هدف از آموزش شبکه متمایزکننده تشخیص مصنوعی یا واقعی بودن نمونه و همچنین زاویه نگاشت آن و هدف از آموزش شبکه مولد تولید نمونه هایی مشابه نمونه های مجموعه دادگان به نحوی که جهت دوران آن ها قابل تشخیص باشد است. در روابط ۱۳-۱۲ و ۱۳-۱۳، $V(G,D)$ همان رابطه ۱-۱۳ است که در حالت کلی به عنوان تابع هزینه ساختارهای متخاصم مولد تعریف می شود، R جهت دوران، Q_D درست نمائی خروجی شبکه متمایزکننده برای تشخیص جهت دوران، $x,\hat{x}$ به ترتیب نمونه های مجموعه دادگان و نمونه های مصنوعی حاصل از شبکه مولد و x نمونه ورودی به شبکه متمایزکننده برای بررسی دوران نمونه، مصنوعی یا واقعی، و α,β ضرایب اسکالر کنترلی برای تاثیر جمله مربوط به دوران است. هدف از آموزش شبکه متمایزکننده به حداکثر رساندن مقدار رابطه ۱۳-۱۲ و هدف از آموزش شبکه مولد به حداقل رساندن مقدار رابطه ۱۳-۱۳ است. در این ساختار برای هر دو شبکه از روش گردایان افزایشی استفاده می شود؛ گرادیان نزولی با اعمال ضریب منفی به $V(G,D)$ در رابطه ۱۳-۱۳ در شبکه مولد تعریف می شود، بنابراین رابطه به روزرسانی پارامترهای این ساختار برای هر دو شبکه متمایزکننده و مولد مطابق رابطه ۱۳-۱۴ تعریف می شود.

$$E_D = V(G,D) - \beta \mathrm{E}_x \mathrm{E}_{\hat{x}}[\log Q_D(R \mid x^r)] \qquad (۱۳-۱۲)$$

[1] Discriminator forgetting
[2] Self-supervised approach

$$E_G = -V(G,D) - \alpha \mathrm{E}_x \mathrm{E}_{\hat{x}}[\log Q_D(R \mid x^r)] \qquad (۱۳-۱۳)$$

$$\theta_{(k+1)} = \theta_{(k)} + \eta \frac{\partial E}{\partial \theta}(k) \qquad (۱۳-۱۴)$$



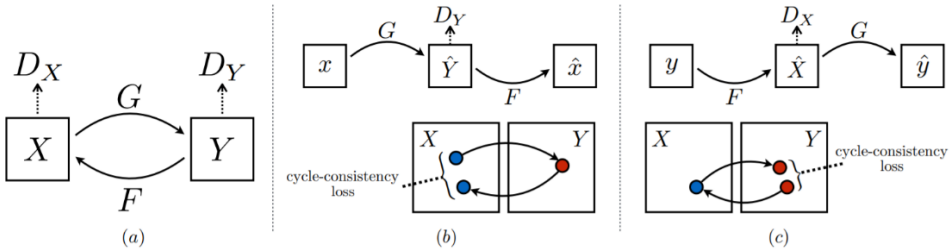
شکل ۴-۱۳: ساختار شبکه Self-supervised GAN [۱۶۲]

۱۳.۲.۵.۵ Cycle-GAN

این ساختار در اصل ترکیب دو ساختار مولد متخاصم در جهت معکوس مطابق شکل ۵-۱۳a است؛ بدین صورت که خروجی شبکه مولد یک ساختار به عنوان ورودی ساختار دیگر محسوب می شود. در این ساختار نمونه های دادگان هر دامنه در طول روند آموزش به عنوان بردار ورودی، بردار نویز، یکی از دو شبکه مولد در نظر گرفته می شود. روند عملکرد شبکه مولد متمایزکننده برای هر دو ساختار مشابه ساختار های پیشین است. از این ساختار می توان برای تبدیل دو دامنه متفاوت به هم؛ مانند تبدیل صوت به تصویر و بالعکس نیز استفاده کرد. در این ساختارها برای ایجاد تکرارپذیری به ساختار یک جمله به تابع هزینه رابطه ۱۳-۱ با اعمال یک اسکالر تنظیم کننده، مطابق رابطه ۱۵-۱۳ اضافه می شود، با توجه به رابطه ۱۳-۱۵ و رابطه ۱-۱۳ تاثیر این جمله فقط به شبکه مولد اعمال می شود. به این جمله تابع هزینه ثبات چرخه[1] اطلاق می شود. شکل های ۵-۱۳ b و c نحوه عملکرد ساختار برای تبدیل یک دامنه به دامنه دیگر و تاثیر تابع هزینه ثبات چرخه را نشان می دهد. در رابطه ۱۵-۱۳ و شکل ۵-۱۳، G و F شبکه های مولد و D شبکه متمایزکننده بوده و همچنین در رابطه ۱۵-۱۳، E(.) نماد امید ریاضی است.

$$L_{Cycle-consistensy}(G,F) = \mathrm{E}_x(\|F(G(x)) - x\|_1) + \mathrm{E}_y(\|G(F(y)) - y\|_1) \qquad (۱۳-۱۵)$$

[1] Cycle-consistency loss function



شکل ۵-۱۳: ساختار شبکه Cycle-GAN [۱۸۲]

۱۳.۳ ترکیب خودرمزگذارها و شبکه های مولد متخاصم

در مطالب فصـــل ۱۰ به معرفی خودرمزگذارهای متغیر پرداخته و انواع مختلف آن از جمله ساختارهای شامل لایه های پیچشی را بررسی کردیم. در مطالب این فصل نیز اشاره کردیم که در حالت کلی ورودی شبکه مولد یک نویز تصادفی است و کنترل خاصی بر آن اعمال نمی شود، در صـورتی که می توان با کنترل توزیع این ورودی روند آموزش سـاختار را بهبود داد بدین ترتیب در ابتدای ســـاختار شـــبکه مولد یک رمزگذار متغیر اعمال می کنیم که ورودی آن همان نمونه های مجموعه دادگان استفاده شده به عنوان ورودی شبکه متمایزکننده است. بدین ترتیب ورودی نویز شـبکه مولد بر اسـاس نمونه های تولید شـده توزیع پنهان خودرمزگذار متغیر تعریف می شـود؛ به بیانی دیگر ساختار حاصل یک شبکه خودرمزگذار متغیر است که قسمت کدگشای آن به صورت یک شبکه مولد متخاصم تعریف می شود.

۱۳.۴ مسائل

۱. مشتق ورودی نسبت به خروجی توابع رابطه
۲. روابط پیشرو و آموزش مربوط به یک ساختار شبکه متخاصم مولد شرطی که ساختار شبکه های مولد و متمایزکننده آن از لایه های پرسپترون تشکیل شده است را بنویسید.
۳. با اعمال تغییراتی در ساختار شبکه DCGAN آن را به صورت شبکه متخاصم مولد شرطی توسعه دهید.
۴. به جز استفاده از نگاشت تصادفی دوران در مورد سایر رویکردهای ارائه شده برای ساختارهای متخاصم مولد خود ناظر تحقیق کنید.

فصل ۱۴
مدل های زبانی و مکانیزم توجه[1]

۱۴.۱ مقدمه

در فصل ۱۱ در مورد واحدهای حافظه داری مانند LSTM و GRU بحث کرده و ساختار آن ها را معرفی کردیم. این واحدهای حافظه دار را در فصل ۱۱ به عنوان یک نورون در ساختارهای عصبی استفاده می کردیم. در این فصل سایر ساختارهای متشکل از این واحدهای حافظه دار را بررسی خواهیم کرد. کاربرد اکثر این واحدها در پردازش زبان های طبیعی[2] و توسعه مدل های زبانی است که در ادامه فصل به بحث در مورد آن ها نیز خواهیم پرداخت؛ لازم به ذکر است مطالب مربوط به پردازش زبان های طبیعی بسیار گسترده بوده و در این بخش صرفا مطالبی که در حوزه یادگیری عمیق کاربرد دارند بررسی می شوند. پس از آن مکانیزم توجه و انواع مختلف را که کاربرد گسترده ای در پردازش زبان های طبیعی دارد بررسی خواهیم کرد. چنان که در مطالب فصل ۱۲ نیز بررسی کردیم مکانیزم توجه کاربرد گسترده ای در سایر ساختارها مانند ساختارهای پیچشی نیز دارد؛ در برخی از این کاربردها مانند نویسه خوان نوری[3]، این مکانیزم به صورت گونه ای از یک مدل زبانی تعریف شده و در برخی دیگر به طور مستقل در ساختار تعریف شده است، مشابه ساختار پیچشی SE-Net. مکانیزم توجه در ابتدا برای مدل های زبانی تعریف شده و توسعه آن نیز بر اساس چالش های مطرح در همین حوزه بوده است، از این رو بررسی و تعمیق مطالب مربوط به آن در کنار مباحث مربوط به پردازش زبان های طبیعی و مدل های زبانی نسبت به سایر ساختارها بیشتر حائز اهمیت است. پس از این مطالب در انتهای فصل مسائل مربوط به منظور آشنائی و تحقیق بیشتر در مورد مطالب مطرح شده است.

۱۴.۲ مدل های زبانی

در مطالب فصل ۱۱ ساختارهای حافظه داری که بررسی کردیم، مشابه ساختار شبکه های پرسپترون چند لایه تعریف می شدند با این تفاوت که در ساختار آن ها به جای نورون های مرسوم از واحدهای حافظه داری مانند LSTM و GRU مطابق شکل ۱-۱۴ استفاده می شد. مشکل این

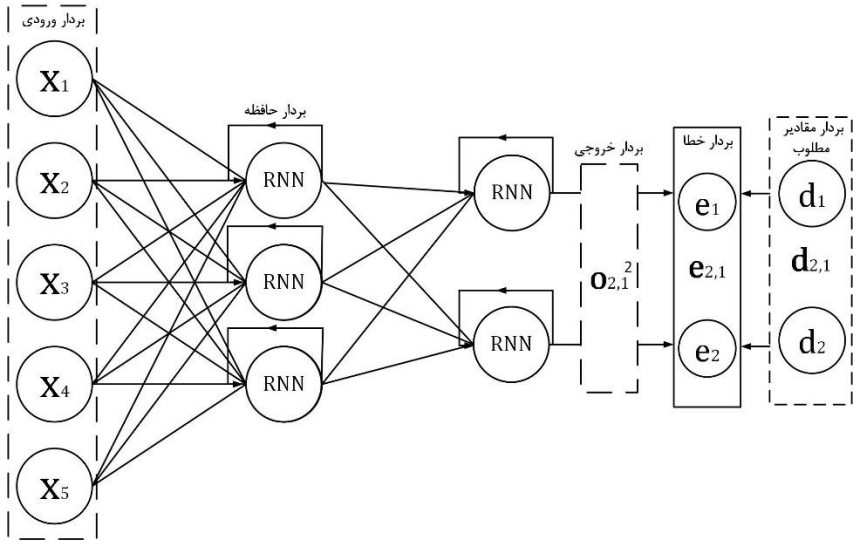
---

[1] Attention mechanism
[2] Natural language processing (NLP)
[3] Optical character recognition (OCR)

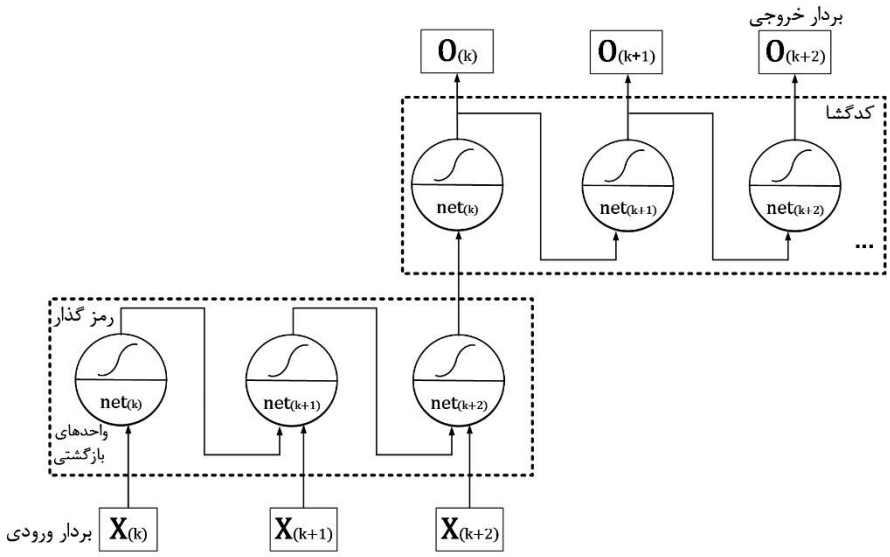
ساختارها در پردازش ورودی هایی با بعد بالا مانند یک جمله از متن است؛ بزرگ تر شدن ابعاد ورودی به ساختاری مشابه ساختار شکل ۱-۱۴ باعث بروز مشکلات مربوط به ابعاد بالای بردار ورودی که در فصل ۷ بررسی شد می شود. برای رفع این معضل از مدل هایی مشابه شکل ۲-۱۴ برای پردازش ورودی هایی با ابعاد بالا که ابعاد مختلف آن نسبت به هم وابستگی داشته باشند استفاده کنیم. مدل هایی مشابه مدل شکل ۲-۱۴ به طور کلی از یک رمزگذار[1] و کدگشا[2] تشکیل شده اند؛ در هر کدام از این دو بخش تعدادی واحد حافظه دار، بازگشتی مانند LSTM و GRU تعریف شده است، نحوه اتصال این واحدهای حافظه دار بدین صورت است که بردار حافظه، $\boldsymbol{h}$، هر واحد وارد واحد کناری می شود؛ این بردار حافظه برای اولین واحد به صورت یک بردار صفر تعریف می شود. در بخش رمزگذار این ساختار هر یک از ابعاد بردار ورودی به یک واحد بازگشتی وارد شده و در نهایت بردار حافظه آخرین واحد به عنوان خروجی نهائی بخش رمزگذار وارد بخش کدگشا می شود. در بخش کدگشا نیز به ترتیب خروجی هر واحد که بخشی از بردار خروجی نهائی است محاسبه می شود. به چنین مدل هایی مدل های دنباله به دنباله[3] گفته می شود. کاربرد این مدل ها معمولا در پردازش دادگان متنی و زبانی برای اهدافی مانند ترجمه، پرسش و پاسخ و... است. با ترکیب این مدل ها با ساختارهای پیچشی که در فصل ۱۲ بررسی کردیم می توانیم مدل هایی به عنوان نویسه خوان نوری[4] نیز توسعه دهیم. در کاربرد این مدل برای دادگان متنی بردار ورودی و خروجی به صورت یک جمله تعریف می شوند، هر یک از ابعاد این بردار یکی از واحدهای تشکیل دهنده جمله هستند؛ در این مدل ها معمولا پیش از تعریف جملات به عنوان ورودی مدل عملیات پیش پردازشی مانند ریشه یابی کلمات[5]، برای مثال ریشه کلمات Reading و Reads فعل Read است، و تجزیه[6] جمله به واحدهای تشکیل دهنده شامل کلمات، علائم نگارشی و... به این جملات اعمال می شود. از این رو به هر یک از بعدهای بردار ورودی توکن[7] گفته می شود. این توکن ها معمولا قبل از ورودی به مدل شکل ۲-۱۴ به برداری مشابه با بردارهای ورودی شبکه های مختلفی که تا کنون بررسی کردیم، تبدیل می شوند، برای تبدیل توکن ها به بردار می توان از یک رابطه نگاشت که هر توکن را به یک فضای ویژگی n-بعدی نگاشت می کند، یک شبکه عصبی و... استفاده کرد، ساختار تبدیل کننده توکن ها به بردار با عنوان Word2Vec شناخته می شود[8]. شبکه عصبی Word2Vec می تواند از پیش آموزش داده شده[9] باشد و یا در حین آموزش مدل دنباله به دنباله، شکل ۲-۱۴ آموزش داده شود. باید توجه داشته باشیم که همانند سایر ساختارهایی که تا کنون بررسی کردیم ابعاد ورودی در مدل های دنباله به دنباله نیز در طول روند آموزش، ارزیابی، تست و

[1] Encoder
[2] Decoder
[3] Sequence to sequence (Seq2Seq)
[4] Optical character recognition (OCR)
[5] Stemming
[6] Tokenization
[7] Token
[8] Word embedding
[9] Pre trained

اســـتفاده از مدل ثابت اســـت، از این رو در صـــورتی که ابعاد یک بردار، جمله، ورودی در مجموعه دادگان بیشتر از بعد ورودی تعریف شده باشد مشکلاتی در روند آموزش ساختار بروز پیدا می کند؛ از این نمونه ورودی یا باید استفاده نشود و یا بخشی از آن را در دو مرحله به عنوان ورودی ساختار اســـتفاده کنیم که این مورد باعث افت عملکرد مدل می شـــود. از این رو تعریف بعد مناســـب برای ورودی در تعریف ساختار مدل با توجه به کاربرد آن و مجموعه دادگان مورد استفاده از چالش های مطرح در این مدل ها است. در کاربرد ذکر شده برای این مدل ها مقادیر مطلوب نیز به صورت یک جمله تعریف می شـــوند، برای مثال در ترجمه هر جمله و ترجمه آن به عنوان ورودی و خروجی مطلوب مدل تعریف می شوند. بر روی مقدار مطلوب نیز پیش پردازش های فوق الذکر اعمال شده و سپس به عنوان مقدار مطلوب در ساختار مدل تعریف می شود. در برخی مدل هایی که در سال های اخیر توسعه داده شده اند نیازی به ریشه یابی در مرحله پیش پردازش نبوده و این بخش با مرحله Word2Vec ادغام شده است.



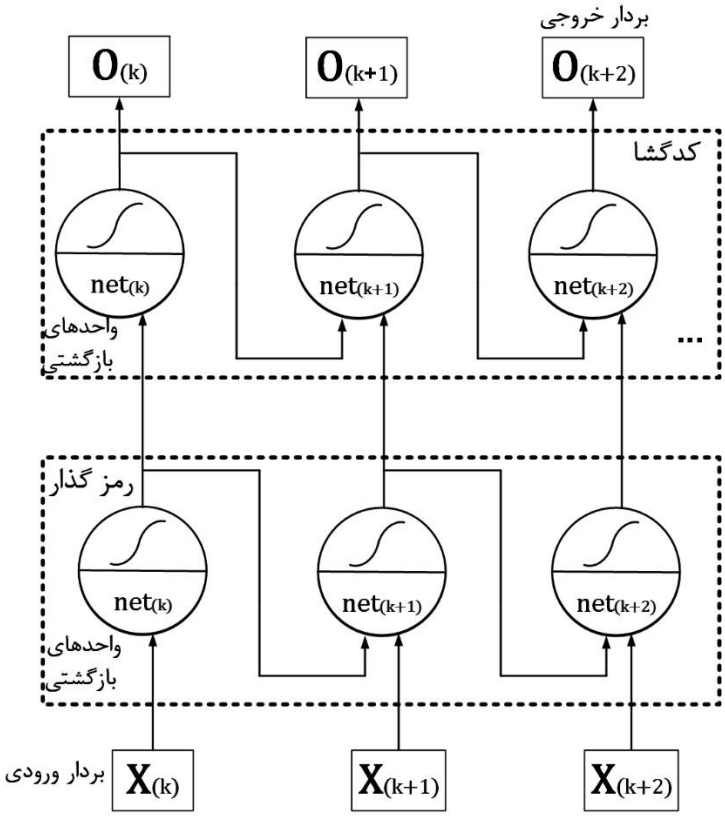
شکل ۱-۱۴: ساختار کلی شبکه های بازگشتی



شکل ۲-۱۴: ساختار کلی مدل دنباله به دنباله

۱۴.۲.۱ معایب مدل های دنباله به دنباله

در مدل هایی مشابه مدل نمایش داده شده در شکل ۲-۱۴ از آنجایی که اتصال دو بخش رمزگذار و کدگشا صرفا از طریق یک بردار، بردار پنهان، حافظه، آخرین واحد رمزگذار تعریف می شود، آموزش ورودی هایی با ابعاد بالا در این ساختارها هموار با چالش هایی مواجه است. برای رفع این معضل ساختارهایی مشابه با ساختار شکل ۳-۱۴ تعریف می شود که در آن ها نسبت به ساختار شکل ۳-۱۴ اتصالات بیشتری بین دو بخش رمزگذار و کدگشا تعریف می شود. مدل های زبانی مانند Bert و GPT از معروف ترین مدل هایی هستند که ساختارهایی شبیه به ساختار شکل ۳-۱۴ دارند. از چالش های مدل هایی مانند مدل ۳-۱۴ نحوه تعریف اتصالات بین رمزگذار و کدگشا است، مکانیزم توجه یکی از رویکردهایی است که برای رفع این چالش توسعه داده شده است. مکانیزم توجه را در ادامه مطالب این فصل بررسی خواهیم کرد. در واحد های قسمت کدگشای مدلی مانند مدل شکل ۳-۱۴ بردار ورودی از قسمت رمزگذار به عنوان بردار ورودی واحد بازگشتی و برداری که از سمت واحد کناری به آن وارد می شود به عنوان بردار حافظه واحد تعریف می شود. در صورتی که این واحدها به صورت واحدهای LSTM تعریف شده باشند هر دو بردار حافظه کوتاه مدت و بلند مدت از سمت واحد کناری وارد خواهند شد.



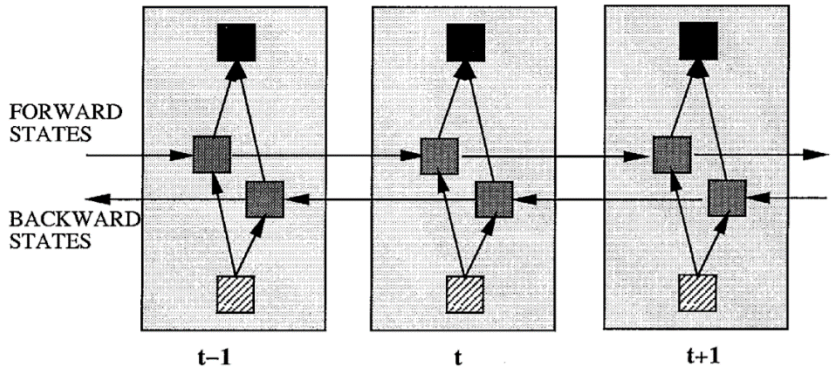
شکل ۳-۱۴: ساختار کلی یک مدل زبانی

۱۴.۲.۲ LSTM دو جهته[1]

در مطالب فصل ۱۱ در بررسی دادگان مربوط به سری های زمانی اشاره کردیم که ابعاد مختلف یک بردار ورودی به یک ساختار بازگشتی نسبت به هم وابستگی زمانی دارند، این وابستگی زمانی ابعاد مختلف همواره نسبت به بعد یا ابعاد پیش از خود تعریف می شود در حالی که در ساختار یک جمله که مدل های زبانی برای پردازش آن ها تعریف می شوند یک کلمه از جمله ممکن است علاوه بر کلمات پیش از خود به کلمات پس از خود نیز وابستگی داشته و معنای آن در ساختار جمله در کنار کلمات پیش و پس از خود تعریف شود. برای اعمال این وابستگی دو طرفه در مدل های زبانی ساختار LSTM دو جهته معرفی شده است، در این ساختار دو دسته واحد حافظه دار به صورت مستقل مطابق شکل ۴-۱۴ تعریف می شود، به طوری که جهت تعریف بردار حافظه واحدهای هر دسته معکوس جهت واحدهای دیگر بوده ولی ورودی هر دو دسته مشترک است. خروجی نهائی این ساختار متشکل از بردارهای حافظه کوتاه مدت، بردار خروجی، هر دو دسته است

[1] Bi-directional LSTM

که در کنار هم قرار داده شده[1] و به صورت یک ماتریس $n \times 2m$؛ n اندازه بردار خروجی هر دسته است، تعریف می شود. بدین ترتیب وابستگی در هر دو جهت در این ساختار اعمال می شود، این ساختار در مدل های زبانی مختلف در قسمت رمزگذار، کدگشا و یا هر دو بخش به کار برده می شود. علاوه بر واحدهای LSTM می توان سایر واحدهای بازگشتی را نیز به صورت دو جهته تعریف کنیم.



شکل ۴-۱۴: ساختار LSTM (بازگشتی) دوجهته [۴]

۱۴.۳ مکانیزیم توجه

در توضیحات مربوط به مدل هایی مانند مدلی که در شکل ۳-۱۴ اشاره کردیم که از عمده چالش های مطرح در تعریف ساختار این مدل ها تعریف اتصالات بین واحدهای مختلف بین دو بخش رمزگذار و کدگشا است. این چالش از آنجایی ناشی می شود که در دادگان متنی و زبانی، برای مثال دو جمله، وابستگی نظیر به نظیر وجود ندارد؛ برای مثال اولین کلمه در هر دو جمله ورودی و خروجی لزوما مربوط به فاعل جمله نیست. این موضوع به ویژه در کاربرد این مدل ها به عنوان مترجم بین دو زبانی که تفاوت های ساختاری زیادی با یکدیگر دارند بیشتر نمود پیدا می کند. بنابراین تعریف این اتصالات به صورت نظیر به نظیر مشابه آنچه که در شکل ۳-۱۴ نشان داده شده است لزوما عملکرد مناسبی در این مدل ها نخواهد داشت. مکانیزم توجه برای رفع این معضل تعریف شده است. از اولین ساختارهای تعریف شده برای مکانیزم توجه، مکانیزم توجه Bahdanau است. در این مکانیزم مطابق شکل ۵-۱۴ میانگین وزنی همه بردارهای حافظه، پنهان، واحدهای قسمت رمزگذار وارد هر یک از واحدهای قسمت کدگشا می شوند. بدین ترتیب بردار ورودی از سمت رمزگذار به هر یک از واحدهای قسمت کدگشا مطابق رابطه ۱-۱۴ محاسبه می شود. در رابطه ۱-۱۴، h_j بردار پنهان واحد j-ام رمزگذار است که وارد واحد i-ام قسمت کدگذار می شود. $\alpha_{i,j}$ نیز وزن مربوط به بردار h_j رمزگذار در بردار ورودی واحد i-ام قسمت کدگذار، c_i، است.

[1] Concatenate

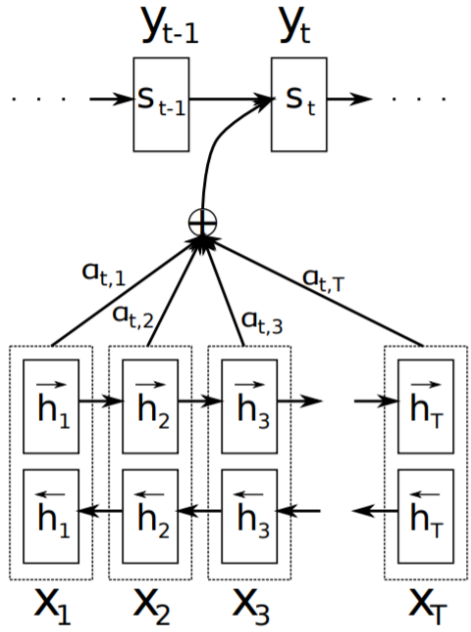
$$c_i = \sum_j \alpha_{i,j} . h_j \tag{۱۴-۱}$$

مقادیر وزن های $\alpha_{i,j}$ در رابطه ۱-۱۴ مطابق رابطه ۲-۱۴ با اعمال تابع بیشینه هموار به مقادیر $e_{i,j}$ محاسبه می شوند.

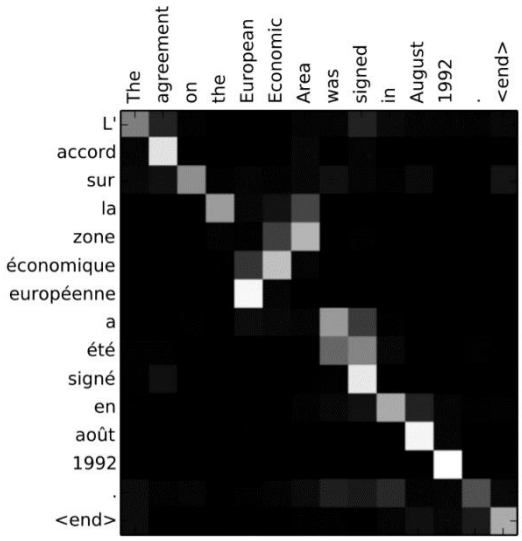
$$\alpha_{i,j} = \text{Softmax}(e_{i,j}) = \frac{e^{e_{i,j}}}{\sum_k e^{e_{i,k}}} \tag{۱۴-۲}$$

در رابطه ۲-۱۴ مقادیر $e_{i,j}$ خروجی یک شبکه عصبی پرسپترون هستند که ورودی آن بردار h_j و بردار خروجی، حافظه، واحد قبلی در قسمت کدگشا، s_{i-1}، مطابق رابطه ۳-۱۴ است. در رابطه ۳-۱۴ منظور از a همان شبکه عصبی است که پارامترهای وزن آن در کنار سایر پارامترهای مربوط به مدل آموزش داده می شوند. بدین ترتیب با اعمال مکانیزم توجه به هر یک از واحدهای جمله ورودی، بردارهای پنهان متناظر با هر یک از واحدهای جمله ورودی، نسبت به هر یک از واحدهای جمله خروجی وزنی نسبت داده می شود که اگر همه آن ها را در کنار هم رسم کنیم ماتریسی مطابق شکل ۶-۱۴ حاصل می شود. در ادامه مکانیزم های توجهی دیگری نیز بر اساس مکانیزم Bahdanauتوسعه داده شده اند که مکانیزم Loung از مهم ترین آن ها است، این مکانیزم ها عموما در نحوه محاسبه وزن های α با هم تفاوت دارند.

$$e_{i,j} = a(s_{i-1}, h_j) \tag{۱۴-۳}$$



شکل ۵-۱۴: ساختار مکانیزم توجه Bahdanau [۱۳۷]



شکل ۶-۱۴: ماتریس خروجی مکانیزم توجه BAHDANAU [۱۳۷]

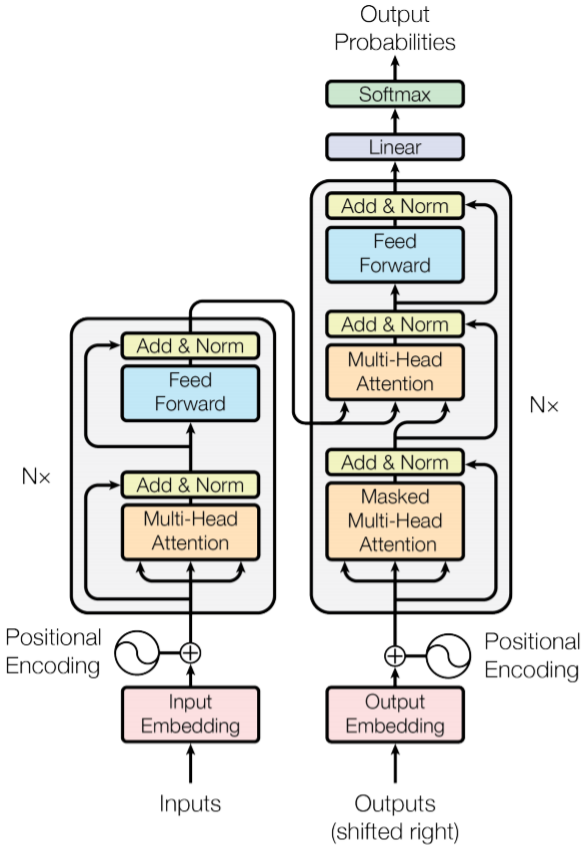
۱۴.۳.۱ مکانیزم خود محور[1]

اساس کار مکانیزم توجهی که در مطالب فوق بررسی کردیم بدین صورت است که به هر یک از

[1] Self-Attention mechanism

واحدها، کلمات، جمله ورودی نسبت به واحدهای جمله خروجی وزنی نسبت داده می شود، اجرای این مکانیزم همواره نیازمند محاسبه واحدهای جمله خروجی است، طبق رابطه ۳-۱۴ در این مکانیزم صرفا بخشی از واحدهای جمله خروجی، واحدهایی که پیش تر محاسبه شده اند، در روند محاسبه وزن دخیل هستند. این موضوع به ویژه در محاسبات مربوط به اولین واحد خروجی چالش برانگیز است. برای رفع این معضل مکانیزم توجه خود محور معرفی شده است. منظور از مکانیزم توجه خود محور نسبت دادن وزن به واحدهایی از یک جمله با توجه به سایر واحدهای تعریف شده در آن جمله است؛ برای مثال در یک جمله واحد هایی مانند حروف ربط و... دارای محتوای معنایی خاصی از جمله نیستند در حالی که فاعل یا فعل جمله حاوی مفاهیم معنایی بیشتری از جمله هستند. بدین ترتیب باید وزن واحدهایی مانند فعل و فاعل در جمله بیشتر از سایر باشد. مکانیزم توجه خود محور در ابتدا در ساختار نگاشت کننده ای[1] مطابق شکل ۷-۱۴ تعریف شده است. از ساختار شکل ۷-۱۴ همانند مکانیزم توجه Bahdanau برای محاسبه وزن بین واحدهای جمله ورودی و خروجی استفاده می شود. جمله خروجی وارد قسمت کدگشا و جمله ورودی وارد قسمت رمزگذار شده و بر روی هر کدام از آن ها مطابق شکل ۷-۱۴ چند عملیات توجه خود محور اعمال می شود. در این ساختار برای تبدیل کلمات به متن از یک رابطه مثلثاتی استفاده می شود. همچنین در این ساختار علاوه بر تبدیل کلمه به متن از یک رابطه مثلثاتی دیگر برای در نظر گرفتن محل هر واحد[2] در جمله و اضافه کردن آن به صورت مقادیر برداری به بردار مرحله قبل نیز استفاده می شود. در نهایت خروجی این ساختار نیز ماتریسی مشابه ماتریس شکل ۶-۱۴ خواهد بود.

[1] Transformer
[2] Position embedding



شکل ۷-۱۴: ساختار نگاشت کننده با مکانیزم توجه خود محور[۵]

مکانیزم توجه خود محور در این ساختار به دو صورت Dot-Product و Multi Head تعریف می شود که به ترتیب در شکل های ۸-۱۴ و ۹-۱۴ نمایش داده شده است. در واقع مکانیزم Dot-Product در داخل مکانیزم Multi Head تعریف می شود. وظیفه مکانیزم Multi Head اعمال وزن های مناسب به ورودی و ایجاد ورودی جستار[۱]، Q، کلید[۲]، K، و مقدار[۳]، V، است. همان طور که در ساختار شکل ۷-۱۴ نیز نشان داده شده است ورودی مکانیزم Multi Head در دو بخش ساختار از یک بردار ورودی مشترک به قسمت رمزگذار و کدگشا و در یک بخش ورودی مربوط به، کلید، K، و مقدار، V، از بردار خروجی رمزگذار و جستار، Q، از قسمت کدگشا تعریف می شود. این روند مشابه روند روند تعریف شده در ساختار شبکه هاپفیلد مدرن فصل ۱۱ و مکانیزم توجه مبتنی بر مکان در ساختارهای پیچشی فصل ۱۲ است. روند اعمال مکانیزم توجه خود محور به ترتیب ذیل است:

[1] Query
[2] Key
[3] Value

۱. ابتدا در مکانیزم Multi Head ماتریس های وزن در بردار ورودی مطابق رابطه ۴-۱۴ اعمال می شوند؛ در اینجا فرض می کنیم هر سه ورودی جستار، کلید و مقدار از یک بردار مشترک تعریف می شوند، همچنین ابعاد سه بردار جستار، کلید و مقدار نیز با هم برابر فرض شده است.

$$K_i = W^K_{i(d_K \times n)} \times x_{n\times 1}, V_i = W^V_{i(d_V \times n)} \times x_{n\times 1}, Q_i = W^Q_{i(d_Q \times n)} \times x_{n\times 1}$$
$$d_K = d_V = d_Q \qquad (۱۴-۴)$$

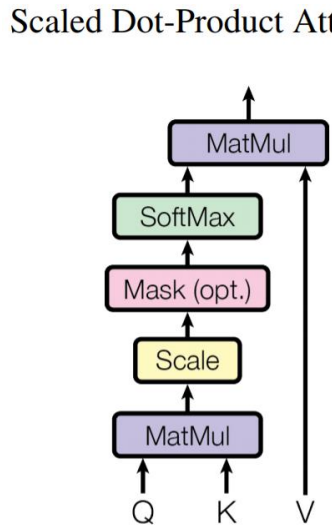
۲. سپس بردارهای جستار، کلید و مقدار حاصل از مرحله قبل وارد مکانیزم Dot-Product می شوند. در این مکانیزم به سه بردار ورودی رابطه ۵-۱۴ اعمال می شود. در رابطه ۵-۱۴ منظور از d_k ابعاد بردار کلید است که برای جلوگیری از ایجاد مقادیر بسیار کوچک در مواردی که ابعاد بردارهای کلید و مقدار بزرگ باشد به صورت یک پارامتر کنترلی اسکالر اعمال می شود.

$$head_i = \text{DotProduct}(K_i, V_i, Q_i) = \text{softmax}(\frac{Q_i K_i^T}{\sqrt{d_k}}) V_i \qquad (۱۴-۵)$$

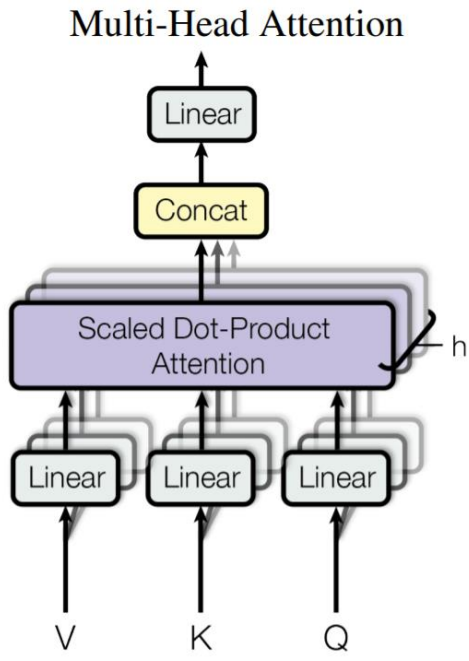
۳. در مکانیزم Multi Head شکل ۹-۱۴ منظور از h تعداد لایه های توجه است که در نهایت خروجی نهائی مکانیزم توجه با کنار هم قرار دادن[1] خروجی لایه های مختلف، head، و اعمال یک ماتریس وزن به آن ها طبق رابطه ۶-۱۴ تعریف می شود.

$$MultiHead(x) = Concat(head_1, ..., head_h)_{1\times hd_v} \times W^O_{hd_v \times n} \qquad (۱۴-۶)$$

[1] Concatenate



شکل ۸-۱۴: ساختار مکانیزم توجه خود محور Dot-Product [۵]



شکل ۹-۱۴: ساختار مکانیزم توجه خود محور Multi Head [۵]

۱۴.۴ مسائل

۱. در مطالب مربوط به چالش های مدل های دنباله به دنباله اشـاره کردیم که همواره ابعاد ورودی به گونه ای انتخاب می شـــود که بزرگ ترین جمله، بردار، ورودی توســـط مدل پوشـش داده شـود، تحقیق کنید که علاوه بر این روش چه رویکردهای دیگری نیز برای رفع این چالش به کاربرده می شود.

۲. در مورد مفاهیم مکانیزم توجه محلی[۱] و سـراسـری[۲] و همچنین مکانیزم توجه سخت[۳] و نرم[۴] تحقیق کنید.

[1] Local attention
[2] Global attention
[3] Hard attention
[4] Soft attention

فصل ۱۵
تعمیم دامنه[1] و یادگیری انتقالی[2]

۱۵.۱ مقدمه

همواره تهیه مجموعه دادگان آموزشی مناسب برای یک ساختار عمیق که دارای برچسب مقادیر مطلوب بوده و دامنه گسترده ای از ورودی را که انتظار می رود مدل برای آن پاسخ مناسبی داشته باشد را پوشش دهد، از چالش های مطرح در این حوزه است. برای کاهش وابستگی به مجموعه دادگان و رفع چالش های مربوط به آن رویکرد های مختلفی برای دستیابی به روند آموزش مناسب برای ساختار تعریف شده با استفاده از مجموعه دادگانی که دارای تعداد نمونه های کمتری نسبت به مسئله تعریف شده، مطرح شده است؛ هدف اصلی این روش ها تعمیم دامنه مدل با استفاده از مجموعه دادگانی با حداقل تعداد نمونه و یا مجموعه دادگانی متفاوت از مسئله تعریف شده است، یادگیری انتقالی و روش های یادگیری N-shot[3] از مهم ترین رویکردهای مطرح در این حوزه هستند که در ادامه این فصل به بررسی آن ها خواهیم پرداخت.

۱۵.۲ تعریف کلی یادگیری انتقالی

رویکرد یادگیری انتقالی برای مسائلی به کار برده می شود که در آن ها تعداد نمونه های مجموعه دادگانی که دارای برچسب مقادیر مطلوب مناسب برای آموزش مدل محدود است، اما در عوض مجموعه دادگان متفاوتی دیگری که ساختار نمونه های آن با مدل تعریف شده سازگار است، از نظر نوع نمونه ها و...، در دسترس است. در این صورت برای اعمال یادگیری انتقالی ابتدا مدل را با استفاده از مجموعه دادگانی که تعداد نمونه های بالائی دارد آموزش می دهیم؛ از آنجائی که معمولا تعداد کلاس و متعاقبا ابعاد خروجی دو مجموعه دادگان ذکر شده با هم متفاوت است، برای آموزش مدل با مجموعه دادگان با تعداد نمونه های بالا تغییراتی در ساختار مدل، معمولا در لایه های انتهائی، اعمال می کنیم، در صورتی که ابعاد ورودی دو مجموعه متفاوت باشد معمولا تغییر به نمونه ها اعمال می شود؛ برای مثال در یک ساختار پیچشی فرضی در صورتی که ابعاد تصاویر

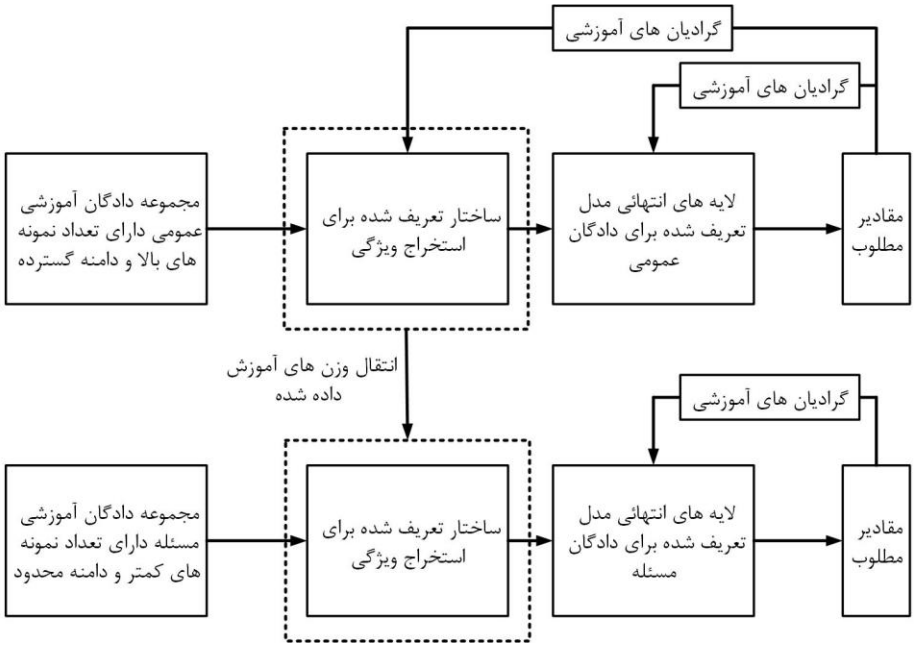
---

[1] Domain adoption
[2] Transfer learning
[3] N-Shot learning

ورودی مجموعه دادگانی که تعداد نمونه های زیادی دارد متفاوت از ابعاد تصـــاویر مجموعه دادگان اصـــلی باشـــد، ابعاد تصـــاویر را تغییر می دهیم. پس از اتمام روند آموزش ســـاختار مدل، لایه های انتهائی، را متناسب با دادگان مسئله اصلی تغییر می دهیم، برای وزن های لایه هایی که دچار تغییر شده اند را با استفاده از روش هایی که در فصل های گذشته بررسی کردیم مقادیر اولیه در نظر می گیریم، برای مقادیر وزن های سـایر لایه ها مقادیر حاصـل از آموزش با مجموعه دادگان دارای نمونه های بالا را در نظر می گیریم؛ به این وزن ها، وزن های از پیش آموزش داده شـده[1] اطلاق می شـود. در این مرحله از آموزش معمولا آموزش صـــرفا برای وزن های لایه های انتهائی که مقادیر اولیه تعریف شـده و مقادیر وزن های لایه هایی که از مرحله قبل حاصـل شـده اسـت بدون تغییر باقی می مانند؛ البته می توان این وزن ها را نیز مجددا با اسـتفاده از مجموعه دادگان مسـئله، دادگان با تعداد نمونه کمتر، در کنار وزن های لایه های انتهائی آموزش دهیم[2].

در صورتی که دامنه مجموعه دادگان دارای تعداد نمونه های بالا به حد کافی گسـترده باشد می توان با اسـتفاده از آن مدلی با سـاختار مناسـب که عملکرد مطلوبی نسـبت به آن مجموعه دادگان داشـــته باشـــد را آموزش داده و با حذف یک یا چند لایه انتهائی آن پس از اتمام روند آموزش یک مدل استخراج کننده ویژگی توسعه دهیم، بدین ترتیب ساختار حاصل یک مدل بازنمائی خواهد بود که به صورت با نظارت آموزش داده شده است. از این مدل بازنمائی می توانیم برای استخراج ویژگی از دادگانی که دارای ساختار و جنسی مشابه با ساختار و جنس دادگان آموزشی اولیه است استفاده کرده و سـپس بردار ویژگی حاصـل را به عنوان ورودی برای سـاختارهای دیگر، مشـابه روند تعریف شده در ساختار خودرمزگذارها که در فصل ۱۰ معرفی کردیم، در نظر بگیریم، این رویکرد چنان که در ادامه این بخش بررســی خواهیم کرد در ســاختارهایی مانند ســاختارهای پیچشــی و مدل های زبانی بسیار مورد توجه است. شکل ۱-۱۵ روند کلی یادگیری انتقالی را نشان می دهد.

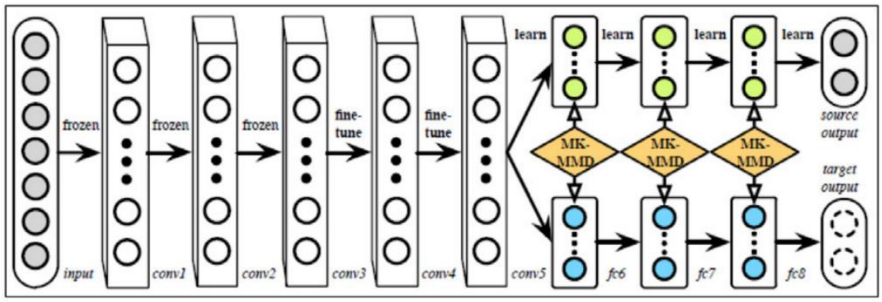
[1] Pre-trained weights
[2] Fine tuning



شکل ۱-۱۵: روند کلی یادگیری انتقالی

در روند یادگیری انتقالی برای تعمیم بهتر دامنه ساختار آموزش داده شده با مجموعه دادگانی که تعداد نمونه های کمتری دارند می توان از رویکرد شبکه های تطبیق دامنه[1] نیز استفاده کرد. در ساختار شبکه های تطبیق دامنه مطابق شکل ۲-۱۵ هدف کاهش تفاوت در مقادیر بردارهای خروجی یک یا چند لایه پنهان انتهایی از ساختار به ازای ورودی های دو مجموعه دادگان اولیه و هدف، مجموعه دادگان با تعداد نمونه های کمتر، است؛ بدین ترتیب در طی روند آموزش، ساختار می تولند بردارهای ویژگی برای ورودی های مجموعه دادگان با تعداد نمونه های کمتر ایجاد کند که نسبتا مشابه با بردارهای ویژگی مجموعه اولیه هستند؛ بدین ترتیب وزن های لایه های اولیه ساختار که با مجموعه دادگان اولیه آموزش داده شده اند تطبیق پذیری بهتری با مجموعه دادگان هدف و لایه های انتهایی که مجددا آموزش داده می شوند، خواهند داشت. برای به کارگیری این رویکرد یک معیار تشابه مانند معیار MMD بین مقادیر بردارهای پنهان ورودی دو مجموعه مختلف به تابع هزینه مربوط به مجموعه دادگان با تعداد نمونه کمتر، مجموعه هدف، در مرحله آموزش مجدد لایه های انتهایی اضافه می شود. شکل ۲-۱۵ روند کلی یک شبکه تطبیق دامنه و نحوه تعریف معیار تشابه را نمایش می دهد.

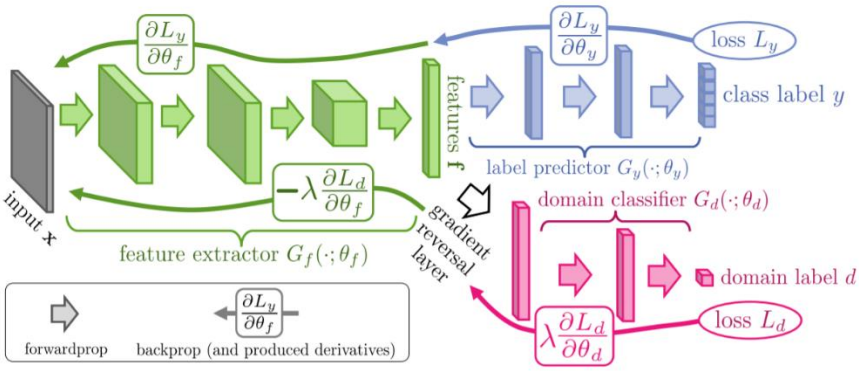
[1] Deep adaption network (DAN)



شکل ۲-۱۵: روند کلی ساختار شبکه های تطبیق دامنه[۱۸۵]

در مواردی که دو مجموعه دادگان متفاوت از یک جنس و ابعاد که دارای دامنه و مقادیر مطلوب متفاوت هستند موجود باشد می توان ساختاری مشابه شکل ۳-۱۵ تعریف کرده و همزمان از گرادیان های حاصل از هر دو مجموعه برای آموزش مدل استخراج ویژگی، مدل بازنمائی، استفاده کرد. به ساختارهای مشابه ساختار شکل ۳-۱۵ شبکه های دامنه متخاصم[1] گفته می شود. در ساختار شکل ۳-۱۵ مقادیر مطلوب مربوط به هر دو مجموعه در کنار هم ادغام شده و گرادیان های آن در مسیر بالایی تعریف می شود، مسیر پایین نیز به صورت یک مسئله طبقه بندی برای تشخیص دامنه، تعلق ورودی به مجموعه دادگان، تعریف می شود. در این ساختار گرادیان های حاصل از هر دو مسیر لایه های مربوط به آن خود را به روش گرادیان نزولی آموزش می دهد، برای آموزش مدل استخراج کننده ویژگی، بخش مشترک ساختار برای هر دو مسیر، قرینه گرادیان های مسیر پایین، پس از ضرب در پارامتر کنترلی λ، با گرادیان های مسیر بالایی جمع شده و پارامترهای این بخش با استفاده از مقدار مجموع گرادیان های دو مجموعه داده به روش گرادیان نزولی به روزرسانی می شوند. بدین ترتیب در این رویکرد از بیش برازش مقادیر پارامترهای ساختار استخراج ویژگی بر روی دامنه یک مجموعه داده جلوگیری شده و مدل حاصل برای طیف گسترده تری از مسائل در مقایسه با روش مرسوم نشان داده شده در شکل ۱-۱۵ عملکرد مطلوبی خواهد داشت.

[1] Domain adversarial neural networks (DANNs)



شکل ۳-۱۵: ساختار شبکه های دامنه متخاصم[۱۲۴]

۱۵.۲.۱ یادگیری انتقالی در ساختارهای پیچشی[1]

در ساختارهای پیچشی با توجه به مشکلات تهیه مجموعه دادگان آموزشی، یادگیری انتقالی و ایجاد مدل های بازنمائی عمومی که برای طیف گسترده ای از مسائل قابل استفاده باشد اهمیت بیشتری دارد. روند یادگیری انتقالی در مدل های پیچشی بدین صورت است که ابتدا با استفاده از یک مجموعه دادگان گسترده، معمولا مجموعه ImageNet که دارای گستره وسیعی از کلاس های مختلف دادگان تصویری است، یک ساختار پیچشی به عنوان یک مدل بازنمائی آموزش داده می شود؛ با توجه به این که در لایه های مختلف ساختارهای پیچشی همواره ویژگی هایی استخراج می شود که برای انواع مختلف داده های تصویری مشترک است، مانند ویژگی های سطح پایینی مانند لبه های تصویر در لایه های ابتدایی و ویژگی های سطح بالاتری که بیانی کلی از اجسام داخل تصویر ارائه می کنند در لایه های انتهایی، از یک مدل پیچشی با ساختار مناسب می توان به عنوان یک مدل بازنمائی مشترک برای انواع مختلف دادگان تصویری استفاده کرد. وزن های این مدل بازنمائی به صورت از پیش آموزش داده شده بر اساس دادگان اولیه استفاده شده و در کاربردهای آن در مسائل مختلف مقادیر آن ها تغییری نمی کند؛ بلکه آموزش صرفا محدود به ساختاری است بردار ویژگی حاصل از این مدل بازنمائی به عنوان ورودی آن در نظر گرفته می شود.

۱۵.۲.۱.۱ شبکه Big Transfer

این مدل که در سال ۲۰۲۰ میلادی معرفی شده است، یک ساختار پیچشی مرسوم است که در آن از لایه های پیچشی و تجمیع متنوعی بهره گرفته شده است، تعداد لایه ها و پارامترهای آموزشی در این ساختار به گونه ای است که می تواند عملکرد مطلوبی را برای مسائل مختلف با دادگان ورودی تصویر تضمین کند؛ بدین صورت که تعداد لایه های ساختار به توجه به حجم دادگانی که برای آموزش اولیه آن در نظر گرفته شده است، حدود ۱۴ میلیون نمونه، در تعداد بالا

[1] CNNs

تعریف شده است. از وزن های این ساختار به عنوان یک مدل استخراج ویژگی استفاده شده و در مسائل مختلف لایه هایی نسبت به کاربرد به انتهای آن اضافه می شود. آموزش تعریف شده برای دادگان هر مسئله صرفا محدود به لایه های افزوده شده صورت گرفته و مقادیر وزن های لایه های ساختار استخراج ویژگی به صورت از پیش آموزش داده شده استفاده می شوند.

۱۵.۲.۲ یادگیری انتقالی در مدل های زبانی[1]

تهیه مجموعه دادگان آموزشی مناسب در مدل های زبانی نیز همواره از چالش های مطرح است. با توجه به استفاده از ساختارهای کدکننده کلمات در ساختار مدل های زبانی می توان یک مدلی را که نیازمند دادگان آموزشی زیادی است مانند مدل Bert، در ابتدا با استفاده از دادگان مناسب آموزش داده و در مثال مختلف وزن های حاصل شده از دادگان گسترده تر را به عنوان وزن اولیه در نظر گرفته و آن ها را مجددا آموزش دهیم. با توجه به شباهت استخراج ویژگی از ورودی متون در مسائل مختلف، در این مدل ها نیز مشابه ساختارهای پیچشی آموزش مجددا می تواند صرفا محدود به لایه های انتهائی باشد.

۱۵.۳ یادگیری N-Shot

در مسائل طبقه بندی یادگیری انتقالی که پیش تر در این فصل بررسی کردیم صرفا زمانی کاربرد دارد که به ازای همه کلاس های تعریف شده در مسئله نمونه هایی در مجموعه دادگان مسئله، مجموعه دادگان با تعداد نمونه کمتر که آموزش چند لایه انتهائی پس از آموزش مدل استخراج ویژگی بر اساس آن انجام می شود، موجود باشد؛ در صورتی که کلاسی در مسئله تعریف شده باشد ولی نمونه ای به ازای آن در مجموعه دادگان مسئله وجود نداشته باشد روند آموزش با استفاده از مجموعه دادگان مسئله که با آن صرفا لایه های انتهائی بعد از مدل بازنمائی آموزش داده می شوند دچار اختلال شده و مدل آموزش داده شده در نهایت نمونه های ورودی کلاسی را که به ازای آن نمونه ای در مجموعه دادگان موجود نیست را به درستی تشخیص نمی دهد، این اختلال در روند آموزش در مواردی که در مجموعه دادگان مسئله به ازای همه کلاس های تعریف شده نمونه هایی موجود باشند ولی تعداد نمونه های کلاس های مختلف با هم برابر نبوده و اختلاف قابل توجهی نسبت به هم داشته باشند[2] نیز به وجود می آید. برای مرتفع کردن این معضل در مسائل طبقه بندی که در آن ها به ازای یک یا چند کلاس تعریف شده نمونه ای در مجموعه دادگان موجود نباشد و یا تعداد نمونه های آن بسیار کمتر از تعداد نمونه های سایر کلاس ها باشد از رویکرد آموزش N-Shot استفاده می شود؛ منظور از N تعداد نمونه های یک کلاس است. رویکرد آموزشی N-Shot معمولا به صورت Zero-Shot، One-Shot، Few-Shot تعریف می شود که در آن

[1] NLP models
[2] Unbalanced dataset

ها تعداد نمونه های کلاس مورد نظر در مجموعه دادگان مسئله به ترتیب صفر، یک و چند، کمتر از پنج، نمونه است. البته در مواردی که تعداد نمونه های تعریف شده برای یک کلاس کمتر از سایر کلاس ها باشد علاوه بر رویکرد آموزش N-Shot می توانیم با استفاده از رویکردهایی نظیر افزایش دادگان[1] که در فصل ۱۲ برای ساختارهای پیچشی بررسی کردیم و یا شبکه های مولد که در فصل ۱۳ معرفی کردیم تعداد نمونه های کلاس مورد نظر را افزایش دهیم، هر یک از این رویکردها با توجه به تعریف مسئله می تواند باعث بهبود آموزش و عملکرد مدل شود.

در آموزش N-Shot هدف کلی آموزش مدلی برای نگاشت نمونه های مختلف ورودی به یک نقطه در فضای ویژگی تعریف شده توسط مدل است؛ به طوری که هر نمونه ورودی متناظر با یک نقطه منحصر به فرد در فضای ویژگی باشد. بعد فضای ویژگی برابر با بعد خروجی مدل تعریف شده خواهد بود. در روند آموزش مدل تابع هزینه ساختار به گونه ای تعریف می شود تا فاصله نقاط متناظر با نمونه های ورودی هر کلاس در فضای ویژگی به هم، به نقطه مرکز آن کلاس، نسبت به سایر نقاط نزدیک تر باشند. در این رویکرد برای هر کلاس یک نقطه مرکز[2] در فضای ویژگی تعریف می شود، نحوه محاسبه مختصات این نقطه در حالت های مختلف بدین شرح است:

- **آموزش Few-Shot:** مختصات نقطه مرکز هر کلاس برابر با میانگین مختصات همه نمونه های تعریف شده برای آن کلاس در مجموعه دادگان آموزشی است.

- **آموزش One-Shot:** مختصات نقطه مرکز کلاس برابر با مختصات نقطه متناظر با تنها نمونه ای است که برای کلاس مد نظر تعریف شده است؛ نقطه مرکز سایر کلاس ها که دارای بیش از یک نمونه هستند مشابه حالت آموزش Few-Shot تعریف می شود.

- **آموزش Zero-Shot:** برای کلاسی که در مجموعه دادگان آموزشی نمونه ای برای آن تعریف نشده است، مرکز دسته برابر با نقطه متناظر با فراداده[3] آن کلاس در فضای ویژگی تعریف می شود، برای تعیین مرکز سایر کلاس های مسئله که به ازای آن ها در مجموعه دادگان آموزشی نمونه هایی تعریف شده است می توان هم از فراداده آن کلاس ها و هم از رویکرد مشابه در حالت آموزش Few-Shot استفاده کرد، البته بهتر است برای مرکز این کلاس ها نیز از فراداده استفاده کنیم تا شرایط تعریف شده برای همه کلاس های مسئله برابر باشد. در ادامه این بخش به بررسی نحوه تعیین نقطه متناظر با فراداده یک کلاس خواهیم پرداخت.

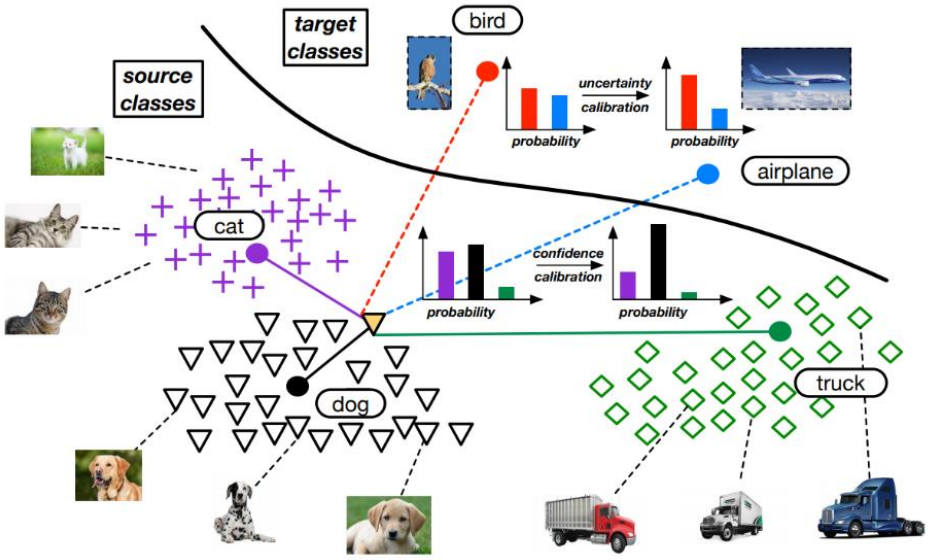
با این توضیحات در این روش شکل کلی تابع هزینه برای آموزش مدلی که نمونه های ورودی را به فضای ویژگی نگاشت می کند به صورت رابطه ۱-۱۵ تعریف می شود. در این رابطه منظور از c_k

[1] Data augmentation
[2] Centroid
[3] Metadata

نقطه مرکز کلاس k در فضای ویژگی، $\acute{k}$ سایر کلاس های تعریف شده در مسئله به جز کلاس k، $f(x)$ مدل تعریف شده برای نگاشت ورودی x_k به فضای ویژگی، نمونه ورودی x_k متعلق به کلاس k است، $d(.,.)$ فاصله، برای مثال فاصله اقلیدسی، بین دو نقطه در فضای ویژگی و N تعداد نمونه های مجموعه دادگان آموزشی است. در این روش برخلاف روش طبقه بندی مرسوم از بردارهای One hot برای تعریف، برچسب، مقدار مطلوب، نمونه ها استفاده نمی شود.

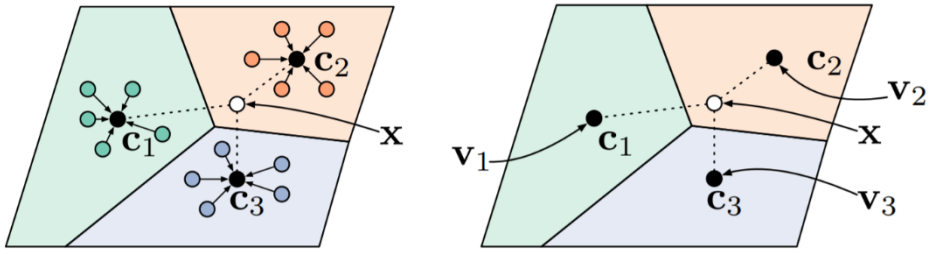
$$E = \frac{1}{N}\left(d(f(x_k), c_k) + \log \sum_{k'} e^{-d(f(x_k), c_{k'})}\right) \quad (۱۵-۱)$$

شکل ۴-۱۵ ساختار کلی فضای ویژگی تعریف شده در روش آموزش Zero-Shot را نشان می دهد، در این شکل منظور از Source Class، کلاس هایی از مسئله است که به ازای آن ها در مجموعه دادگان آموزشی تعدادی نمونه وجود داشته و از آن ها برای آموزش مدل استخراج ویژگی استفاده می شود، Target Class نیز کلاس هایی از مسئله است که به ازای آن ها در مجموعه دادگان آموزشی نمونه ای وجود ندارد. همان طور که در شکل ۴-۱۵ نیز مشخص شده است تعلق هر نمونه ورودی در این روش با استفاده از فاصله آن از مراکز کلاس های مختلف تعیین می شود.



شکل ۴-۱۵: فضای ویژگی رویکرد آموزش N-Shot][

نحوه تعیین مراکز کلاس ها در دو حالت Few-Shot و Zero-Shot نیز به ترتیب در شـکل های ۵-۱۵ الف و ۵-۱۵ ب نشان داده شده است.

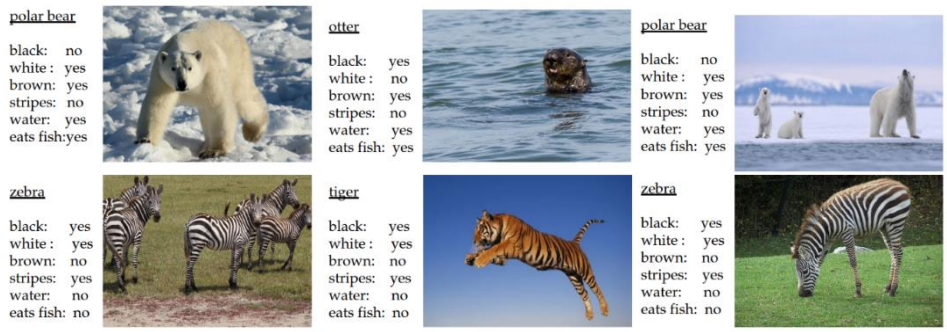


الف ب

شکل ۵-۱۵: تعیین نقطه مرکز کلاس ها در فضای ویژگی[۱۶۱]

۱۵.۳.۱ تعیین مراکز کلاس ها با استفاده فراداده

فراداده برای یک کلاس در رویکرد آموزش N-Shot به صـورت یک نقطه در فضـای ویژگی و یا اطلاعاتی که می توان آن ها را به یک نقطه در فضای ویژگی تبدیل کرد تعریف می شـود؛ منظور از فضـای ویژگی همان فضـایی اسـت که هدف از آموزش مدل نگاشـت نمونه های ورودی به آن فضـا اسـت. فراداده می تواند به صـورت ویژگی های توصـیفی از کلاس مربوط تعریف شـود. شـکل ۶-۱۵ فراداده چند کلاس مربوط به دادگان تصـویری را نشـان می دهد. در صـورتی که هر یک ابعاد فضـای ویژگی به صورت نظیر به نظیر با فراداده تعریف شده باشد؛ برای مثال فضای ویژگی و فراداده هر دو شـامل دو بعد ابعاد ظاهری و وزن تعریف شـده باشـند می توان فراداده هر کلاس را مسـتقیما معادل یک نقطه در فضای ویژگی در نظر گرفت، در غیر این صـورت باید برای نگاشـت فراداده به یک نقطه در فضـای ویژگی از یک مدل نگاشـت کننده اسـتفاده کنیم؛ برای مثال می توان از یک مدل زبانی برای تبدیل فراداده به یک بردار، نقطه، در فضای ویژگی استفاده کنیم.



شکل ۶-۱۵: فراداده کلاس های مختلف برای دادگان تصویری][

۱۵.۴ مسائل

۱. تاثیر انتخاب مجموعه دادگان اولیه در یادگیری انتقالی در عملکرد مدل نهائی که برای مسئله تعریف شده است را بررسی کنید.

۲. در مورد ساختار پیچشی و مدل های زبانی مرسومی که با وزن های از پیش آموزش داده شده به روش یادگیری انتقالی استفاده می شوند تحقیق کرده و عملکرد آن ها را با هم مقایسه کنید.

۳. برای یک مدل ثابت تاثیر آموزش مجدد وزن های مدل استخراج ویژگی، مدل بازنمائی، با استفاده از مجموعه دادگان آموزشی مسئله را بررسی کنید.

فصل ۱۶
رویکردهای جستجوی ساختار عصبی[1]

۱۶.۱ مقدمه

در ساختارهای مختلف انتخاب پارامترهایی مانند تعداد لایه، تعداد نورون ها در لایه های مختلف، نوع توابع فعال ساز و...، که با عنوان فراداده[2] ساختار شناخته می شوند، به طوری که ساختار عملکرد مناسبی از جنبه های مختلف داشته باشد همواره از چالش های مطرح در مسائل مختلف یادگیری ژرف و به طور کلی یادگیری ماشین است. ساده ترین روش برای تعیین مقادیر این پارامترها، تعیین آن ها به صورت دستی[3] است؛ در این روش مقادیر این پارامترهای ساختار یا به صورت سعی و خطا و یا با انجام تحلیل هایی کلی از عملکرد مقادیر و حالت های مختلف این پارامترها تعریف می شود. در کنار روش دستی روش های آموزشی و رویکردهایی بر پایه روش های مختلف جستجو نیز برای تعیین مقادیر بهینه این پارامترها توسعه داده شده است؛ اصول عملکرد اکثر این روش ها معمولا بر اساس ابرشبکه ها[4] تعریف می شود. در این فصل به معرفی و بررسی روش های آموزشی و جستجوی مختلفی که بر اساس ابرشبکه ها تعریف می شوند پرداخته و در مورد مزایا و معایب هر کدام از آن ها بحث می کنیم. در انتهای فصل نیز مسائل مربوط به منظور آشنائی بیشتر با مطالب ارائه شده است.

۱۶.۲ رویکردهای تعیین ساختار عصبی بر پایه یادگیری

از مرسوم ترین رویکردهای آموزشی که برای تعیین ساختار بهینه عصبی استفاده می شود رویکردهای مبتنی بر یادگیری تکاملی[5] و یادگیری تقویتی[6] است. هر دوی این رویکردهای آموزشی معمولا بر پایه ابرشبکه ها تعریف می شوند؛ منظور از ابرشبکه، یک شبکه دارای تعداد لایه و تعداد نورون های بسیار گسترده است که ساختار بهینه به صورتی زیرشبکه[7] ای از آن مشتق می شود. بنابراین می توانیم چنین برداشت کنیم که هدف از این رویکردهای آموزشی بررسی زیرشبکه های

---

[1] Neural architecture search (NAS)
[2] Metadata
[3] Manual
[4] Super networks
[5] Evolutionary learning
[6] Reinforcement learning
[7] Subnetwork

مختلف مشتق شده از ابرشبکه و تعیین زیرشبکه بهینه از نقطه نظرهای مختلف عملکردی مانند حجم محاسبات، خطای خروجی و... است. یکی از مهم ترین رویکردهای مطرح در حوزه یادگیری تکاملی که از آن برای تعیین ساختار مناسب عصبی، در مواردی تعیین وزن های اولیه ساختار عصبی، استفاده می شود الگوریتم ژنتیک[1] است که در ادامه این بخش به بررسی آن می پردازیم، سایر الگوریتم های یادگیری تکاملی که عموما بر پایه مباحث هوش جمعی و هوش جانوری تعریف می شوند و نیز رویکردهای مبتنی بر یادگیری تقویتی برای تعیین ساختار بهینه عصبی خارج از مباحث این فصل است.

۱۶.۲.۱ الگوریتم ژنتیک

الگوریتم ژنتیک یکی از روش های مطرح در حوزه یادگیری تکاملی است که در آن هدف تولید کروموزوم[2] های حاوی مقادیر پارامترهای هدف مسئله و ارزیابی عملکرد این کروموزوم ها در ساختار مسئله با استفاده از یک تابع تناسب تعریف شده است. برای مسئله تعیین تعداد لایه ها و تعداد نورون های یک ساختار عصبی، برای مثال شبکه پرسپترون چند لایه، روند اجرای الگوریتم ژنتیک به شرح ذیل است:

۱. ابتدا یک ابر شبکه دارای تعداد لایه و تعداد نورون های گسترده تعریف می کنیم، وزن های اولیه این ابرشبکه را می توانیم با یکی از روش های تعیین شده در فصل ۷ تعیین کرده و یا برای همه مقادیر وزن ها از مقادیر ثابت استفاده کنیم، در هر صورت باید همواره توجه کنیم که مقادیر این وزن ها در طول اجرا الگوریتم ثابت بوده و تغییری نخواهند داشت.

۲. سپس کروموزوم هایی مشابه شکل ۱-۱۶ تعریف می کنیم؛ هر یک از این کروموزوم ها در اصل یک بردار باینری هستند که ساختار زیرشبکه مشتق شده از ابرشبکه را تعیین می کنند؛ بدین صورت که اندازه این کروموزوم ها، اندازه بردار، برابر با حالت های مختلف قابل تعریف برای ساختار زیرشبکه بوده و هر یک از بعد های مختلف کروموزوم وضعیت یک المان، نورون یا لایه، در ساختار زیرشبکه را تعیین می کند. برای مثال در صورتی که همه ابعاد یک کروموزوم مقدار یک داشته باشد زیرشبکه معادل آن همان ساختار ابرشبکه خواهد بود، با ظاهر شدن مقادیر صفر در هر بعد از کروموزوم المان، نورون یا لایه، متناظر با آن در زیرشبکه حاصل غیرفعال، حذف، می شود. به بیانی دیگر زیرشبکه متناظر با هر کروموزوم ساختاری متشکل از المان های دارای مقادیر یک در ابعاد مختلف کروموزوم است. مقادیر ابعاد مختلف این کروموزوم

[1] Genetic algorithm
[2] Chromosome

ها به صورت تصادفی انتخاب شده و به مجموعه کروموزوم های ایجاد شده در هر مرحله یک جمعیت[1] و به تعداد کروموزوم های هر جمعیت، اندازه جمعیت[2] گفته می شود. اندازه کروموزوم، اندازه بردار، همه جمعیت های تعریف شده در یک مسئله با هم برابر خواهد بود، ولی لندازه جمعیت در جمعیت های مختلف لزوما با هم برابر نیست.

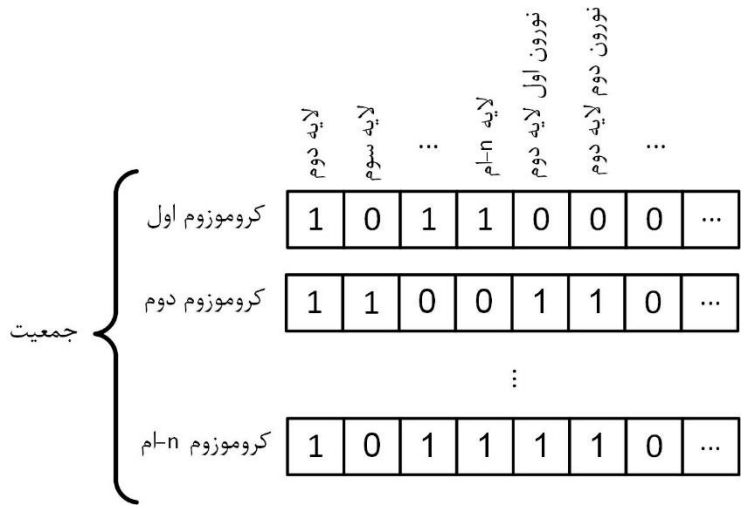
۳. به ازای هر یک از کروموزوم های جمعیت اولیه که در مرحله قبل ایجاد کردیم، زیرشبکه آن را تشکیل داده و با استفاده از نمونه های موجود در مجموعه دادگان عملکرد زیرشبکه را محاسبه می کنیم؛ این روند مشابه روند یک گام[3] ارزیابی در روند شبکه است؛ بدین صورت که به ازای همه نمونه ها مقدار تابع هزینه ساختار را محاسبه کرده و در نهایت مقدار میانگین آن ها را به عنوان مقدار تابع تناسب آن کروموزوم در نظر می گیریم. در یک جمعیت کروموزوم هایی که دارای کمترین مقدار تابع تناسب باشند، ساختار زیرشبکه ها آن ها بهینه است. با توجه به تعریف تابع تناسب، ممکن است در برخی مسائل بهترین عملکرد مربوط به کروموزوم هایی با بیشترین مقدار تابع تناسب باشد.

۴. به جمعیتی که در مرحله قبل عملکرد کروموزم های آن را بررسی کردیم جمیعت والد[4] گفته می شود. در ادامه با اعمال برش[5] و جهش[6] به کروموزوم های جمعیت فرزند[7] را ایجاد کرده و مجددا به ازای کروموزوم های جمعیت فرزندان مشابه مرحله ۳ عملکرد آن ها را می سنجیم. در ادامه نحوه اعمال برش و جهش را بررسی می کنیم.

۵. تولید جمعیت فرزند از جمعیت والد، جمعیت پیش از خود، مشابه روند مراحل ۳ و ۴ ادامه پیدا می کند. در نهایت زیرشبکه متناظر با کروموزومی که در بین همه جمعیت های ایجاد شده بهترین عملکرد را داشته باشد به عنوان بهینه ترین ساختار مشتق شده از ابرشبکه در نظر گرفته می شود. در ادامه پارامترهای ساختار حاصل با استفاده از روش هایی که در فصل های پیشین بررسی کردیم آموزش داده می شود.

---

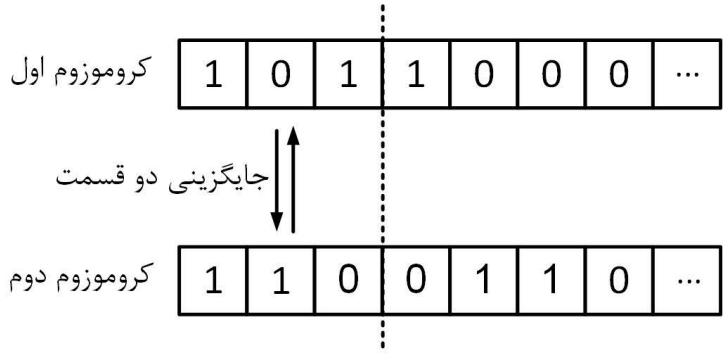
[1] Population
[2] Population size
[3] Epoch
[4] Parent population
[5] Cross over
[6] Mutation
[7] Child population



شکل ۱-۱۶: نحوه ایجاد کروموزوم های یک جمعیت در الگورتیم ژنتیک

۱۶.۲.۱.۱ برش

برش یکی از عملیات های تعریف شده در الگورتیم ژنتیک برای تولید جمعیت فرزند از جمعیت والد است. برای اعمال برش ابتدا دو کروموزوم به صورت تصادفی از جمعیت والد انتخاب می شوند. سپس یک بعد از ابعاد کروموزوم به صورت تصادفی انتخاب شده و هر دو کروموزوم از بعد انتخاب شده به دو قسمت تقسیم می شوند. سپس یکی از این چهار قسمت ایجاد شده با قسمت متناظر با خود، قسمتی که اندازه آن برابر است، در کروموزوم دیگری مطابق شکل ۲-۱۶ جایگزین می شود. بدین ترتیب دو کروموزوم فرزند از دو کروموزوم والد تولید می شود.



شکل ۲-۱۶: روند عملیات برش در الگورتیم ژنتیک

به دو کروموزوم والد می توان چند بار برش اعمال کرده و بدین ترتیب کروموزوم های فرزند بیشتری ایجاد کرد، همچنین در صورتی که اندازه یک کروموزوم بزرگ باشد می توانیم چند قسمت

مختلف دو کروموزوم را با هم جایگزین کنیم؛ کروموزوم ها را به جای دو قسمت به قسمت های بیشتری تقسیم کنیم.

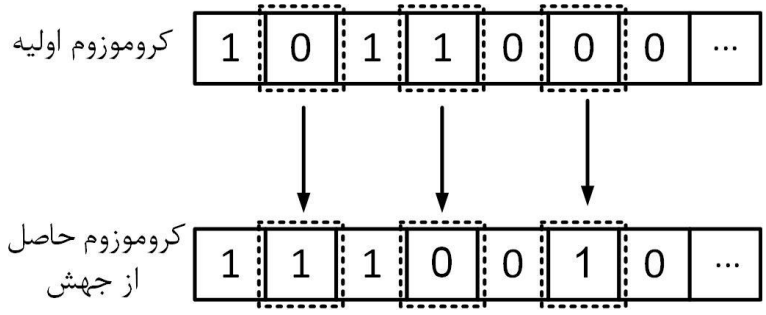
۱۶.۲.۱.۲ جهش

جهش یکی دیگر از عملیات های تعریف شده در الگوریتم ژنیتک برای تولید کروموزوم های فرزند از کروموزوم های والد است. روند اجرای آن بدین صورت است که ابتدا یک کروموزوم از جمعیت والد انتخاب می شود. سپس چند بعد مختلف از این کروموزوم به صورت تصادفی انتخاب شده و مقادیر آن بعد ها به صورت زیر تغییر می کنند:

۱. در صورتی کروموزوم ها به صورت باینری تعریف شده باشند، مقادیر صفر به یک و مقادیر یک به صفر تغییر می کنند.

۲. در صورتی که ابعاد مختلف کروموزوم ها دارای مقادیری حقیقی باشند مقدار هر بعد با مقدار قرینه آن جایگزین می شود. مقدار قرینه در این حالت نسبت به مقدار نیمه بازه اعداد تعریف می شود؛ بنابراین در صورتی که بازه اعداد بین منفی یک تا یک باشد این مقدار برابر با صفر و در صورتی که بازه اعداد بین صفر تا یک باشد ین مقدار برابر با ۰.۵ خواهد بود.

شکل ۳-۱۶ نمایش عملیات جهش برای یک کروموزوم باینری است.



شکل ۳-۱۶: روند عملیات جهش در الگوریتم ژنتیک

۱۶.۲.۱.۳ بررسی عملکرد الگوریتم ژنتیک

با توجه به اینکه الگورتیم ژنتیک معمولا برای مسائلی که دارای راه حل تحلیلی نیستند استفاده می شود، در عمل حد معینی برای توقف روند اجرای الگوریتم تعیین نمی شود. در الگوریتم ژنتیک لزوما عملکرد کروموزوم های فرزند نسبت به کروموزوم های جمعیت های پیش از خود بهتر نیست، از این رو تعیین حدی برای توقف اجرای الگورتیم چالش برانگیزتر می شود. از این رو معمولا حد

توقفی براساس زمان و یا حجم محاسبات برای این الگوریتم در نظر می گیرند. علاوه بر این حجم محاسبات بالا از دیگر معایب این الگوریتم است. در مقابل معایب ذکر شده الگوریتم ژنتیک برخلاف روش های آموزشی مانند پس انتشار خطا مبتنی بر گرادیان نبوده و می توان آن را برای طیف گسترده ای از مسائل استفاده کرد.

از الگوریتم ژنتیک می توان برای تعیین مقادیر وزن های یک ساختار عصبی نیز استفاده کرد، این روند معمولا پس از تعیین ساختار بهینه که در مطالب فوق ذکر شد اجرا می شود ولی می توان آن را همزمان با تعیین ساختار بهینه نیز اجرا کرد؛ البته اجرا همزمان آن با الگوریتم تعیین ساختار بهینه باعث افزایش حجم محاسبات و در مواردی تضعیف عملکرد الگوریتم می شود. برای تعیین مقادیر وزن ها، در کروموزوم ها ابعادی برابر با تعداد پارامترهای وزن ساختار در نظر گرفته می شود، این ابعاد دارای مقادیری حقیقی در بازه تعریف شده برای وزن های ساختار خواهند بود. وزن های حاصل از الگوریتم ژنتیک برای یک ساختار معمولا در ادامه به منظور بهبود عملکرد ساختار توسط یک روش مبتنی بر گرادیان مجددا آموزش داده می شوند[1]. مجموعه دادگانی که برای ارزیابی ساختار در این روش و یا سایر روش هایی که در ادامه فصل بررسی خواهیم کرد مورد استفاده قرار می گیرند در اکثر موارد همان مجموعه دادگان آموزشی مسئله و یا زیر مجموعه ای از آن هستند که به آن مجموعه دادگان نماینده[2] گفته می شود. با توجه به اینکه این مجموعه دادگان در طول تعیین ساختار ثابت هستند معمولا الگوریتم تعیین ساختار بهینه صرفا برای لایه های پنهان تعریف شده و ساختار لایه خروجی و ورودی با توجه به مجموعه دادگان تعریف می شود.

۱۶.۳ رویکردهای تعیین ساختار عصبی بر پایه روش های جستجو

رویکرد های مبتنی بر روش های جستجو برای تعیین ساختار بهینه عصبی نیز مشابه رویکردهای مبتنی بر آموزش بر پایه ابرشبکه تعریف می شوند. بدین صورت که ابتدا یک ابرشبکه تعریف شده و با استفاده از مجموعه دادگان آموزشی مسئله آموزش داده می شود. ساختار این ابرشبکه به گونه ای تعریف می شود که دارای خطای بسیار کمی بوده ولی سایر جنبه های عملکردی آن مانند حجم محاسبات و... بهینه نیست. پس از اتمام روند آموزش این ابرشبکه، یک روش جستجو برای یافتن زیرشبکه ای از آن که بتواند شرایط مطلوب مسئله را مرتفع کند تعریف می شود. شرایط مطلوب مسئله به صورت قیدهایی تعریف می شوند؛ برای مثال هدف جستجوی زیرشبکه ای از ابرشبکه تعیین می شود به طوری که حجم محاسبات، تعداد پارامترها، آن کمتر از نصف ابرشبکه باشد ولی خطای آن بیشتر از دو درصد نسبت به ابرشبکه افزایش پیدا نکند. ساختار شبکه پیچشی MobileNet V.3 که در فصل ۱۲ بررسی کردیم با استفاده از این روش از یک ابرشبکه مشتق شده است.

[1] Fine tunning
[2] Proxy dataset

۱۶.۴ رویکردهای خود سازمانده[1] در تعیین ساختار عصبی بهینه

در کنار روش های مبتنی بر ابرشبکه که پیشتر بررسی کردیم برخی رویکردهای خودسازمانده نیز برای تعیین ساختار بهینه توسعه داده شده اند. این رویکردها معمولا در طی روند آموزش ساختار اجرا می شوند. بدین صورت که ابتدا یک ساختار عصبی با ساختاری کوچک، ساختاری با تعداد لایه کم، تعریف شده و روند آموزش آن با استفاده از یک روش مبتنی بر گرادیان مشابه آنچه در فصل ۹ بررسی کردیم آغاز می شود. گام های آموزشی ساختار تا جایی که مقدار توابع هزینه گام آموزش و ارزیابی به یک مقدار همگرا شوند ادامه پیدا می کند، در این مرحله یک لایه به ساختار اضافه شده و روند آموزش ادامه پیدا می کند، اگر مقادیر توابع هزینه پس از چند گام نسبت به حالت همگرائی پیشین کمتر شوند لایه اضافه شده در ساختار باقی مانده و اگر این مقادیر افزایش یابند و یا تغییری نکنند این لایه مجددا از ساختار حذف می شود. این رویکرد معمولا فقط تعداد لایه های ساختار تعیین می شود. با اضافه شدن لایه جدید سایر لایه های ساختار نیز باید نسبت به آن، از نظر اندازه ماتریس وزن ها، تغییر کنند. انتخاب محل مناسب برای افزودن لایه جدید از چالش های این رویکرد است. از آنجائی که با اضافه کردن یک لایه عملا مسئله بهینه سازی در طول روند بهینه سازی تغییر پیدا می کند، از این رو استفاده از این رویکرد می تواند نقض تئوریک مسئله بهینه سازی تلقی شود اما با وجود این نقض استفاده از این رویکرد معمولا به ساختارهای مناسب برای شبکه تعریف شده برای مسئله منجر می شود.

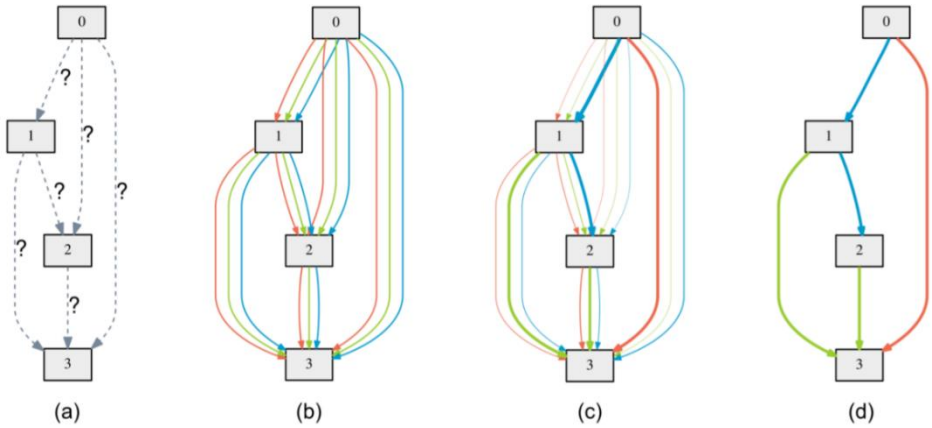
۱۶.۵ رویکرد جستجوی ساختار عصبی مشتق پذیر[2]

رویکرد مشتق پذیر معمولا برای انتخاب عملیات مناسب در لایه های مختلف یک ساختار عصبی به کار برده می شود. بدین صورت که ابتدا ساختار کلی شبکه مطابق شکل ۴-۱۶ a تعریف می شود. در ادامه برای هر یک از اتصالات تعریف شده بین ساختار همه عملیات های تعریف شده مطابق شکل ۴-۱۶ b تعریف می شود، به هر یک از این عملیات ها یک پارامتر α نسبت داده می شود، مشابه ساختار نورون های راف[3]، مقادیر این وزن ها در ابتدا با هم برابر است. یک تابع بیشینه هموار[4] به مقادیر پارامترهای عملیات های تعریف شده در هر اتصال، α، اعمال شده و مقدار متناظر حاصل از تابع بیشینه هموار در خروجی هر عملیات ضرب می شوند. خروجی های عملیات های مختلف که مقدار پارامتر حاصل از تابع بیشینه هموار متناظر با آن ها در هر کدام ضرب شده است در نهایت با هم جمع شده و به صورت یک میانگین وزنی خروجی نهائی اتصال را ایجاد می کنند. در طول روند آموزش ساختار مطابق شکل ۴-۱۶ c برای این پارامترهای وزن α نیز در کنار سایر پارامترهای ساختار گرادیان های آموزشی تعریف شده و در نهایت پس از اتمام روند آموزش عملیات

[1] Self-organized
[2] Differentiable architecture search (DARTS)
[3] Rough
[4] Soft-Max

دارای بیشـــترین مقدار وزن به عنوان عملیات منتخب آن اتصـــال تعیین شـــده و در نهایت ســـاختار حاصل مطابق شکل ۴-۱۶ d خواهد بود.

در این روش برای آموزش مقادیر وزن های α یک مسـئله بهینه سـازی دو سـطحی[1] تعریف می شـود؛ بدین صـورت که ابتدا مقادیر وزن های همه عملیات ها آموزش داده شـده و پس از اتمام روند بهینه سازی وزن عملیات های مختلف، مقادیر پارامترهای α با استفاده از گرادیان های تعریف شده آموزش داده می شوند، ولی با توجه به حجم محاسبات بالای این روش با فرض ساده سازی روند به روزرســانی مقادیر وزن ها و مقادیر پارامترهای α در هر گام آموزشــی به صــورت ســری در کنار هم تعریف می شـــوند. از این روش می توان برای انتخاب توابع فعال ســـاز مناســـب و... در ســـاختارهای مختلف استفاده کرد.



شکل ۴-۱۶: روند رویکرد جستجوی ساختار عصبی مشتق پذیر[۱۵۷]

[1] Bilevel optimization

۱۶.۶ مسائل

۱. علاوه بر الگوریتم ژنتیک در مورد سایر روش های مطرح در یادگیری تکاملی که برای تعیین ساختار بهینه به کاربرده می شوند تحقیق کنید.

۲. در مورد مفاهیم یادگیری تقویتی و نحوه به کارگیری آن ها برای تعیین ساختار عصبی بهینه تحقیق کنید.

۳. الگوریتم های مختلف جستجو را بررسی کرده و مزایا و معایب هر کدام را بیان کنید. آیا ساختار تعریف شده در مسئله در انتخاب الگوریتم جستجو تاثیرگذار است؟

۴. روابط آموزش مربوط به پارامترهای α در رویکرد جستجوی ساختار عصبی مشتق پذیر را به روش گرادیان نزولی توسعه دهید.

پیوست ۱

۱۷.۱ پیچیدگی محاسباتی[1]

روش های مختلفی برای محاسبه حجم محاسبات لازم برای اجرای یک الگوریتم ارائه می شود، مانند تعداد پارامترها و... ولی به طور کلی در متون مختلف پیچیدگی محاسباتی بر اساس روند تغییرات زمان اجرای الگوریتم با تغییر ابعاد ورودی است. برای مثال یک حلقه را در نظر بگیرید که برای اجرای یک عملیات با زمان اجرای ثابت بر روی ابعاد مختلف یک بردار ورودی تعریف شده است، در این صورت زمان اجرای حلقه با افزوده شدن هر یک بعد به بردار ورودی همواره به میزان ثابتی افزایش می یابد، بنابراین پیچیدگی محاسباتی این الگوریتم به صورت خطی تعریف شده برابر با $O(n)$ خواهد بود که در آن n تعداد ابعاد بردار است. در صورتی که این الگوریتم برای یک ماتریس با ابعاد $m \times n$ تعریف شود پیچیدگی آن معادل $O(m \times n)$ خواهد بود. روند تغییر زمان اجرای همه الگوریتم ها به با ابعاد ورودی رابطه خطی ندارند، برای مثال ممکن است پیچیدگی محاسباتی برخی الگوریتم ها به صورت توانی از بعد ورودی، $O(n^m)$، باشد در این صورت این الگوریتم زمان اجرای بسیار بالایی برای ابعاد ورودی بزرگ خواهد داشت، علاوه برا پیچدگی خطی و توانی، الگوریتم ها پیچیدگی های مختلفی مانند پیچیدگی محاسباتی لگاریتمی، $O(\log(n))$ نیز ممکن است داشته باشند. در مواردی که روند تغییرات زمان اجرای الگوریتم از رابطه خاصی پیروی نکند معمولا آن را با یک رابطه تقریب می زنند.

۱۷.۲ روند مقعر محدب[2]

طبق قضیه روند مقعر محدب، CCCP، اگر یک رابطه به صورت مجموع دو رابطه مقعر و محدب مطابق رابطه a تعریف کنیم:

$$E_{(\boldsymbol{x})} = E_{concave(\boldsymbol{x})} + E_{convex(\boldsymbol{x})} \quad \text{(a)}$$

در این صورت رابطه b برقرار خواهد بود.

$$\nabla E_{convex(x_{t+1})} = -\nabla E_{concave(x_t)} \quad \text{(b)}$$

برای روابط مقعر و محدب طبق نامساوی جنسون[3] به ترتیب روابط c و d را داریم:

$$E_{concave(\boldsymbol{x}_{t+1})} \leq E_{concave(\boldsymbol{x}_t)} + (\boldsymbol{x}_{t+1} - \boldsymbol{x}_t).\nabla E_{concave(\boldsymbol{x}_t)} \quad \text{(c)}$$

---

[1] Computational complexity (CC)
[2] The concave convex procedure (CCCP)
[3] Jenson's inequality

$$E_{convex(x_t)} \geq E_{convex(x_{t+1})} + (x_t - x_{t+1}).\nabla E_{convex(x_{t+1})} \quad \text{(d)}$$

در روابط c و d، x_t و x_{t+1} نقاطی ورودی هستند. با جایگذاری رابطه b در روابط c و d و سپس جمع طرفین دو رابطه با هم رابطه e حاصل می شود:

$$E_{concave(x_{t+1})} + E_{convex(x_{t+1})} \leq E_{concave(x_t)} + E_{convex(x_t)} \quad \text{(e)}$$

طبق نامساوی جنسون و تعاریف روابط مقعر و محدب رابطه e همواره برقرار است، از این رو رابطه b اثبات می شود. بدین ترتیب برای رابطه a نقطه بهینه، همان زینی[1] حاصل از دو رابطه مقعر و محدب خواهد بود. برای اثبات این موضوع رابطه a معادل رابطه f در نظر می گیریم:

$$E_{(x)} = E_{convex1(x)} - E_{convex2(x)} \quad \text{(f)}$$

در رابطه f نقطه بهینه نقطه ای است که به ازای آن مقدار تانژانت خط مماس بر دو رابطه محدب با هم برابر باشند که همان نقطه زینی رابطه a است.

[1] Saddling point

مراجع

[1] Bishop, Christopher. *Pattern Recognition and Machine Learning (Information Science and Statistics)*. Springer, 2018.

[2] Goodfellow, Ian, et al. *Deep Learning (Adaptive Computation and Machine Learning Series)*. Illustrated, The MIT Press, 2016.

[3] Yuille, A. L., and Anand Rangarajan. “The Concave-Convex Procedure.” *Neural Computation*, vol. 15, no. 4, 2003, pp. 915–36. *Crossref*, doi:10.1162/08997660360581958.

[4] Schuster, M., and K. K. Paliwal. “Bidirectional Recurrent Neural Networks.” *IEEE Transactions on Signal Processing*, vol. 45, no. 11, 1997, pp. 2673–81. *Crossref*, doi:10.1109/78.650093.

[5] Vaswani, Ashish, et al. "Attention is all you need." *Advances in neural information processing systems*. 2017.

[6] Chu, Jielei, et al. "Restricted boltzmann machines with gaussian visible units guided by pairwise constraints." *IEEE transactions on cybernetics* 49.12 (2018): 4321-4334.

[7] Hu, Hengyuan, Lisheng Gao, and Quanbin Ma. "Deep restricted boltzmann networks." *arXiv preprint arXiv:1611.07917* (2016).

[8] Tolstikhin, Ilya, et al. "Wasserstein auto-encoders." *arXiv preprint arXiv:1711.01558* (2017).

[9] Park, Saerom, and Jaewook Lee. "Stability Analysis of Denoising Autoencoders Based on Dynamical Projection System." *IEEE Transactions on Knowledge and Data Engineering* (2020).

[10] Vincent, Pascal, et al. "Stacked denoising autoencoders: Learning useful representations in a deep network with a local denoising criterion." *Journal of machine learning research* 11.12 (2010).

[11] Hinton, Geoffrey E. "Boltzmann machine." *Scholarpedia* 2.5 (2007): 1668.

[12] Alain, Guillaume, and Yoshua Bengio. "What regularized auto-encoders learn from the data-generating distribution." *The Journal of Machine Learning Research* 15.1 (2014): 3563-3593.

[13] Ng, Andrew. "Sparse autoencoder." *CS294A Lecture notes* 72.2011 (2011): 1-19.

[14] Hinton, Geoffrey E. "A practical guide to training restricted Boltzmann machines." *Neural networks: Tricks of the trade*. Springer, Berlin, Heidelberg, 2012. 599-619.

[15] Chen, Fu-qiang, et al. "Contractive de-noising auto-encoder." *International Conference on Intelligent Computing*. Springer, Cham, 2014.

[16] Hwang, Juno, Wonseok Hwang, and Junghyo Jo. "Tractable loss function and color image generation of multinary restricted Boltzmann machine." *arXiv preprint arXiv:2011.13509* (2020).

[17] Carlson, David, Volkan Cevher, and Lawrence Carin. "Stochastic spectral descent for restricted Boltzmann machines." *Artificial Intelligence and Statistics*. PMLR, 2015.

[18] Thickstun, John. "Kantorovich-Rubinstein Duality." (2019).

[19] Hoffman, Matthew D., et al. "Stochastic variational inference." *Journal of Machine Learning Research* 14.5 (2013).

[20] Cremer, Chris, Xuechen Li, and David Duvenaud. "Inference suboptimality in

variational autoencoders." *International Conference on Machine Learning*. PMLR, 2018.
[21] Rezende, Danilo Jimenez, Shakir Mohamed, and Daan Wierstra. "Stochastic backpropagation and approximate inference in deep generative models." *International conference on machine learning*. PMLR, 2014.
[22] Kingma, Diederik P., and Max Welling. "Auto-encoding variational bayes." *arXiv preprint arXiv:1312.6114* (2013).
[23] Zhang, Cheng, et al. "Advances in variational inference." *IEEE transactions on pattern analysis and machine intelligence* 41.8 (2018): 2008-2026.
[24] Wainwright, Martin J., and Michael Irwin Jordan. *Graphical models, exponential families, and variational inference*. Now Publishers Inc, 2008.
[25] Johnson, Matthew J., et al. "Composing graphical models with neural networks for structured representations and fast inference." *Advances in neural information processing systems* 29 (2016): 2946-2954.
[26] Yoshizawa, Shuji, Masahiko Morita, and Shun-Ichi Amari. "Capacity of associative memory using a nonmonotonic neuron model." *Neural Networks* 6.2 (1993): 167-176.
[27] Torres, Joaquín J., Lovorka Pantic, and Hilbert J. Kappen. "Storage capacity of attractor neural networks with depressing synapses." *Physical Review E* 66.6 (2002): 061910.
[28] Demircigil, Mete, et al. "On a model of associative memory with huge storage capacity." *Journal of Statistical Physics* 168.2 (2017): 288-299.
[29] Amit, Daniel J., Hanoch Gutfreund, and Haim Sompolinsky. "Statistical mechanics of neural networks near saturation." *Annals of physics* 173.1 (1987): 30-67.
[30] Krotov, Dmitry, and John J. Hopfield. "Dense associative memory for pattern recognition." *Advances in neural information processing systems* 29 (2016): 1172-1180.
[31] Cho, Youngmin. *Kernel methods for deep learning*. University of California, San Diego, 2012.
[32] Osogami, Takayuki. "Boltzmann machines and energy-based models." *arXiv preprint arXiv:1708.06008* (2017).
[33] Hopfield, John J. "Neural networks and physical systems with emergent collective computational abilities." *Proceedings of the national academy of sciences* 79.8 (1982): 2554-2558.
[34] Gretton, Arthur, et al. "A kernel two-sample test." *The Journal of Machine Learning Research* 13.1 (2012): 723-773.
[35] Yuan, Ao, et al. "U-statistic with side information." *Journal of multivariate analysis* 111 (2012): 20-38.
[36] Ramsauer, Hubert, et al. "Hopfield networks is all you need." *arXiv preprint arXiv:2008.02217* (2020).
[37] Widrich, Michael, et al. "Modern hopfield networks and attention for immune repertoire classification." *arXiv preprint arXiv:2007.13505* (2020).
[38] Demircigil, Mete, et al. "On a model of associative memory with huge storage capacity." *Journal of Statistical Physics* 168.2 (2017): 288-299.
[39] Elman, Jeffrey L. "Finding structure in time." *Cognitive science* 14.2 (1990): 179-211.
[40] Krotov, Dmitry, and John Hopfield. "Large associative memory problem in neurobiology and machine learning." *arXiv preprint arXiv:2008.06996* (2020).
[41] Kim, Do-Hyun, Jinha Park, and Byungnam Kahng. "Enhanced storage capacity with errors in scale-free Hopfield neural networks: An analytical study." *PloS one* 12.10

(2017): e0184683.
[42] Amit, Daniel J., Hanoch Gutfreund, and Haim Sompolinsky. "Statistical mechanics of neural networks near saturation." *Annals of physics* 173.1 (1987): 30-67.
[43] Jordan, M. I. *Serial order: a parallel distributed processing approach. Technical report, June 1985-March 1986*. No. AD-A-173989/5/XAB; ICS-8604. California Univ., San Diego, La Jolla (USA). Inst. for Cognitive Science, 1986.
[44] Lipton, Zachary C., John Berkowitz, and Charles Elkan. "A critical review of recurrent neural networks for sequence learning." *arXiv preprint arXiv:1506.00019* (2015).
[45] Ratcliff, Roger. "A theory of memory retrieval." *Psychological review* 85.2 (1978): 59.
[46] Montúfar, Guido. "Restricted boltzmann machines: Introduction and review." *Information Geometry and Its Applications IV*. Springer, Cham, 2016.
[47] Aarts, Emile HL, and Jan HM Korst. "Boltzmann machines and their applications." *International Conference on Parallel Architectures and Languages Europe*. Springer, Berlin, Heidelberg, 1987.
[48] Jaeger, Herbert, and Harald Haas. "Harnessing nonlinearity: Predicting chaotic systems and saving energy in wireless communication." *science* 304.5667 (2004): 78-80.
[49] Robbins, Herbert, and Sutton Monro. "A stochastic approximation method." *The annals of mathematical statistics* (1951): 400-407.
[50] Dauphin, Yann, et al. "Identifying and attacking the saddle point problem in high-dimensional non-convex optimization." *arXiv preprint arXiv:1406.2572* (2014).
[51] Darken, Christian, Joseph Chang, and John Moody. "Learning rate schedules for faster stochastic gradient search." *Neural networks for signal processing*. Vol. 2. 1992.
[52] Sutton, Richard. "Two problems with back propagation and other steepest descent learning procedures for networks." *Proceedings of the Eighth Annual Conference of the Cognitive Science Society, 1986*. 1986.
[53] Qian, Ning. "On the momentum term in gradient descent learning algorithms." *Neural networks* 12.1 (1999): 145-151.
[54] Nesterov, Yurii E. "A method for solving the convex programming problem with convergence rate O (1/k^ 2)." *Dokl. akad. nauk Sssr*. Vol. 269. 1983.
[55] Bengio, Yoshua, Nicolas Boulanger-Lewandowski, and Razvan Pascanu. "Advances in optimizing recurrent networks." *2013 IEEE International Conference on Acoustics, Speech and Signal Processing*. IEEE, 2013.
[56] Duchi, John, Elad Hazan, and Yoram Singer. "Adaptive subgradient methods for online learning and stochastic optimization." *Journal of machine learning research* 12.7 (2011).
[57] Sutskever, Ilya. *Training recurrent neural networks*. Toronto, Canada: University of Toronto, 2013.
[58] Dean, Jeffrey, et al. "Large scale distributed deep networks." *Advances in neural information processing systems* 25 (2012): 1223-1231.
[59] Pennington, Jeffrey, Richard Socher, and Christopher D. Manning. "Glove: Global vectors for word representation." *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*. 2014.
[60] Zeiler, Matthew D. "Adadelta: an adaptive learning rate method." *arXiv preprint arXiv:1212.5701* (2012).
[61] Heusel, Martin, et al. "Gans trained by a two time-scale update rule converge to a local nash equilibrium." *Advances in neural information processing systems* 30

(2017).
[62] Dozat, Timothy. "Incorporating nesterov momentum into adam." (2016).
[63] Huang, Gao, et al. "Densely connected convolutional networks." *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2017.
[64] Johnson, Melvin, et al. "Google's multilingual neural machine translation system: Enabling zero-shot translation." *Transactions of the Association for Computational Linguistics* 5 (2017): 339-351.
[65] Reddi, Sashank J., Satyen Kale, and Sanjiv Kumar. "On the convergence of adam and beyond." *arXiv preprint arXiv:1904.09237* (2019).
[66] Loshchilov, I., and F. Hutter. "Decoupled weight decay regularization. arXiv." *Preprint published January* 4 (2019).
[67] Ma, Jerry, and Denis Yarats. "Quasi-hyperbolic momentum and adam for deep learning." *arXiv preprint arXiv:1810.06801* (2018).
[68] Lucas, James, et al. "Aggregated momentum: Stability through passive damping." *arXiv preprint arXiv:1804.00325* (2018).
[69] Niu, Feng, et al. "Hogwild!: A lock-free approach to parallelizing stochastic gradient descent." *arXiv preprint arXiv:1106.5730* (2011).
[70] McMahan, Brendan, and Matthew Streeter. "Delay-tolerant algorithms for asynchronous distributed online learning." *Advances in Neural Information Processing Systems* 27 (2014): 2915-2923.
[71] Abadi, Martín, et al. "Tensorflow: Large-scale machine learning on heterogeneous distributed systems." *arXiv preprint arXiv:1603.04467* (2016).
[72] Zhang, Sixin, Anna Choromanska, and Yann LeCun. "Deep learning with elastic averaging SGD." *arXiv preprint arXiv:1412.6651* (2014).
[73] LeCun, Y., L. Bottou, and G. B. Orr. "Neural Networks-Tricks of the Trade, vol. 1524, ed. by G. Orr and K. Müller." (1998): 5-50.
[74] Bengio, Yoshua, et al. "Curriculum learning." *Proceedings of the 26th annual international conference on machine learning*. 2009.
[75] Zaremba, Wojciech, and Ilya Sutskever. "Learning to execute." *arXiv preprint arXiv:1410.4615* (2014).
[76] Ioffe, Sergey, and Christian Szegedy. "Batch normalization: Accelerating deep network training by reducing internal covariate shift." *International conference on machine learning*. PMLR, 2015.
[77] Neelakantan, Arvind, et al. "Adding gradient noise improves learning for very deep networks." *arXiv preprint arXiv:1511.06807* (2015).
[78] Bradbury, James, et al. "Quasi-recurrent neural networks." *arXiv preprint arXiv:1611.01576* (2016).
[79] Balduzzi, David, and Muhammad Ghifary. "Strongly-typed recurrent neural networks." *International Conference on Machine Learning*. PMLR, 2016.
[80] Jaeger, Herbert. "Echo state network." *scholarpedia* 2.9 (2007): 2330.
[81] Gallicchio, Claudio, and Alessio Micheli. "Deep echo state network (deepesn): A brief survey." *arXiv preprint arXiv:1712.04323* (2017).
[82] Vlachas, Pantelis R., et al. "Backpropagation algorithms and reservoir computing in recurrent neural networks for the forecasting of complex spatiotemporal dynamics." *Neural Networks* 126 (2020): 191-217.
[83] Ba, Jimmy Lei, Jamie Ryan Kiros, and Geoffrey E. Hinton. "Layer normalization." *arXiv preprint arXiv:1607.06450* (2016).
[84] Krizhevsky, Alex, Ilya Sutskever, and Geoffrey E. Hinton. "Imagenet classification with deep convolutional neural networks." *Advances in neural information processing systems* 25 (2012): 1097-1105.

[85] Khan, Asifullah, et al. "A survey of the recent architectures of deep convolutional neural networks." *Artificial Intelligence Review* 53.8 (2020): 5455-5516.
[86] Szegedy, Christian, et al. "Inception-v4, inception-resnet and the impact of residual connections on learning." *Thirty-first AAAI conference on artificial intelligence*. 2017.
[87] Dang, Lanxue, Peidong Pang, and Jay Lee. "Depth-wise separable convolution neural network with residual connection for hyperspectral image classification." *Remote Sensing* 12.20 (2020): 3408.
[88] Chollet, François. "Xception: Deep learning with depthwise separable convolutions." *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2017.
[89] Maas, Andrew L., Awni Y. Hannun, and Andrew Y. Ng. "Rectifier nonlinearities improve neural network acoustic models." *Proc. icml*. Vol. 30. No. 1. 2013.
[90] Shang, Wenling, et al. "Understanding and improving convolutional neural networks via concatenated rectified linear units." *international conference on machine learning*. PMLR, 2016.
[91] Clevert, Djork-Arné, Thomas Unterthiner, and Sepp Hochreiter. "Fast and accurate deep network learning by exponential linear units (elus)." *arXiv preprint arXiv:1511.07289* (2015).
[92] Agarap, Abien Fred. "Deep learning using rectified linear units (relu)." *arXiv preprint arXiv:1803.08375* (2018).
[93] Dumoulin, Vincent, and Francesco Visin. "A guide to convolution arithmetic for deep learning." *arXiv preprint arXiv:1603.07285* (2016).
[94] Woo, Sanghyun, et al. "Cbam: Convolutional block attention module." *Proceedings of the European conference on computer vision (ECCV)*. 2018.
[95] Hu, Jie, Li Shen, and Gang Sun. "Squeeze-and-excitation networks." *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2018.
[96] Kumar, Siddharth Krishna. "On weight initialization in deep neural networks." *arXiv preprint arXiv:1704.08863* (2017).
[97] Elfwing, Stefan, Eiji Uchibe, and Kenji Doya. "Sigmoid-weighted linear units for neural network function approximation in reinforcement learning." *Neural Networks* 107 (2018): 3-11.
[98] Lu, Zhilong, et al. "LSTM variants meet graph neural networks for road speed prediction." *Neurocomputing* 400 (2020): 34-45.
[99] Krause, Ben, et al. "Multiplicative LSTM for sequence modelling." *arXiv preprint arXiv:1609.07959* (2016).
[100] Wu, Yuhuai, et al. "On multiplicative integration with recurrent neural networks." *arXiv preprint arXiv:1606.06630* (2016).
[101] He, Kaiming, et al. "Mask r-cnn." *Proceedings of the IEEE international conference on computer vision*. 2017.
[102] Redmon, Joseph, et al. "You only look once: Unified, real-time object detection." *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2016.
[103] Tan, Mingxing, and Quoc Le. "Efficientnet: Rethinking model scaling for convolutional neural networks." *International Conference on Machine Learning*. PMLR, 2019.
[104] Kolesnikov, Alexander, et al. "Big transfer (bit): General visual representation learning." *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part V 16*. Springer International Publishing, 2020.

[105] Lin, Tsung-Yi, et al. "Focal loss for dense object detection." *Proceedings of the IEEE international conference on computer vision*. 2017.
[106] Szegedy, Christian, et al. "Going deeper with convolutions." *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2015.
[107] Szegedy, Christian, et al. "Inception-v4, inception-resnet and the impact of residual connections on learning." *Thirty-first AAAI conference on artificial intelligence*. 2017.
[108] Ronneberger, Olaf, Philipp Fischer, and Thomas Brox. "U-net: Convolutional networks for biomedical image segmentation." *International Conference on Medical image computing and computer-assisted intervention*. Springer, Cham, 2015.
[109] Bansal, Aayush, et al. "Pixelnet: Representation of the pixels, by the pixels, and for the pixels." *arXiv preprint arXiv:1702.06506* (2017).
[110] Zhang, Cheng, et al. "Advances in variational inference." *IEEE transactions on pattern analysis and machine intelligence* 41.8 (2018): 2008-2026.
[111] Gulrajani, Ishaan, et al. "Pixelvae: A latent variable model for natural images." *arXiv preprint arXiv:1611.05013* (2016).
[112] Stuner, Bruno, Clément Chatelain, and Thierry Paquet. "Handwriting recognition using cohort of LSTM and lexicon verification with extremely large lexicon." *Multimedia Tools and Applications* 79.45 (2020): 34407-34427.
[113] Wigington, Curtis, et al. "Start, follow, read: End-to-end full-page handwriting recognition." *Proceedings of the European Conference on Computer Vision (ECCV)*. 2018.
[114] Martínek, Jiří, Ladislav Lenc, and Pavel Král. "Training strategies for OCR systems for historical documents." *IFIP International Conference on Artificial Intelligence Applications and Innovations*. Springer, Cham, 2019.
[115] Sabour, Sara, Nicholas Frosst, and Geoffrey E. Hinton. "Dynamic routing between capsules." *arXiv preprint arXiv:1710.09829* (2017).
[116] Patrick, Mensah Kwabena, et al. "Capsule networks–a survey." *Journal of King Saud University-computer and information sciences* (2019).
[117] Choi, Jaewoong, et al. "Attention routing between capsules." *Proceedings of the IEEE/CVF International Conference on Computer Vision Workshops*. 2019.
[118] Fu, Jun, et al. "Dual attention network for scene segmentation." *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2019.
[119] Maity, Rajib. "Basic Concepts of Probability and Statistics." *Statistical Methods in Hydrology and Hydroclimatology*. Springer, Singapore, 2018. 7-51.
[120] Liu, Linfeng, and Liping Liu. "Localizing and Amortizing: Efficient Inference for Gaussian Processes." *Asian Conference on Machine Learning*. PMLR, 2020.
[121] Ng, Andrew, and Sparse Autoencoder. "CS294A Lecture notes." *Dosegljivo: https://web. stanford. edu/class/cs294a/sparseAutoencoder_2011new. pdf.[Dostopano 20. 7. 2016]* (2011).
[122] Roy, Jean-Francis, Mario Marchand, and François Laviolette. "A column generation bound minimization approach with PAC-Bayesian generalization guarantees." *Artificial Intelligence and Statistics*. PMLR, 2016.
[123] Ganin, Yaroslav, et al. "Domain-adversarial training of neural networks." *The journal of machine learning research* 17.1 (2016): 2096-2030.
[124] Martens, James. "New insights and perspectives on the natural gradient method." *arXiv preprint arXiv:1412.1193* (2014).
[125] Pascanu, Razvan, and Yoshua Bengio. "Revisiting natural gradient for deep networks." *arXiv preprint arXiv:1301.3584* (2013).
[126] Ly, Alexander, et al. "A tutorial on Fisher information." *Journal of Mathematical*

Psychology 80 (2017): 40-55.
[127] Kingma, Diederik P., and Jimmy Ba. "Adam: A method for stochastic optimization." *arXiv preprint arXiv:1412.6980* (2014).
[128] Nguyen, XuanLong, Martin J. Wainwright, and Michael I. Jordan. "On surrogate loss functions and f-divergences." *The Annals of Statistics* 37.2 (2009): 876-904.
[129] Polyanskiy, Yury. "January 7, 2020 Typed by Suzanne Sigalla (ENSAE, CREST)." (2020).
[130] Glorot, Xavier, and Yoshua Bengio. "Understanding the difficulty of training deep feedforward neural networks." *Proceedings of the thirteenth international conference on artificial intelligence and statistics*. JMLR Workshop and Conference Proceedings, 2010.
[131] He, Kaiming, et al. "Delving deep into rectifiers: Surpassing human-level performance on imagenet classification." *Proceedings of the IEEE international conference on computer vision*. 2015.
[132] Mishkin, Dmytro, and Jiri Matas. "All you need is a good init." *arXiv preprint arXiv:1511.06422* (2015).
[133] Mitrović, Dalibor, Matthias Zeppelzauer, and Christian Breiteneder. "Features for content-based audio retrieval." *Advances in computers*. Vol. 78. Elsevier, 2010. 71-150.
[134] Sheikhpour, Razieh, et al. "A survey on semi-supervised feature selection methods." *Pattern Recognition* 64 (2017): 141-158.
[135] Kuhn, Max, and Kjell Johnson. *Applied predictive modeling*. Vol. 26. New York: Springer, 2013.
[136] Chun-Lin, Liu. "A tutorial of the wavelet transform." *NTUEE, Taiwan* (2010).
[137] Bahdanau, Dzmitry, Kyunghyun Cho, and Yoshua Bengio. "Neural machine translation by jointly learning to align and translate." *arXiv preprint arXiv:1409.0473* (2014).
[138] Gao, Junyu, Qi Wang, and Yuan Yuan. "SCAR: Spatial-/channel-wise attention regression networks for crowd counting." *Neurocomputing* 363 (2019): 1-8.
[139] van de Geijn, Robert, and Margaret Myers. "Advanced Linear Algebra: Foundations to Frontiers." *Creative Commons NonCommercial (CC BY-NC)* (2020).
[140] Kouemou, Guy Leonard, and Dr Przemyslaw Dymarski. "History and theoretical basics of hidden Markov models." *Hidden Markov Models, Theory and Applications* 1 (2011).
[141] Chen, Long, et al. "Sca-cnn: Spatial and channel-wise attention in convolutional networks for image captioning." *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2017.
[142] Wang, Qilong, et al. "ECA-Net: efficient channel attention for deep convolutional neural networks, 2020 IEEE." *CVF Conference on Computer Vision and Pattern Recognition (CVPR). IEEE*. 2020.
[143] Liu, Yang, et al. "An attention-gated convolutional neural network for sentence classification." *Intelligent Data Analysis* 23.5 (2019): 1091-1107.
[144] Popescu, Gabriel. *Quantitative phase imaging of cells and tissues*. McGraw-Hill Education, 2011.
[145] Hemanth, D. Jude, Deepak Gupta, and Valentina Emilia Balas, eds. *Intelligent Data Analysis for Biomedical Applications: Challenges and Solutions*. Academic Press, 2019.
[146] Kehtarnavaz, Nasser. "Frequency Domain Processing." *Digital Signal Processing System Design*, 2008.
[147] Cerna, Michael, and Audrey F. Harvey. *The fundamentals of FFT-based signal analysis and measurement*. Application Note 041, National Instruments, 2000.
[148] L. Martignon. *International Encyclopedia of the Social & Behavioral Sciences*, Pergamon,

2001.
[149] Sinha, Ankur, Pekka Malo, and Kalyanmoy Deb. "A review on bilevel optimization: from classical to evolutionary approaches and applications." *IEEE Transactions on Evolutionary Computation* 22.2 (2017): 276-295.
[150] Torrey, Lisa, and Jude Shavlik. "Transfer learning." *Handbook of research on machine learning applications and trends: algorithms, methods, and techniques*. IGI global, 2010. 242-264.
[151] Hospedales, Timothy, et al. "Meta-learning in neural networks: A survey." *arXiv preprint arXiv:2004.05439* (2020).
[152] Wang, Xiaolong, et al. "Non-local neural networks." *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2018.
[153] Ullah, Amin, et al. "Action recognition in video sequences using deep bi-directional LSTM with CNN features." *IEEE access* 6 (2017): 1155-1166.
[154] Zhang, Han, et al. "Self-attention generative adversarial networks." *International conference on machine learning*. PMLR, 2019.
[155] Wick, Christoph, Christian Reul, and Frank Puppe. "Comparison of OCR Accuracy on Early Printed Books using the Open Source Engines Calamari and OCRopus." *J. Lang. Technol. Comput. Linguistics* 33.1 (2018): 79-96.
[156] Yin, Wenpeng, et al. "Abcnn: Attention-based convolutional neural network for modeling sentence pairs." *Transactions of the Association for Computational Linguistics* 4 (2016): 259-272.
[157] Liu, Hanxiao, Karen Simonyan, and Yiming Yang. "Darts: Differentiable architecture search." *arXiv preprint arXiv:1806.09055* (2018).
[158] Sutskever, Ilya, Oriol Vinyals, and Quoc V. Le. "Sequence to sequence learning with neural networks." *Advances in neural information processing systems*. 2014.
[159] Cheng, Jianpeng, Li Dong, and Mirella Lapata. "Long short-term memory-networks for machine reading." *arXiv preprint arXiv:1601.06733* (2016).
[160] Luong, Minh-Thang, Hieu Pham, and Christopher D. Manning. "Effective approaches to attention-based neural machine translation." *arXiv preprint arXiv:1508.04025* (2015).
[161] Snell, Jake, Kevin Swersky, and Richard S. Zemel. "Prototypical networks for few-shot learning." *arXiv preprint arXiv:1703.05175* (2017).
[162] Chen, Ting, et al. "Self-supervised gans via auxiliary rotation loss." *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2019.
[163] Chen, Ting, et al. "Self-supervised gans via auxiliary rotation loss." *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2019.
[164] LeCun, Yann, et al. "Gradient-based learning applied to document recognition." *Proceedings of the IEEE* 86.11 (1998): 2278-2324.
[165] Krizhevsky, Alex, Ilya Sutskever, and Geoffrey E. Hinton. "Imagenet classification with deep convolutional neural networks." *Advances in neural information processing systems* 25 (2012): 1097-1105.
[166] Simonyan, Karen, and Andrew Zisserman. "Very deep convolutional networks for large-scale image recognition." *arXiv preprint arXiv:1409.1556* (2014).
[167] He, Kaiming, et al. "Deep residual learning for image recognition." *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2016.
[168] Huang, Gao, et al. "Densely connected convolutional networks." *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2017.
[169] Sandler, Mark, et al. "Mobilenetv2: Inverted residuals and linear bottlenecks." *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2018.
[170] Howard, Andrew G., et al. "Mobilenets: Efficient convolutional neural networks for mobile vision applications." *arXiv preprint arXiv:1704.04861* (2017).
[171] Howard, Andrew, et al. "Searching for mobilenetv3." *Proceedings of the IEEE/CVF International Conference on Computer Vision*. 2019.
[172] Girshick, Ross, et al. "Rich feature hierarchies for accurate object detection and semantic segmentation." *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2014.
[173] Girshick, Ross. "Fast r-cnn." *Proceedings of the IEEE international conference on computer*

vision. 2015.
[174] Ren, Shaoqing, et al. "Faster r-cnn: Towards real-time object detection with region proposal networks." *Advances in neural information processing systems* 28 (2015): 91-99.
[175] Uijlings, Jasper RR, et al. "Selective search for object recognition." *International journal of computer vision* 104.2 (2013): 154-171.
[176] Felzenszwalb, Pedro F., and Daniel P. Huttenlocher. "Efficient graph-based image segmentation." *International journal of computer vision* 59.2 (2004): 167-181.
[177] Liu, Wei, et al. "Ssd: Single shot multibox detector." *European conference on computer vision*. Springer, Cham, 2016.
[178] Lin, Tsung-Yi, et al. "Feature pyramid networks for object detection." *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2017.
[179] Mirza, Mehdi, and Simon Osindero. "Conditional generative adversarial nets." *arXiv preprint arXiv:1411.1784* (2014).
[180] Radford, Alec, Luke Metz, and Soumith Chintala. "Unsupervised representation learning with deep convolutional generative adversarial networks." *arXiv preprint arXiv:1511.06434* (2015).
[181] Arjovsky, Martin, Soumith Chintala, and Léon Bottou. "Wasserstein generative adversarial networks." *International conference on machine learning*. PMLR, 2017.
[182] Zhu, Jun-Yan, et al. "Unpaired image-to-image translation using cycle-consistent adversarial networks." *Proceedings of the IEEE international conference on computer vision*. 2017.
[183] Liu, Steven, et al. "Diverse image generation via self-conditioned gans." *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2020.
[184] Liu, Shichen, et al. "Generalized zero-shot learning with deep calibration network." *Advances in Neural Information Processing Systems*. 2018.
[185] Wang, Mei, and Weihong Deng. "Deep visual domain adaptation: A survey." *Neurocomputing* 312 (2018): 135-153.
[186] Xian, Yongqin, et al. "Zero-shot learning—a comprehensive evaluation of the good, the bad and the ugly." *IEEE transactions on pattern analysis and machine intelligence* 41.9 (2018): 2251-2265.
[187] Brown, Tom B., et al. "Language models are few-shot learners." *arXiv preprint arXiv:2005.14165* (2020).
[188] Devlin, Jacob, et al. "Bert: Pre-training of deep bidirectional transformers for language understanding." *arXiv preprint arXiv:1810.04805* (2018).
[189] Zhou, Bolei, Aditya Khosla, Agata Lapedriza, Aude Oliva, and Antonio Torralba. "Learning deep features for discriminative localization." In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 2921-2929. 2016.

Deep Learning

Theoretical and Practical Approach

Author:
Sina Ranjbar Kooh Farhadi

DOI: 10.5281/zenodo.6672318