



Waveform-based music processing with deep learning

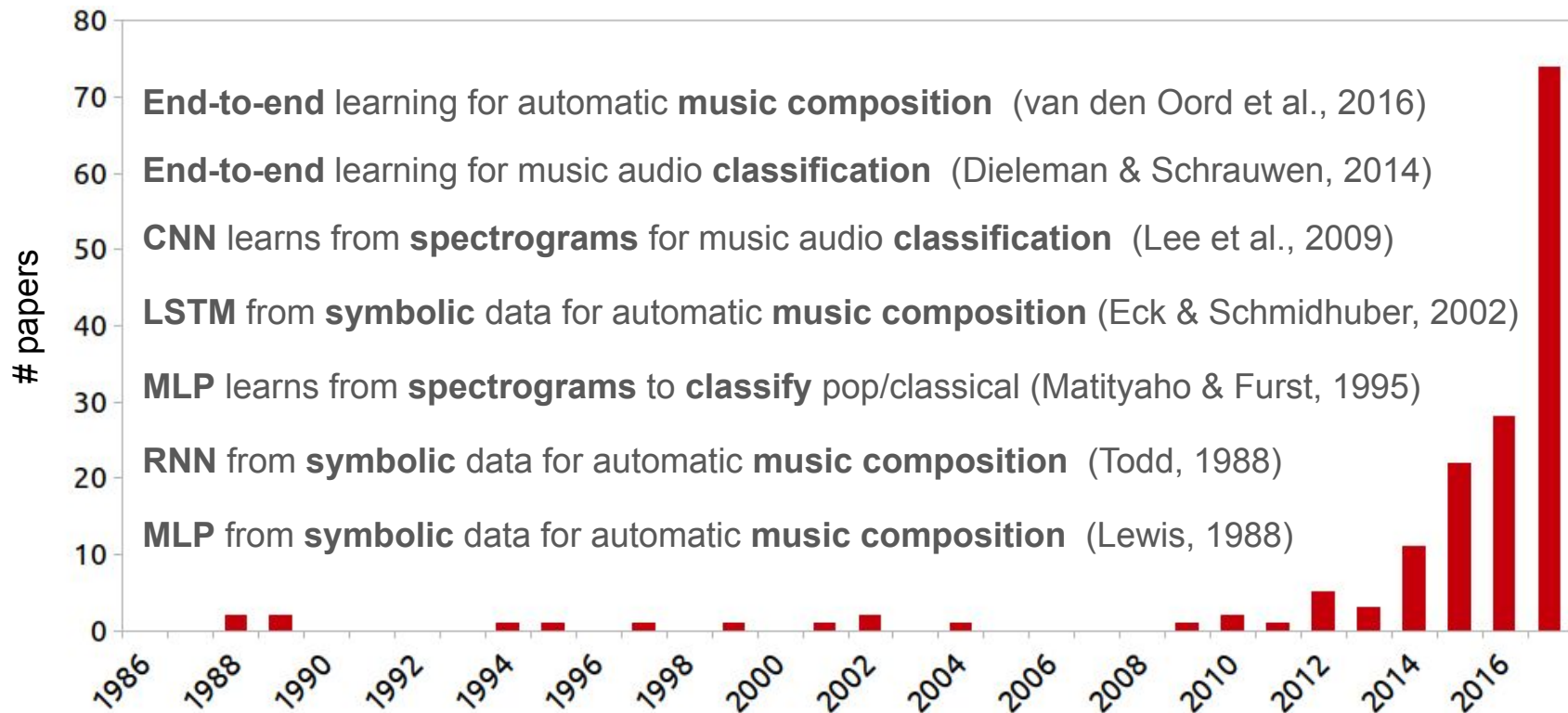
Sander Dieleman, Jordi Pons, Jongpil Lee



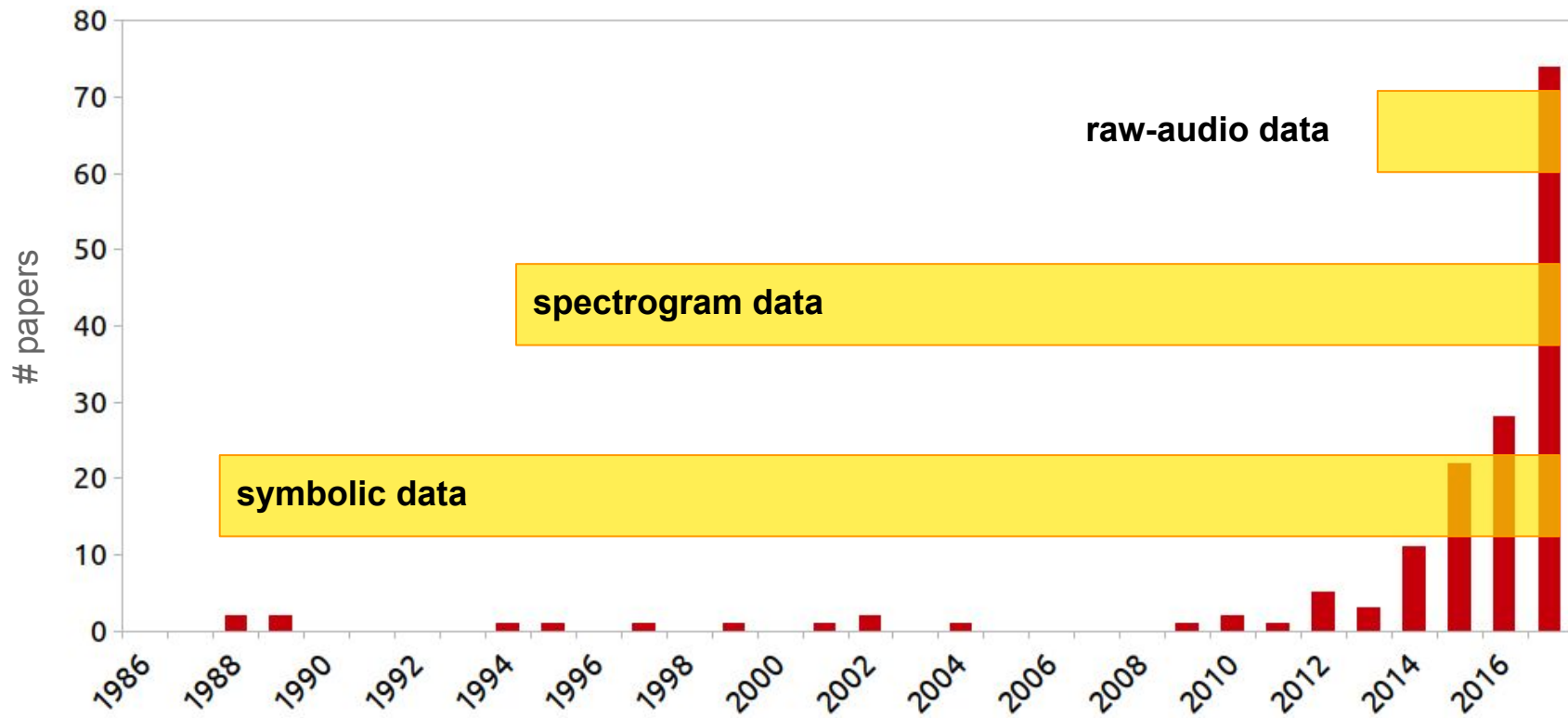
Historical perspective

Why now?

Papers on “neural networks & music”: milestones



Papers on “neural networks & music”: input data



References

Van den Oord et al., 2016. “Wavenet: a generative model for audio” in arXiv.

Dieleman and Schrauwen, 2014. “End-to-end learning for music audio”
in International Conference on Acoustics, Speech and Signal Processing (ICASSP).

Lee et al., 2009. “Unsupervised feature learning for audio classification using convolutional deep belief networks”
in Advances in Neural Information Processing Systems (NIPS).

Matityaho and Furst, 1995. “Neural network based model for classification of music type”
in 18th Convention of Electrical and Electronics Engineers in Israel. IEEE, 4–3.

Eck and Schmidhuber, 2002. “Finding temporal structure in music: Blues improvisation with LSTM recurrent networks”
in Proceedings of the Workshop on Neural Networks for Signal Processing.

Todd, 1988. “A sequential network design for musical applications”
in Proceedings of the Connectionist Models Summer School.

Lewis, 1988. Creation by Refinement: A creativity paradigm for gradient descent learning networks”
in International Conference on Neural Networks.

Historical review from: Pons, 2019. “Deep neural networks for music and audio tagging”. PhD Thesis.

Waveform-based music processing with deep learning

Classification


by Jongpil Lee, PhD candidate at KAIST in Daejeon, South Korea.

Source Separation

by Jordi Pons, researcher at Dolby Laboratories in Barcelona.

Generation

by Sander Dieleman, research scientist at DeepMind in London, UK.



Music Classification

Learning specific properties of music

Outline

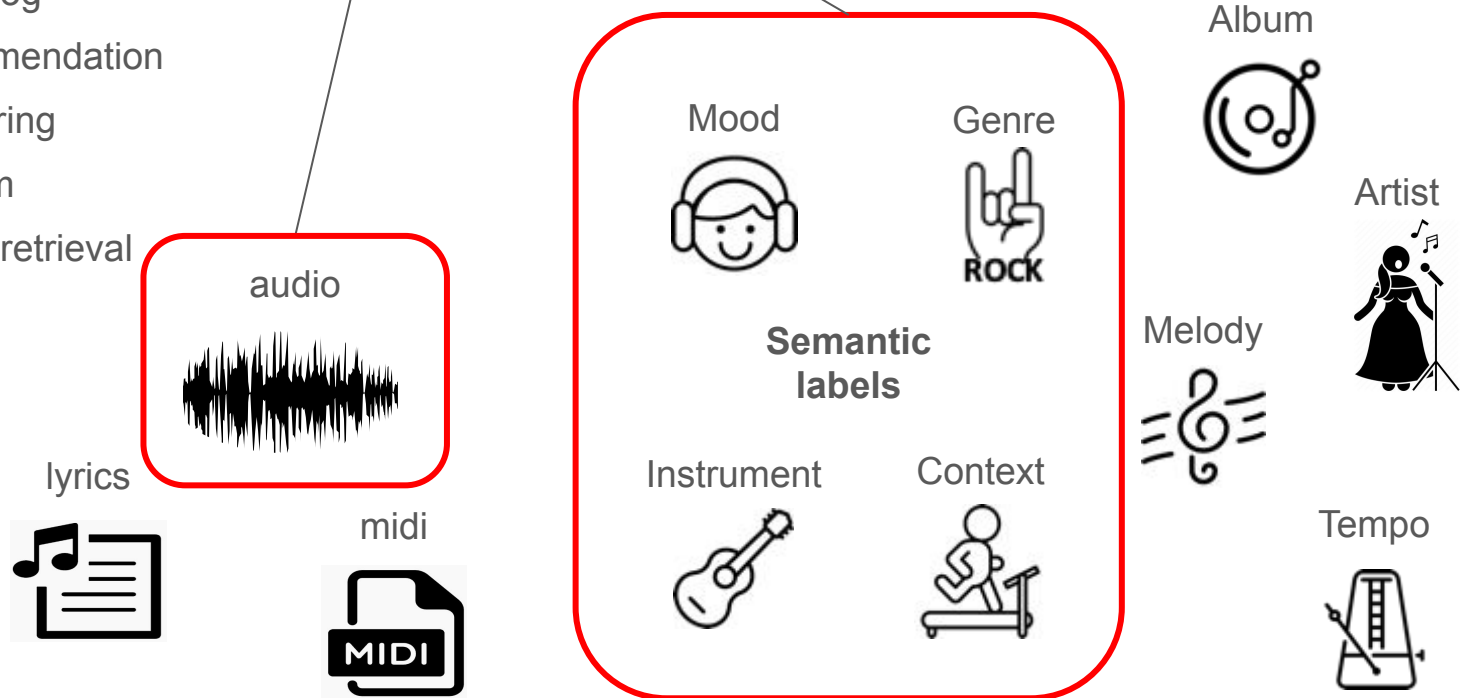
- Problem statement
- Audio classification models
 - *Engineered feature based model*
 - *Frame-level waveform based model*
 - *Sample-level waveform based model*
- Connect signal processing and deep learning
- Summary

Outline

- **Problem statement**
- Audio classification models
 - *Engineered feature based model*
 - *Frame-level waveform based model*
 - *Sample-level waveform based model*
- Connect signal processing and deep learning
- Summary

Music classification

Large music catalog
Retrieval / recommendation
Collaborative filtering
Cold-start problem
Tag-based music retrieval
Auto-tagging



Music



Scene



Image

Music (artwork)



Speech (object)



Audio



Multi source **polyphonic**
Unstructured sound sources

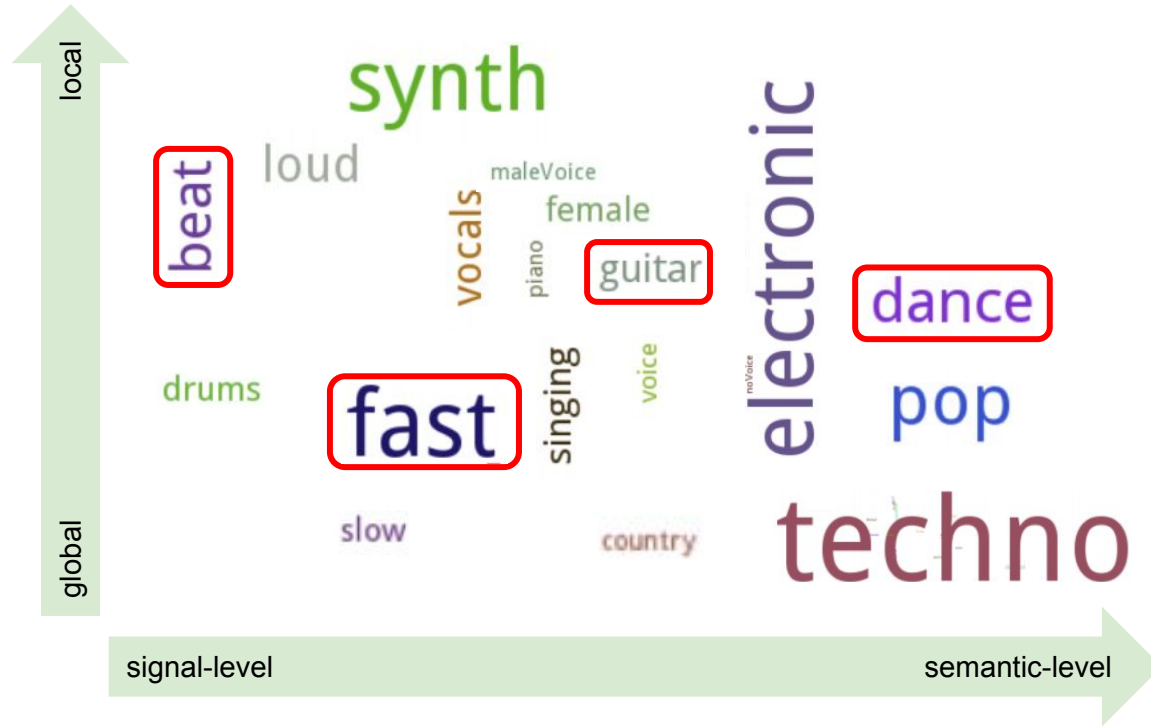


Multi timbre (texture) **polyphonic**
Structure of sound sources
Dynamic control (mixing)



Mostly **single** speaker
A **structured** sound source

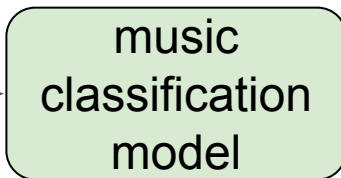
Semantic labels



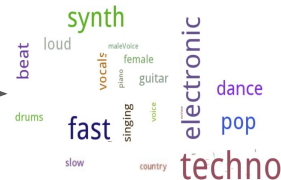
Semantic labels are highly diverse and have different levels of abstraction

Problem definition

music audio



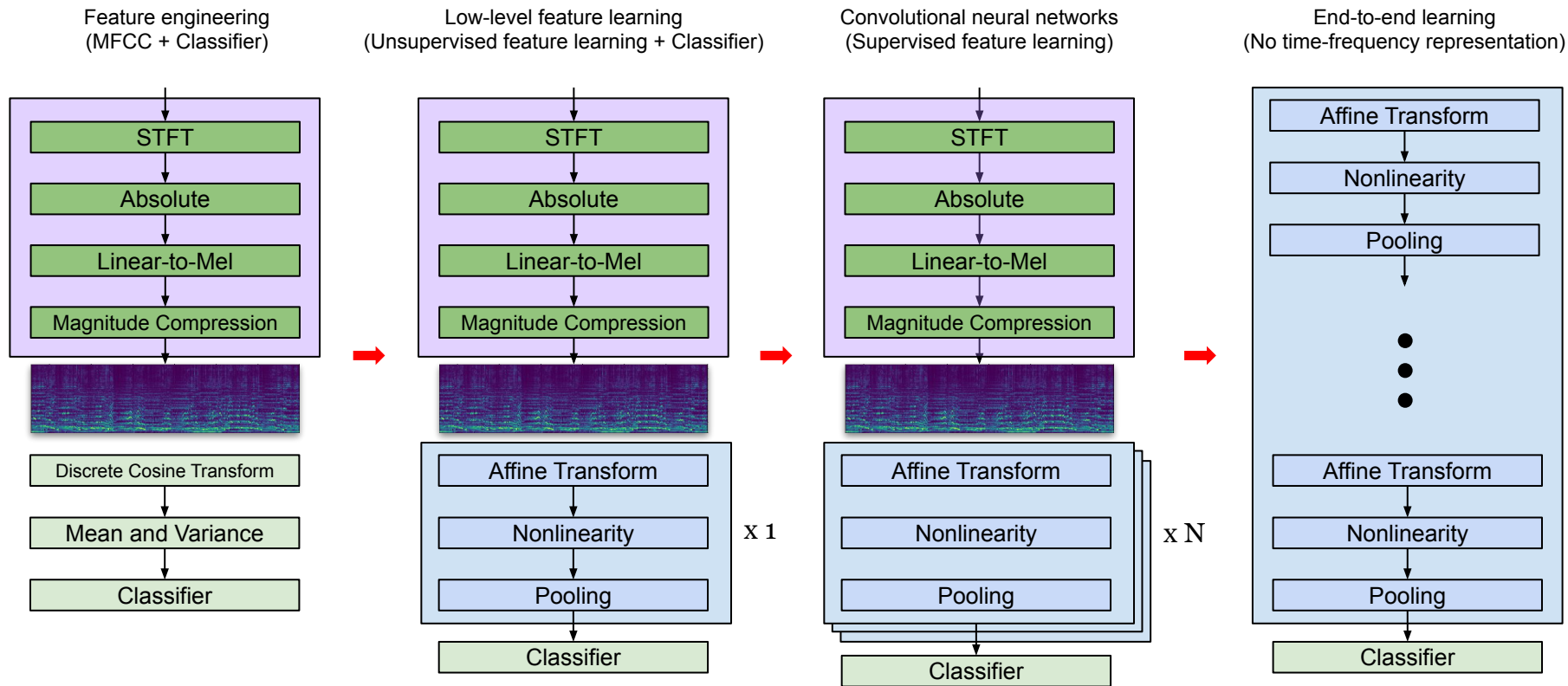
semantic labels



Outline

- Problem statement
- **Audio classification models**
 - *Engineered feature based model*
 - *Frame-level waveform based model*
 - *Sample-level waveform based model*
- Connect signal processing and deep learning
- Summary

From feature engineering to end-to-end learning



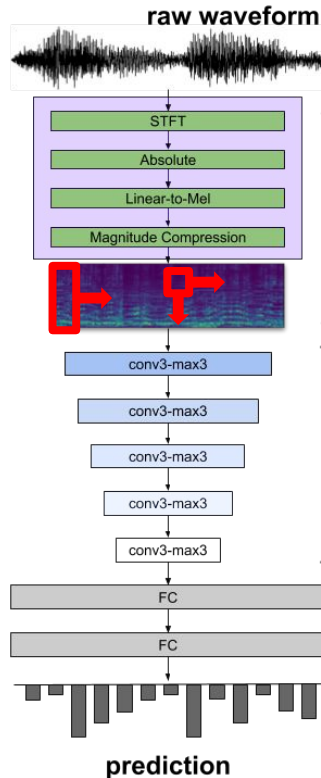
Nam et al., 2018. "Deep learning for audio-based music classification and tagging", *IEEE Signal Processing Magazine*.

Humphrey et al., 2013. "Feature learning and deep architectures: new directions for music informatics", *Journal of Intelligent Information Systems*.

Outline

- Problem statement
- Audio classification models
 - ***Engineered feature based model***
 - *Frame-level waveform based model*
 - *Sample-level waveform based model*
- Connect signal processing and deep learning
- Summary

Engineered feature based model



Front-end

Back-end

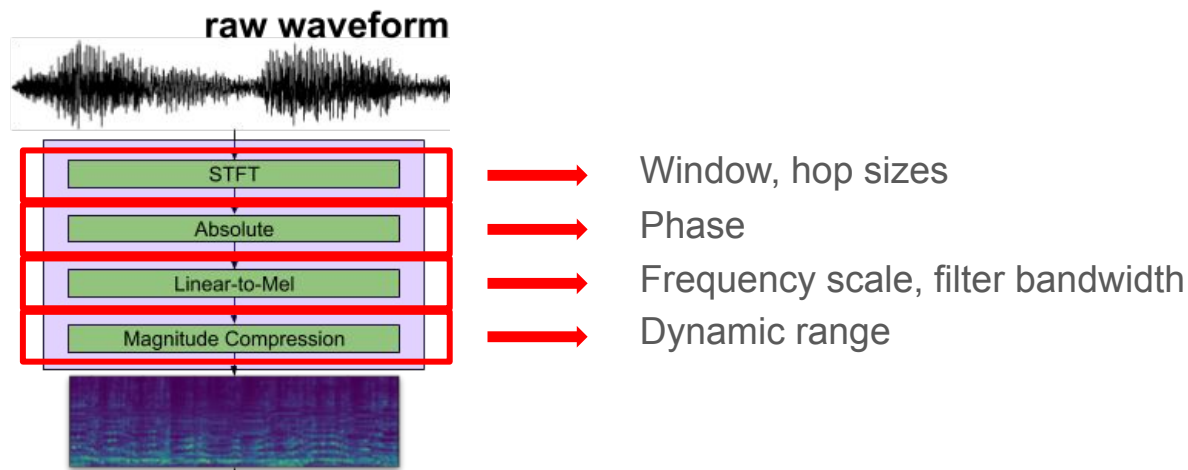
- MFCC, Chroma, Bag of low-level-features
- **Mel-spectrogram**
- Scattering transform
- **1D CNNs**
- **2D CNNs**
- RNNs
- Attention

Pons et al., 2018. "End-to-end learning for music audio tagging at scale", *ISMIR*.

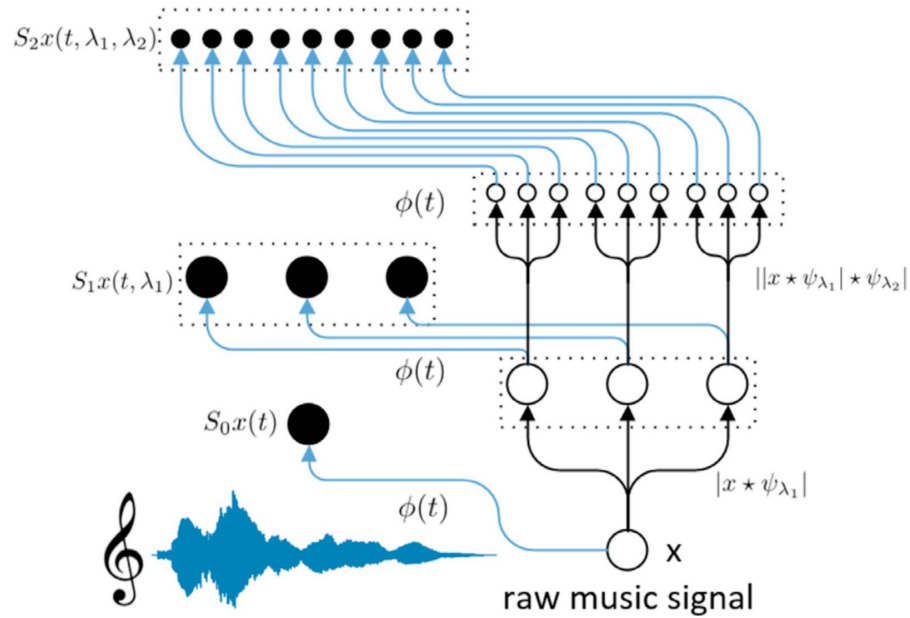
Choi et al., 2016. "Automatic tagging using deep convolutional neural networks", *ISMIR*.

Purwins et al., 2019. "Deep Learning for Audio Signal Processing", *IEEE Journal of Selected Topics in Signal Processing*.

Fixed parameters in preprocessing stage



Scattering transform



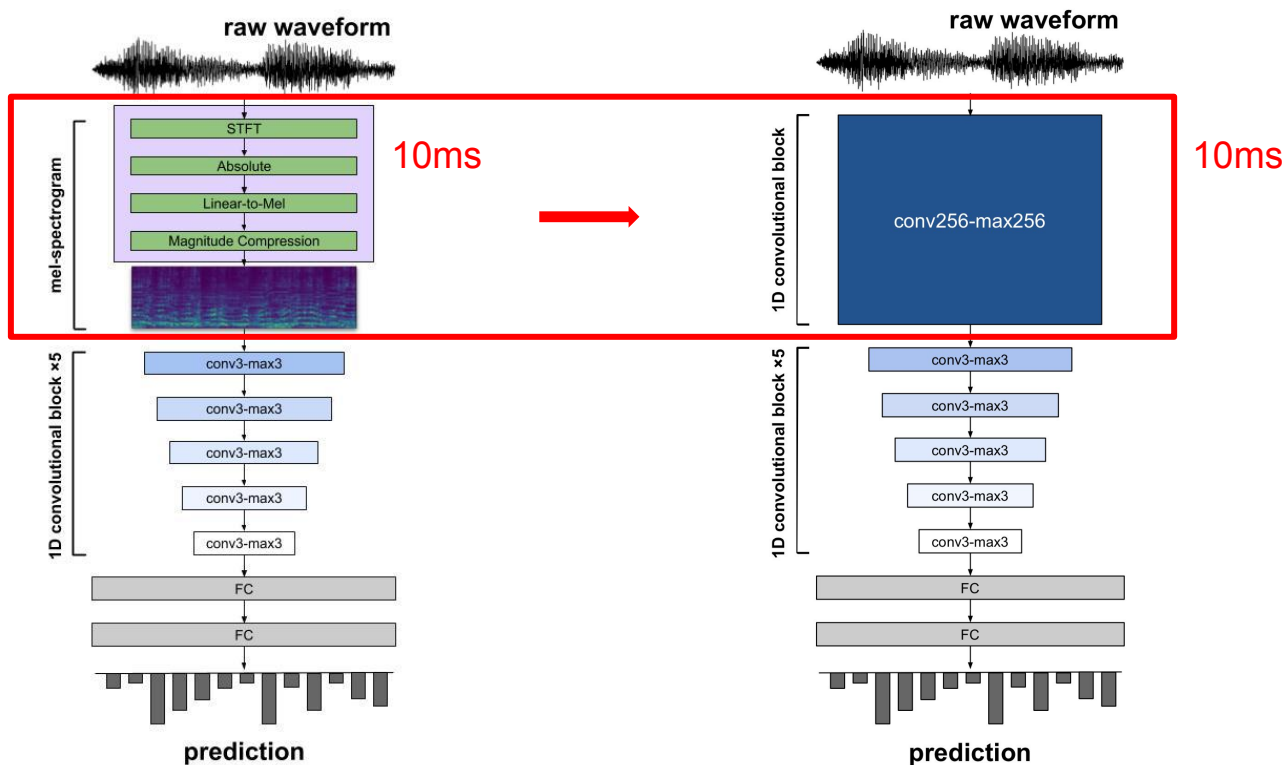
Andén et al., 2011. "Multiscale Scattering for Audio Classification", *ISMIR*.

Song et al., 2018. "Music auto-tagging using deep Recurrent Neural Networks", *Neurocomputing*.

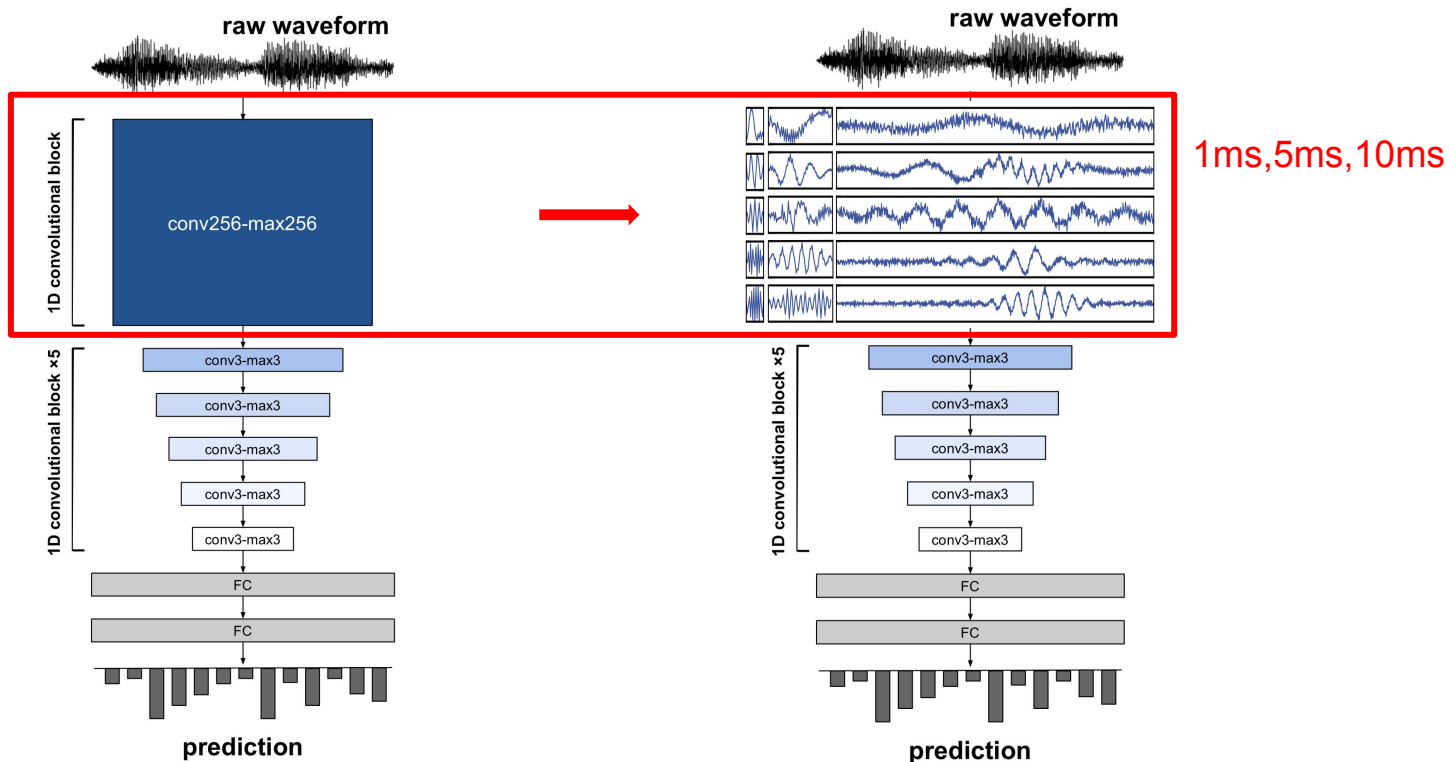
Outline

- Problem statement
- Audio classification models
 - *Engineered feature based model*
 - ***Frame-level waveform based model***
 - *Sample-level waveform based model*
- Connect signal processing and deep learning
- Summary

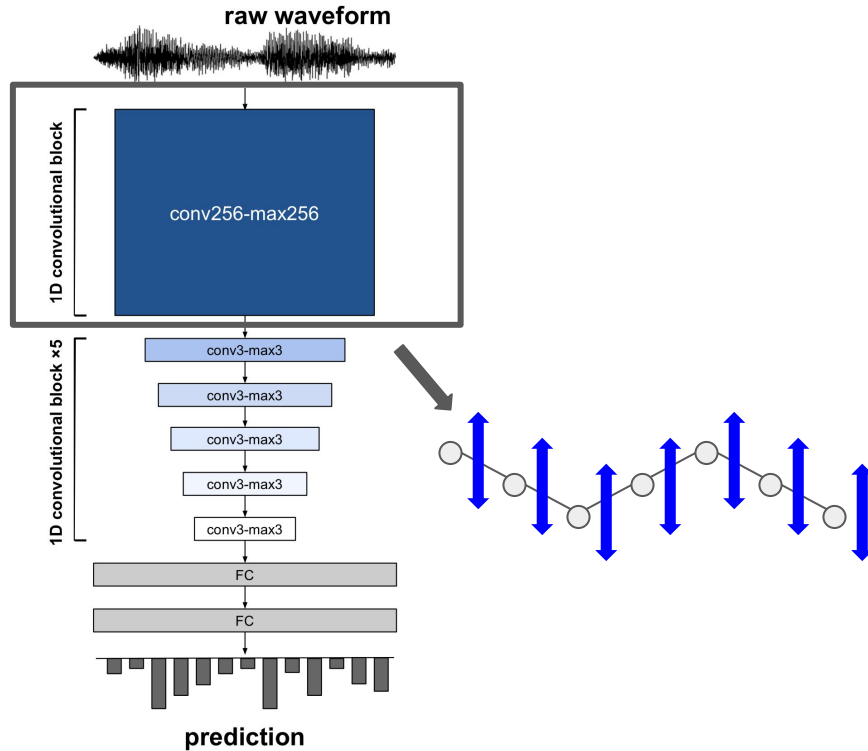
Frame-level waveform based model



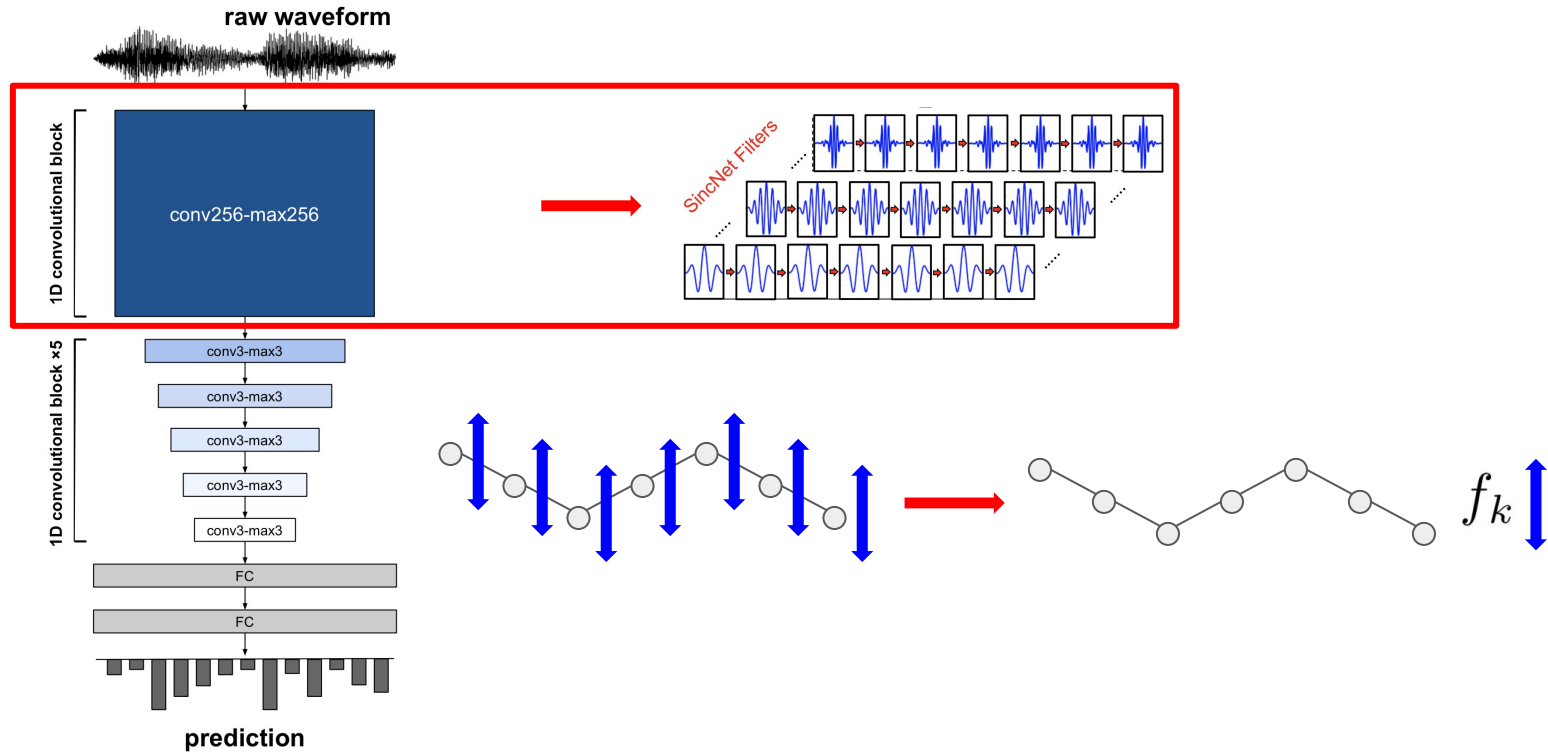
Multi-scale approaches



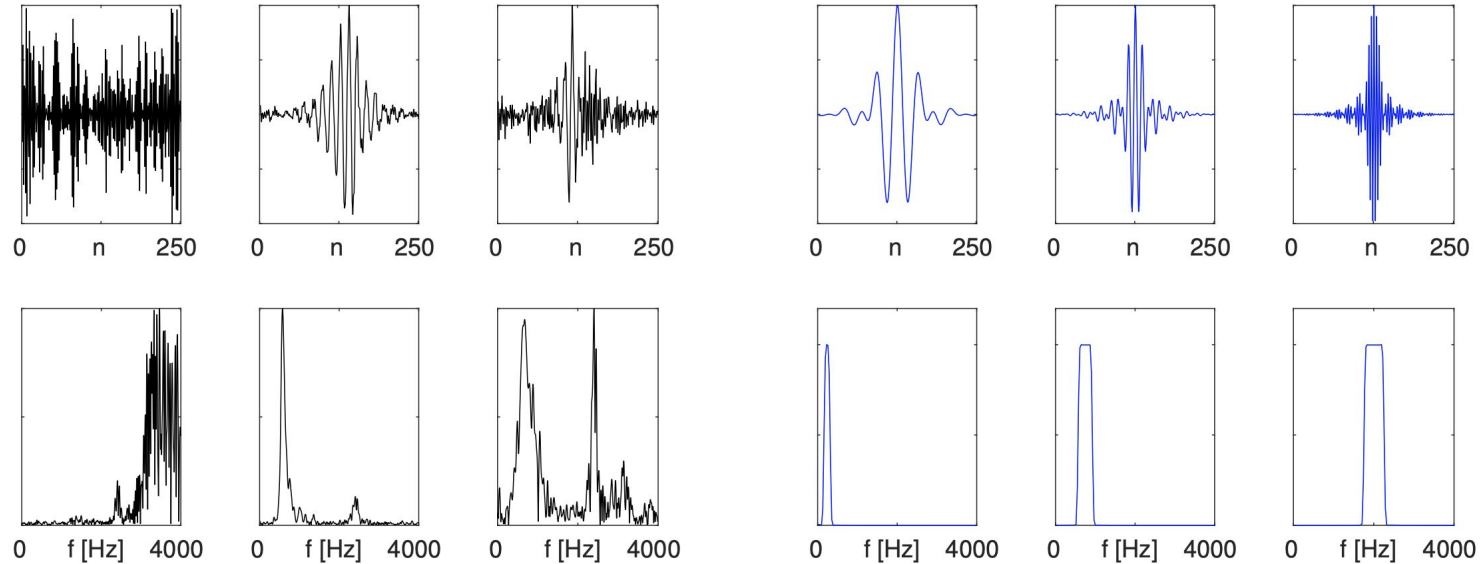
SincNet



SincNet



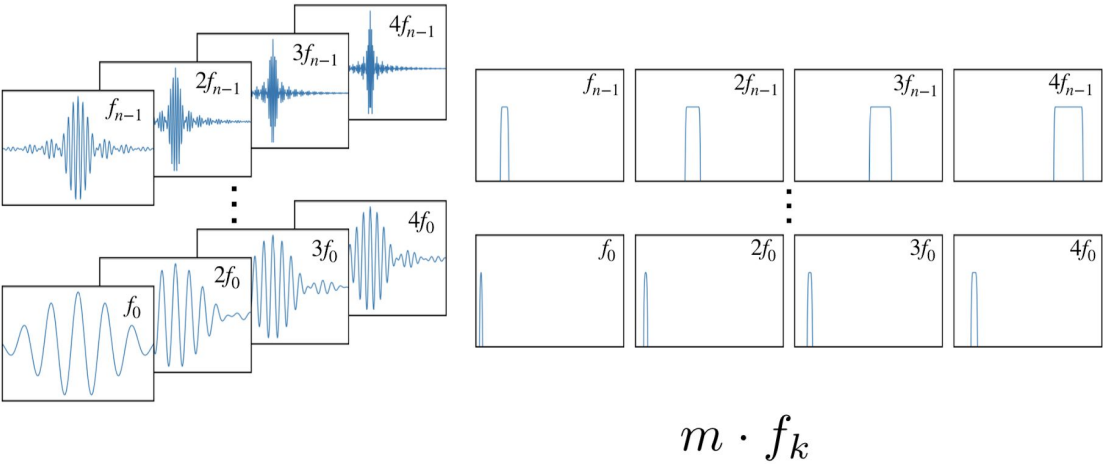
SincNet



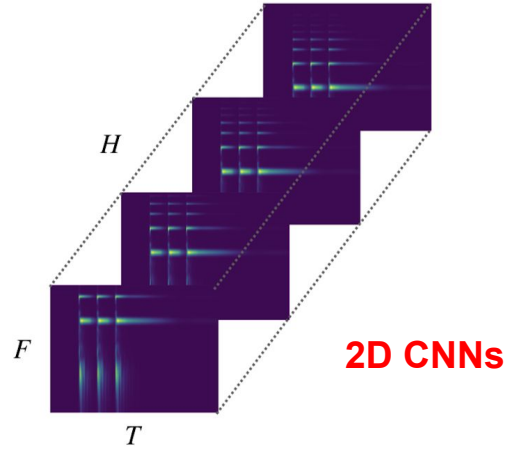
(a) CNN Filters

(b) SincNet Filters

HarmonicCNN

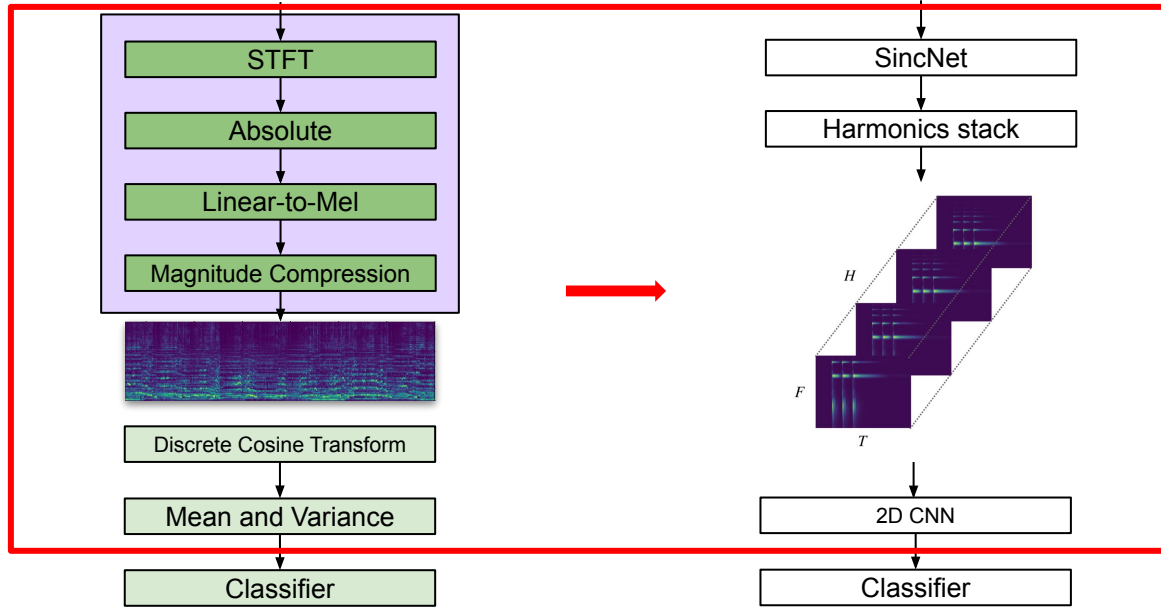


Harmonic embedding features are stacked at **channel dimension**



HarmonicCNN

Feature engineering
(MFCC + Classifier)

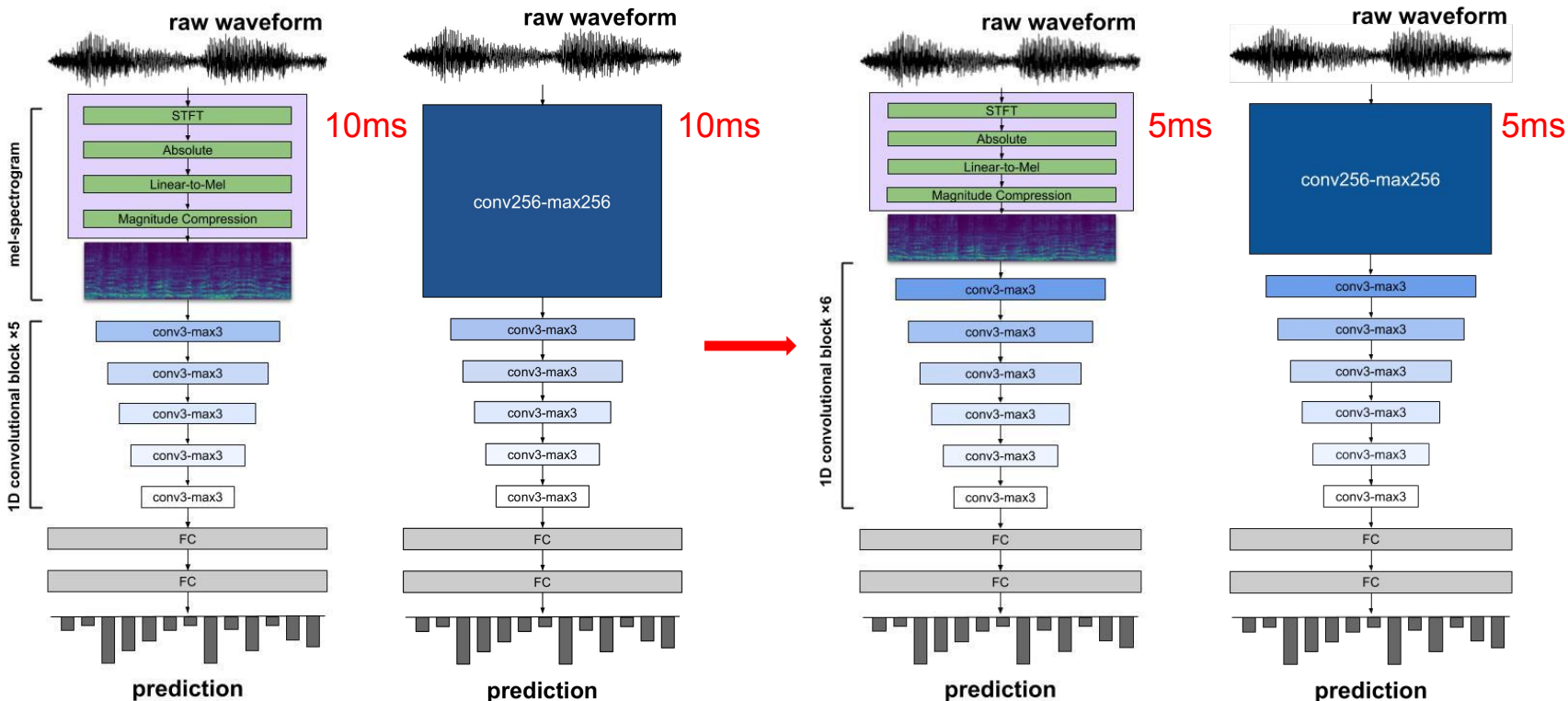


Outline

- Problem statement
- Audio classification models
 - *Engineered feature based model*
 - *Frame-level waveform based model*
 - ***Sample-level waveform based model***
- Connect signal processing and deep learning
- Summary

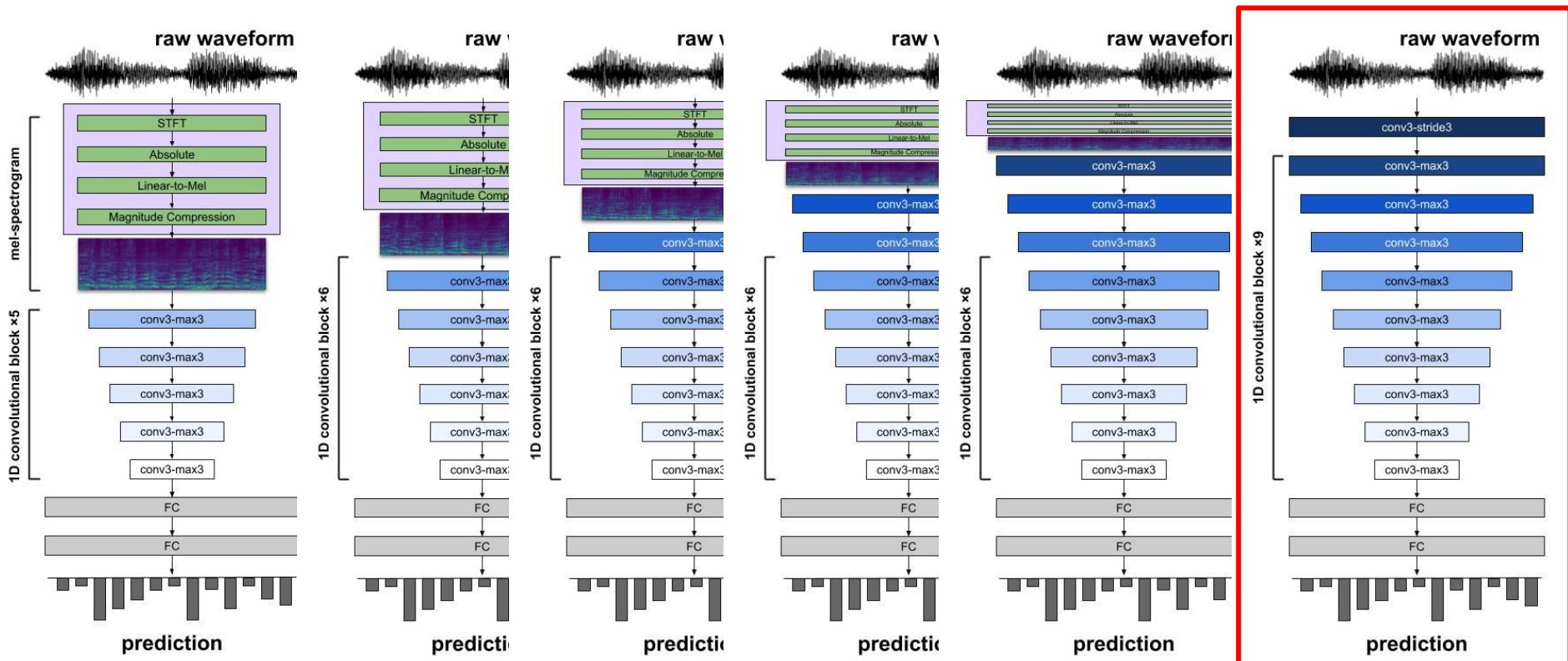
Connect 1D CNN model to sample-level model

by reducing the sizes of window/filter and hop/stride, and, accordingly, increasing the number of convolutional blocks.

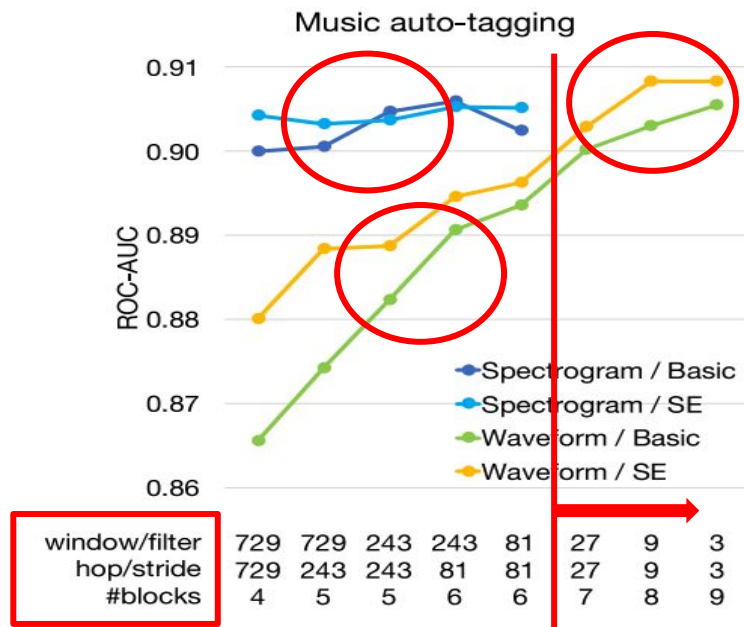


Connect 1D CNN model to sample-level model

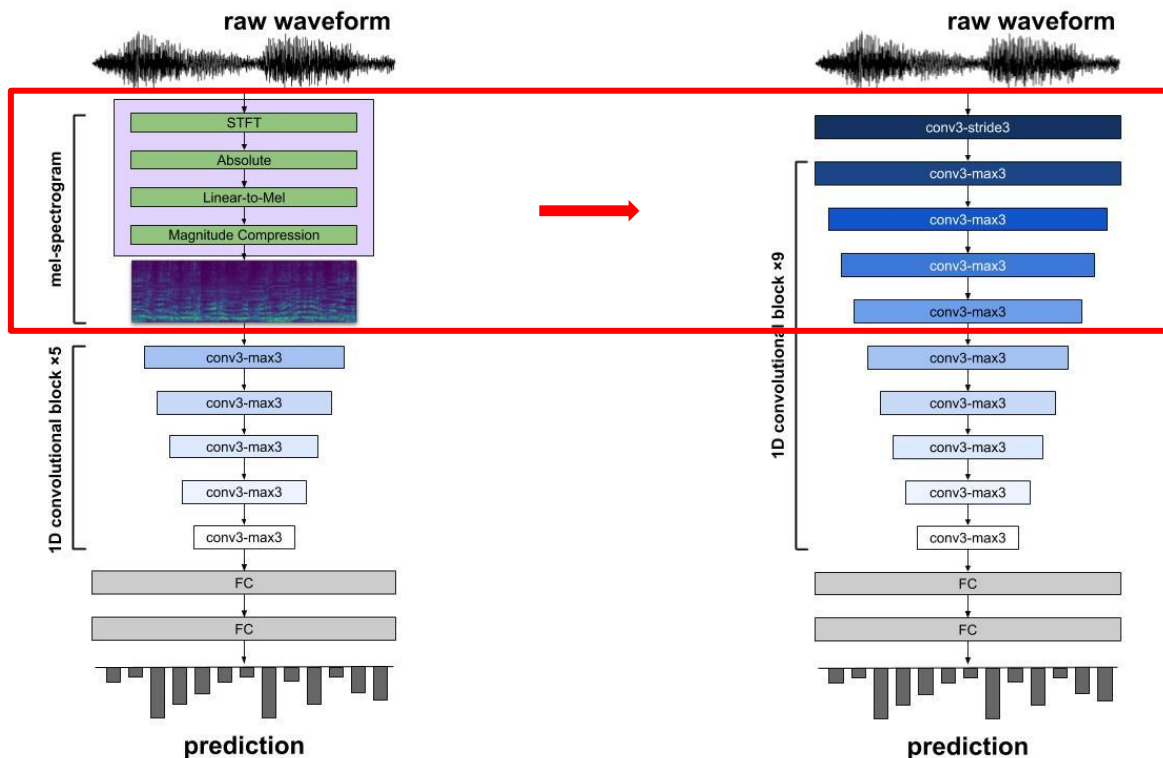
by reducing the sizes of window/filter and hop/stride, and, accordingly, increasing the number of convolutional blocks.



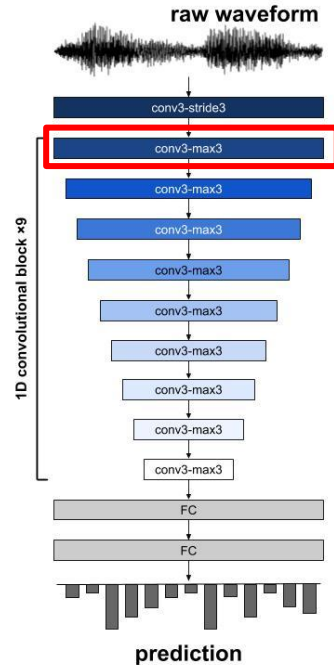
Connect 1D CNN model to sample-level model



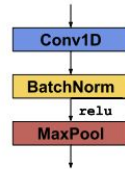
Sample-level waveform based model



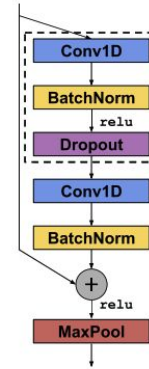
More building blocks



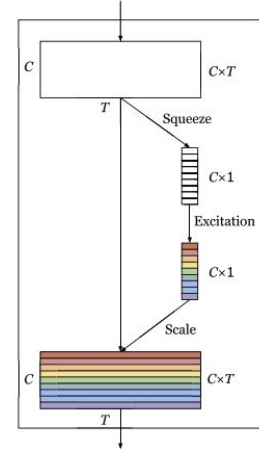
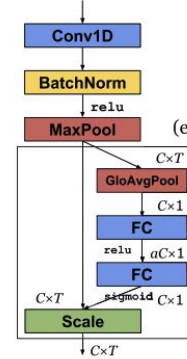
Basic



Res



SE



Outline

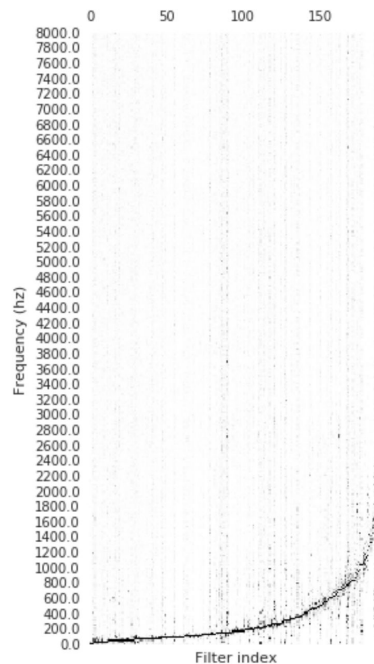
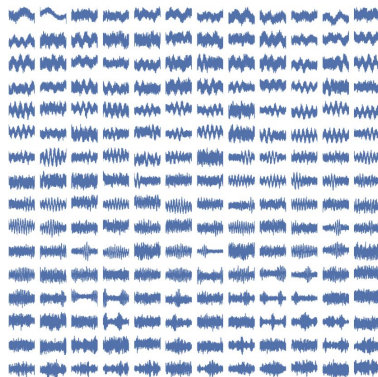
- Problem statement
- Audio classification models
 - **Engineered feature** based model
 - **Frame-level waveform** based model
 - **Sample-level waveform** based model
- Connect signal processing and deep learning
- Summary

Outline

- Problem statement
- Audio classification models
 - *Engineered feature based model*
 - *Frame-level waveform based model*
 - *Sample-level waveform based model*
- **Connect signal processing and deep learning**
- Summary

Learned filter visualization

First convolutional layer filter of frame-level waveform model

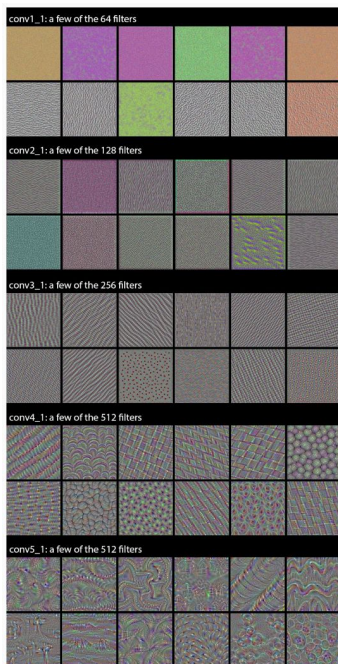


Dieleman and Schrauwen, 2014. "End-to-end learning for music audio", *ICASSP*.

Ardila et al., 2016. "Audio deepdream: Optimizing raw audio with convolutional networks", *ISMIR LBD*.

Learned filter visualization

Activation Maximization

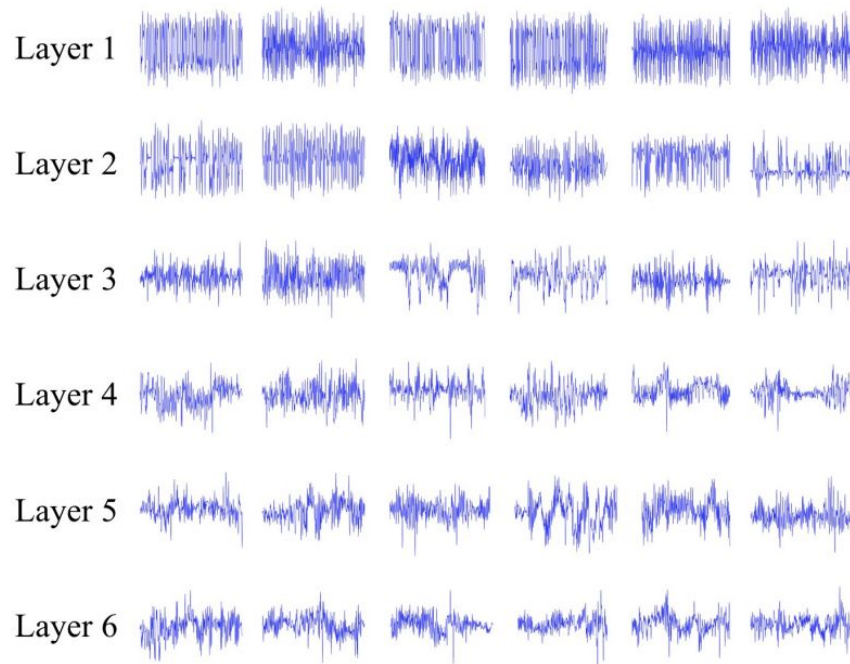
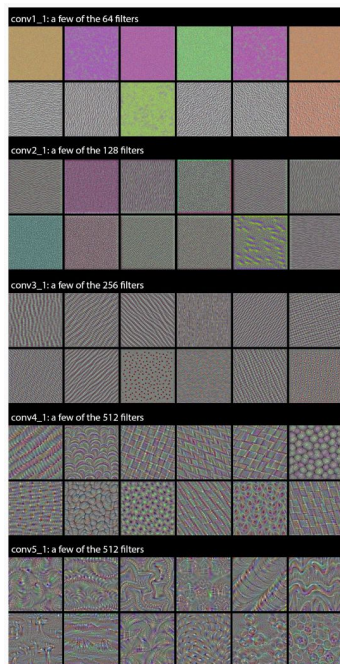


1. Generate random waveform
2. Feed-forward to the target filter
3. Obtain the gradient of the input layer (waveform)
4. Update waveform
5. Repeat

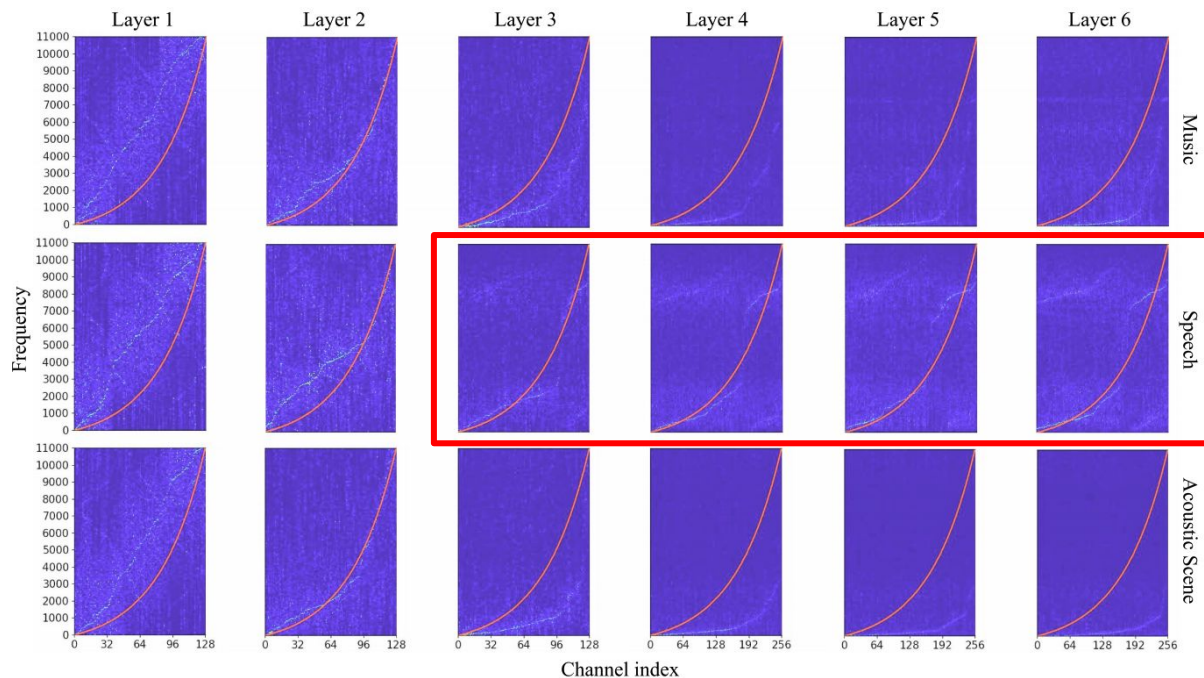
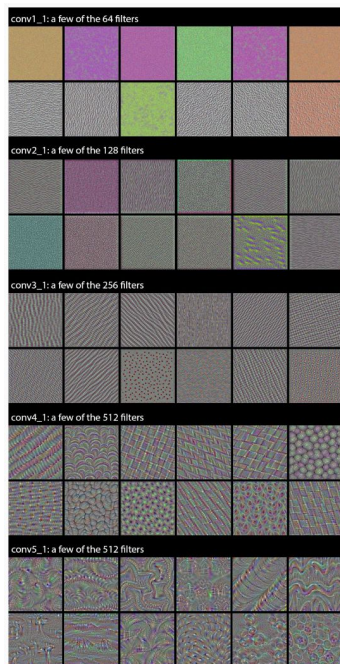
Image borrowed from "<https://blog.keras.io/how-convolutional-neural-networks-see-the-world.html>"

Erhan et al., 2009. "Visualizing higher-layer features of a deep network", *University of Montreal*.

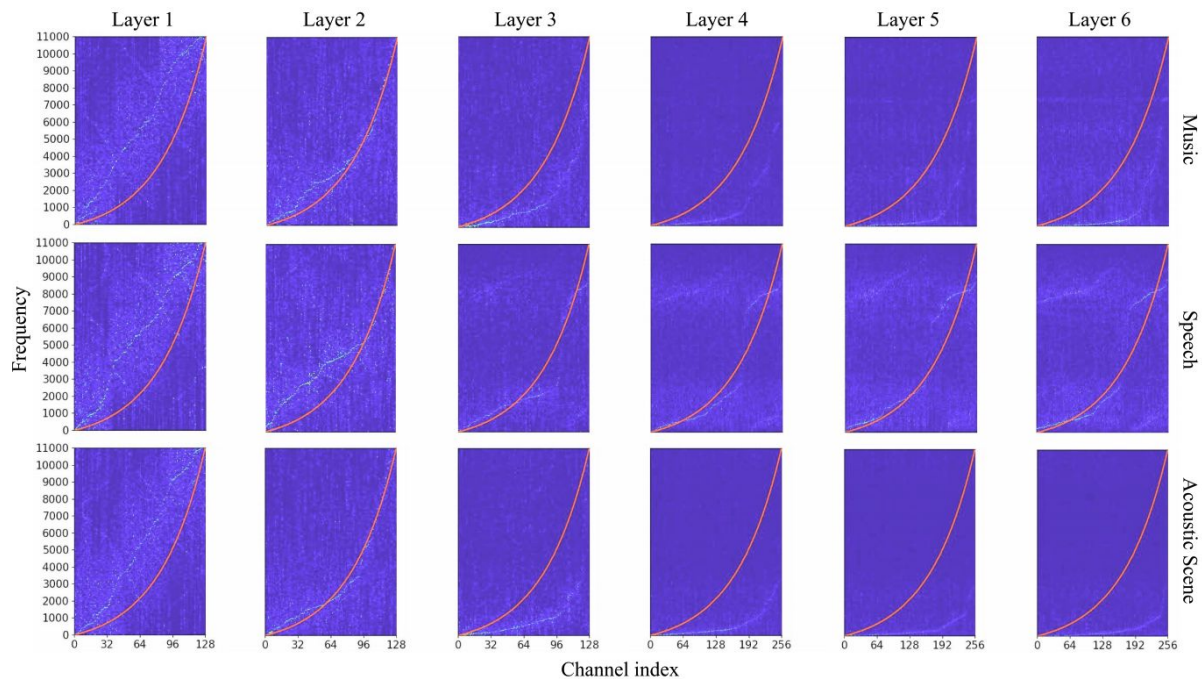
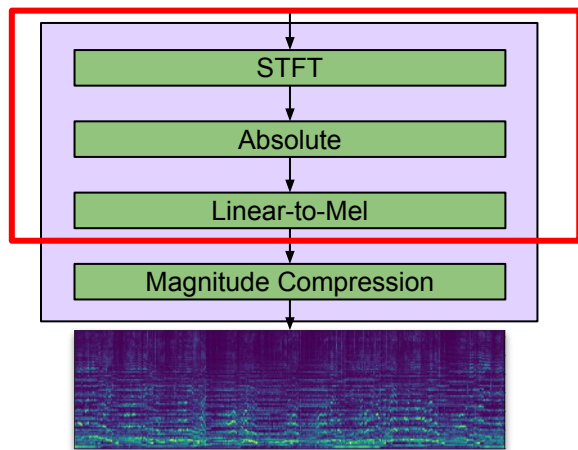
Learned filter visualization



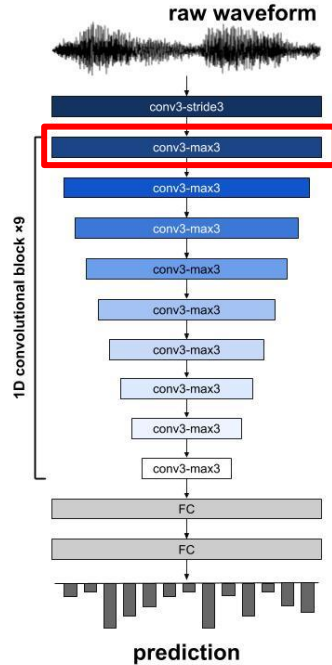
Learned filter visualization



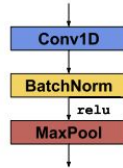
Learned filter visualization



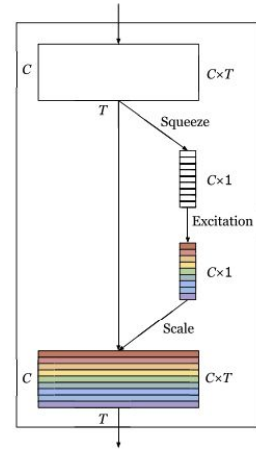
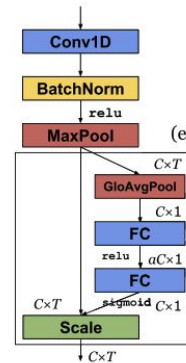
Extended Architecture



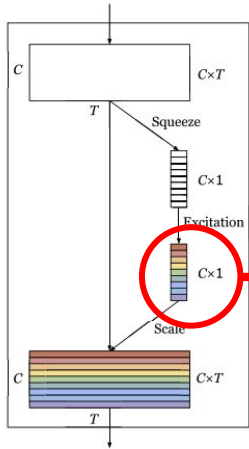
Basic



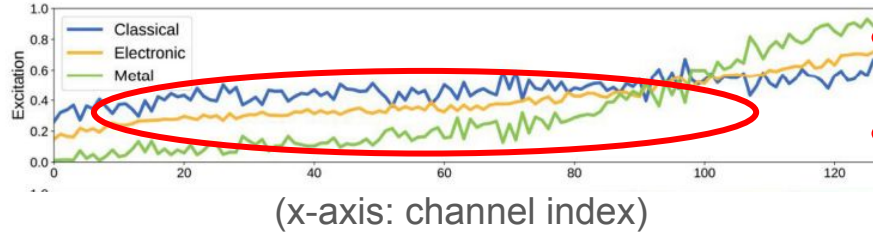
SE



Excitation Analysis



Averaged excitation value of the first SE block by class.



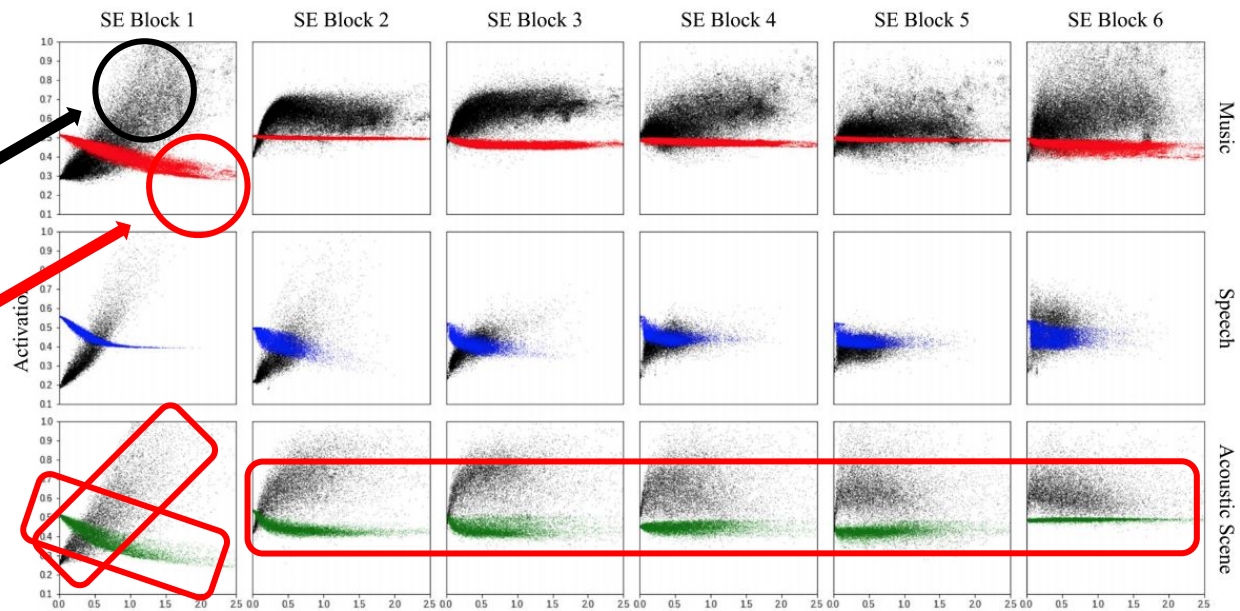
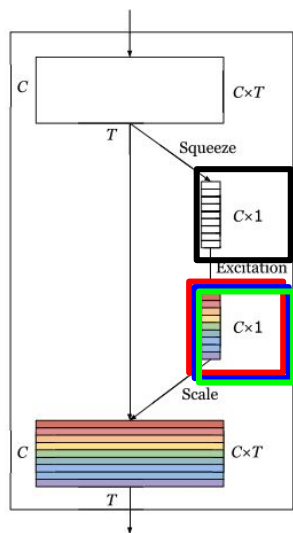
Averaged relative loudness by class.



Excitation Analysis

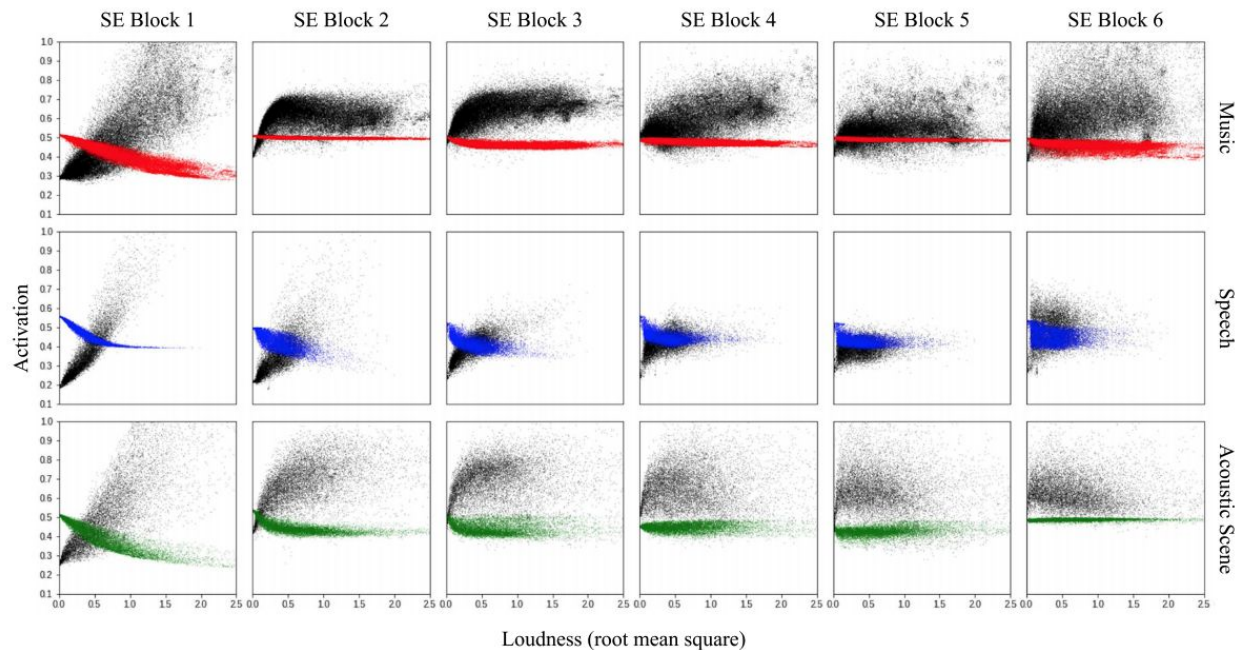
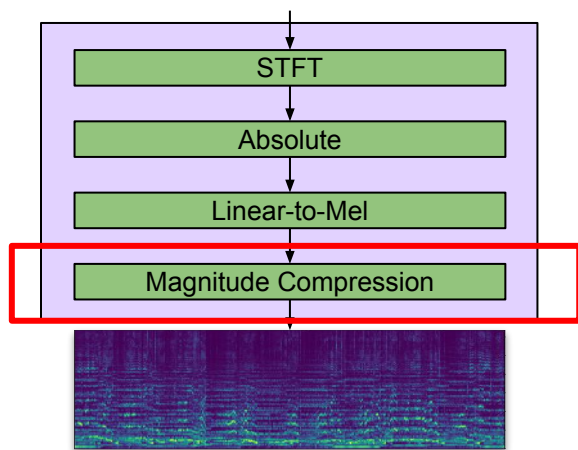
Colored : Excitation value

Black : Averaged channel magnitude



x-axis : loudness (root mean square)

Excitation Analysis

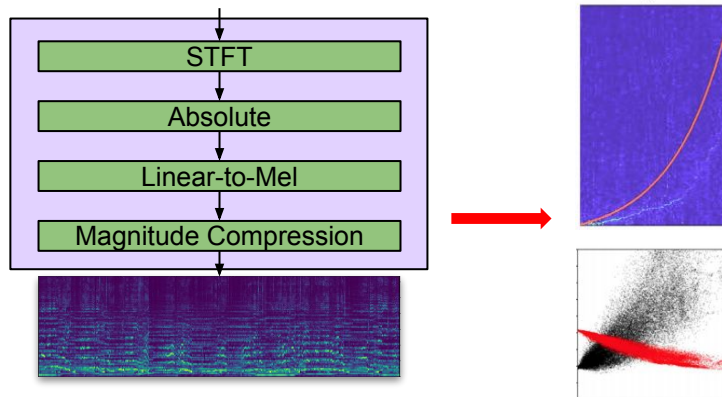


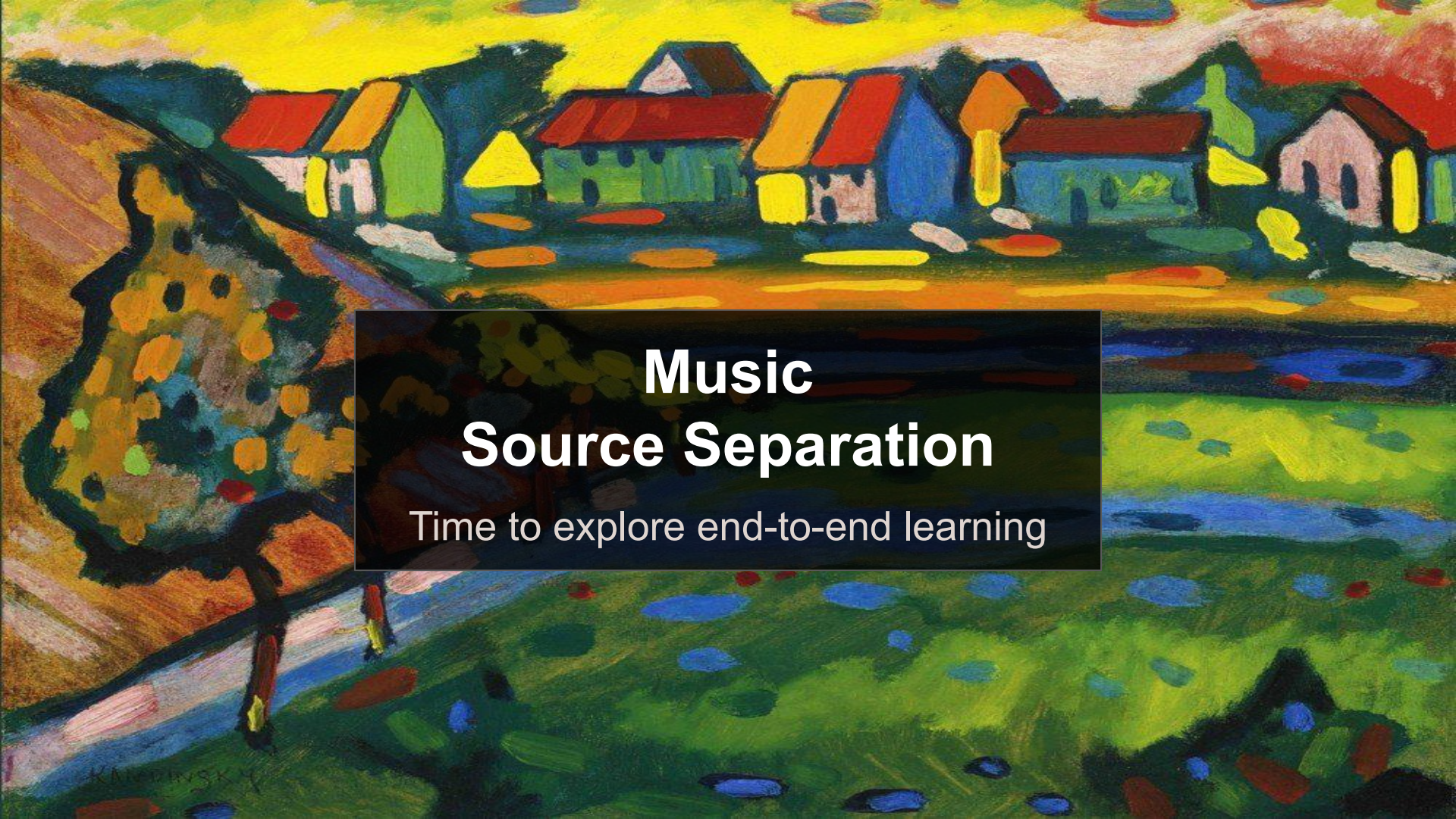
Outline

- Problem statement
- Audio classification models
 - *Engineered feature based model*
 - *Frame-level waveform based model*
 - *Sample-level waveform based model*
- Connect signal processing and deep learning
- **Summary**

Summary

- Music classification
- Audio classification models
- Waveform based model analysis

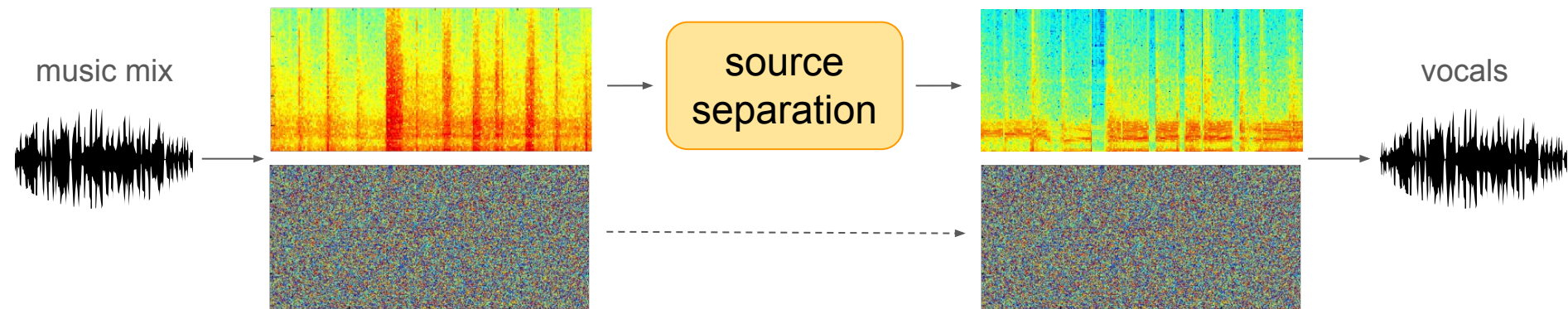




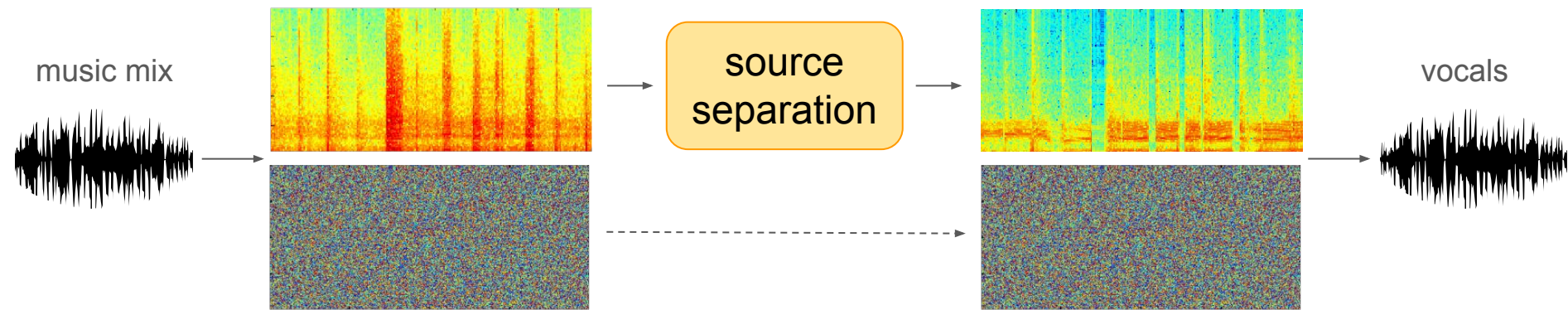
Music Source Separation

Time to explore end-to-end learning

Why end-to-end music source separation?



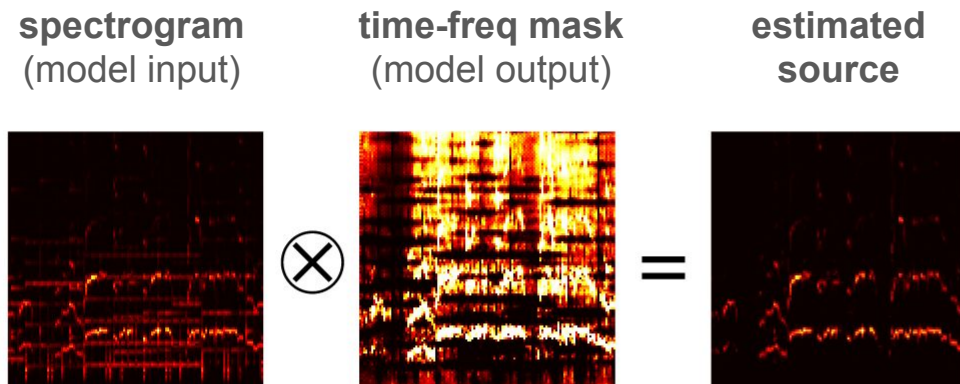
Why end-to-end music source separation?



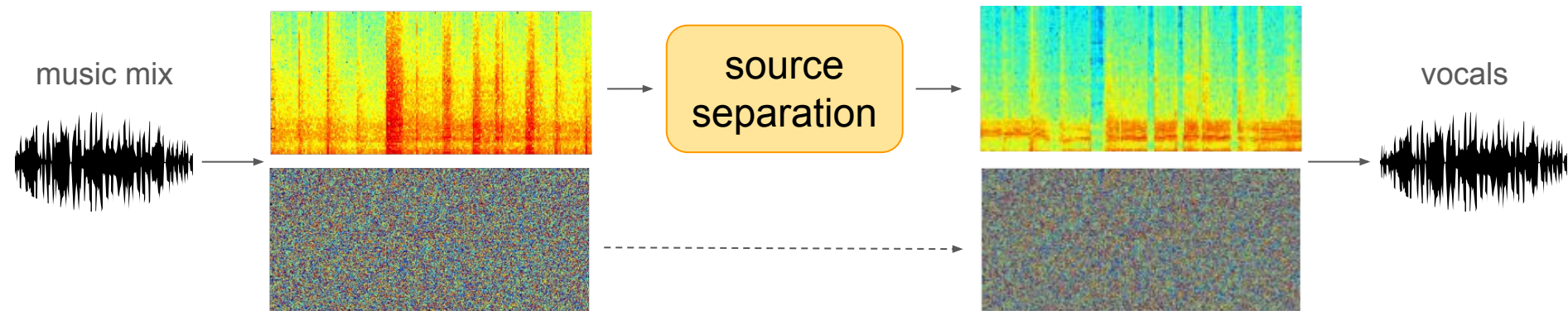
I) Are we missing crucial information when **discarding the phase**?

II) When using the **phase of the mixture at synthesis time**, are we introducing artifacts that are limiting our model's performance?

Why filtering spectrograms with masks?



III) **Filtering** spectrograms does **not allow recovering masked (perceptually hidden sound) signals**



- I) Are we missing crucial information when **discarding the phase**?
- II) When using the **phase of the mixture at synthesis time**, are we introducing artifacts that are limiting our model's performance?
- III) **Filtering** spectrograms does **not allow recovering masked signals**

End-to-end music source separation

music mix



deep
learning



vocals



Other (active) research directions:

Use the complex STFT as i/o interface?

Kameoka et al., 2009. “ComplexNMF: A new sparse representation for acoustic signals” in ICASSP.

Dubey et al., 2017. “Does phase matter for monaural source separation?” in arXiv.

Le Roux et al., 2019. “Phasebook and friends: Leveraging discrete representations for source separation” in IEEE Journal of Selected Topics in Signal Processing.

Tan et al., 2019. “Complex Spectral Mapping with a CRNN for Monaural Speech Enhancement” in ICASSP.

Liu et al., 2019. “Supervised Speech Enhancement with Real Spectrum Approximation” in ICASSP.

Other (active) research directions:
Alternative models at synthesis time?

Virtanen and Klapuri, 2000. “Separation of harmonic sound sources using **sinusoidal modeling**,” in ICASSP.

Chandna et al., 2019. “A **vocoder** based method for singing voice extraction” in ICASSP.

End-to-end music source separation

music mix



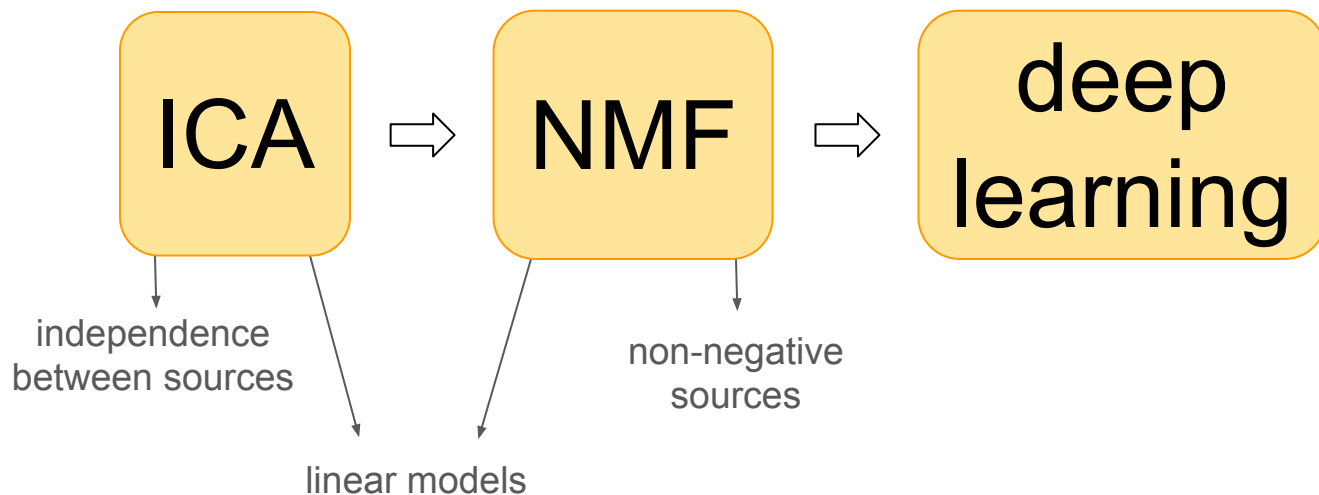
deep
learning



vocals



Historical perspective: unsupervised & linear models



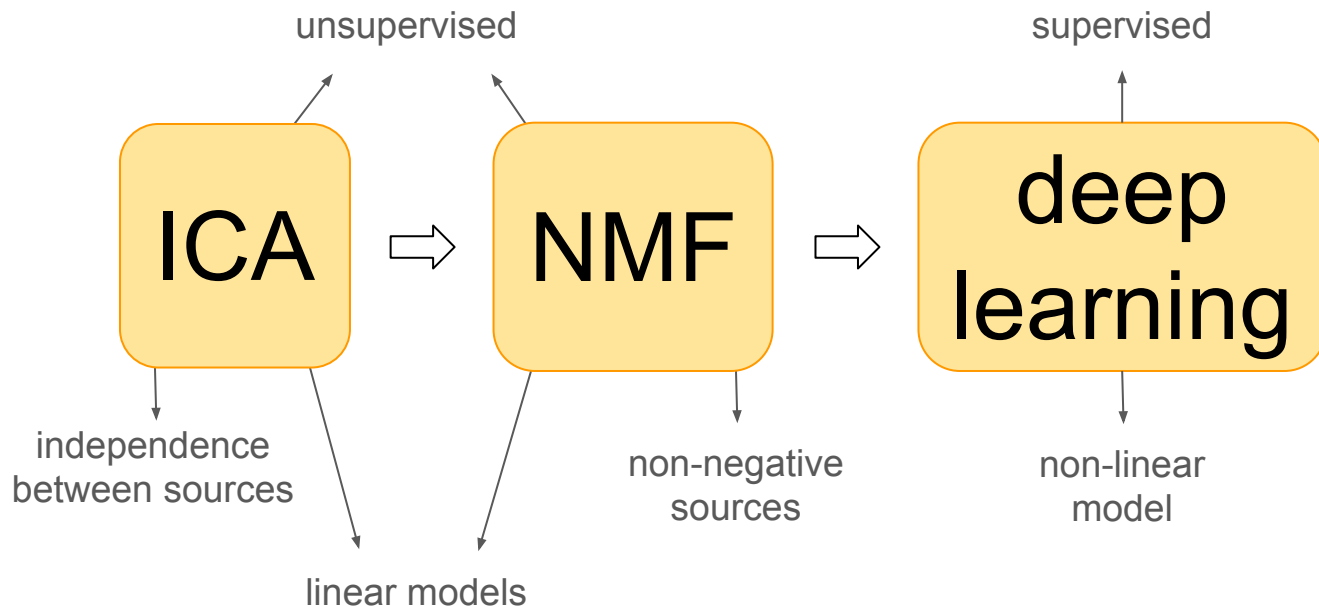
Linear model example

$$X \approx \hat{X} \stackrel{\text{linear approximation}}{=} \sum_k w_k h_k \stackrel{\text{activations}}{=} WH$$

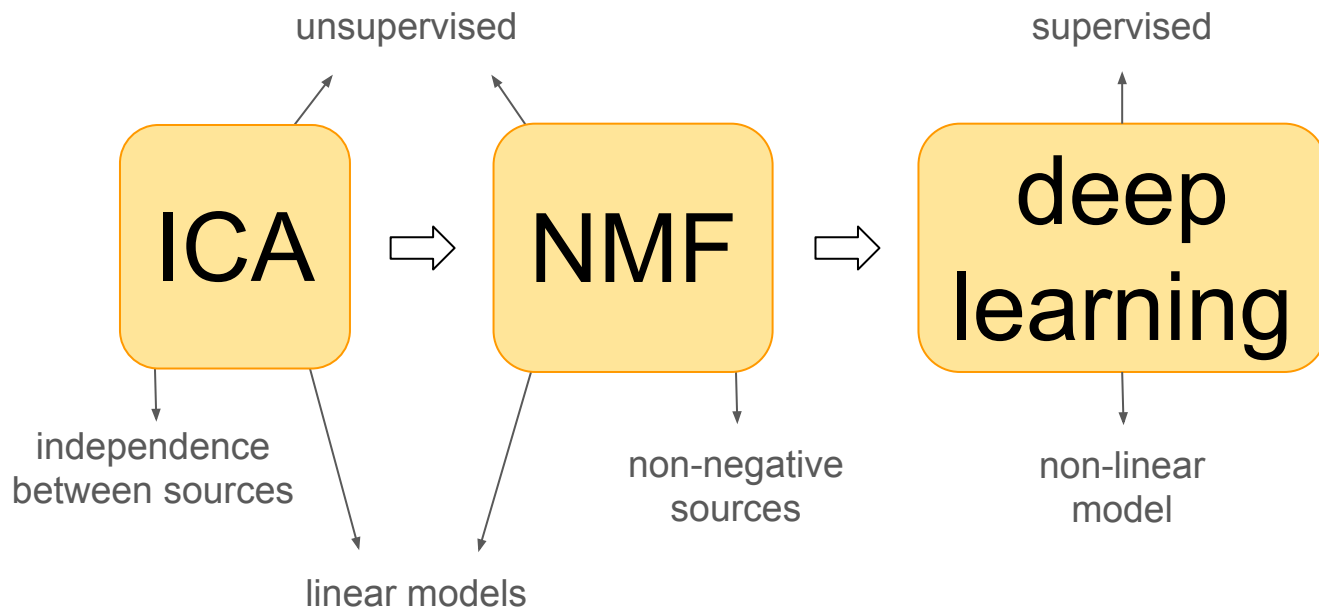
↑
bases

Unsupervised factorization of the mixture
into **bases** (w) and **activations** (h)

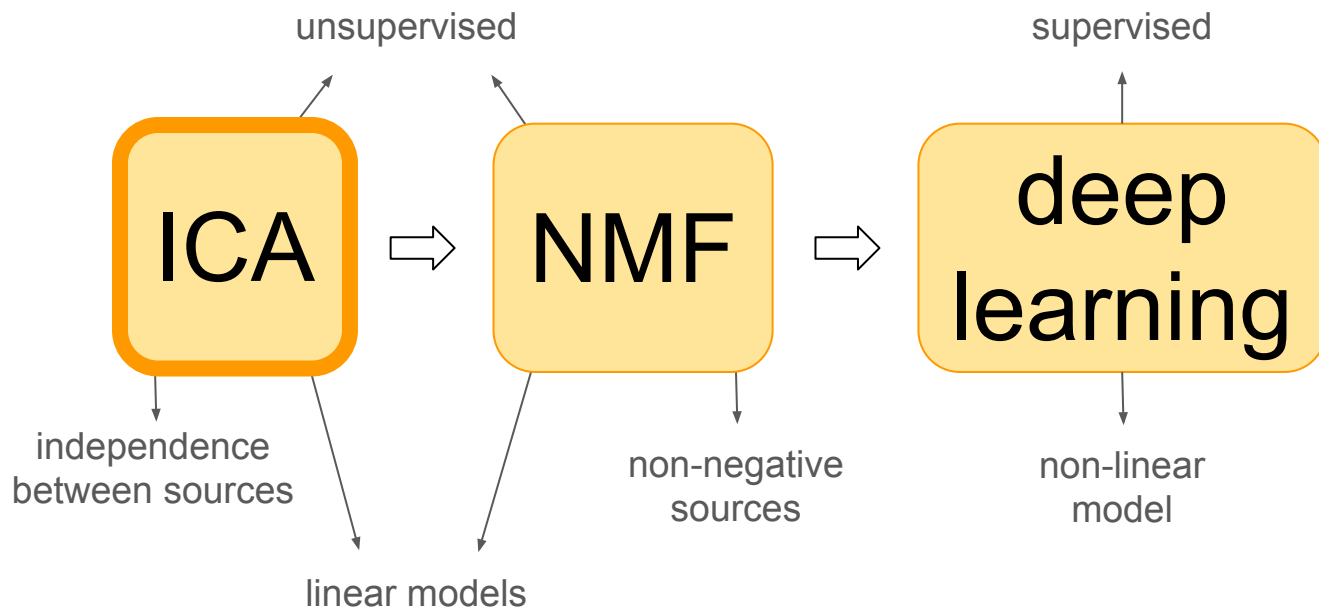
Historical perspective: unsupervised & linear models



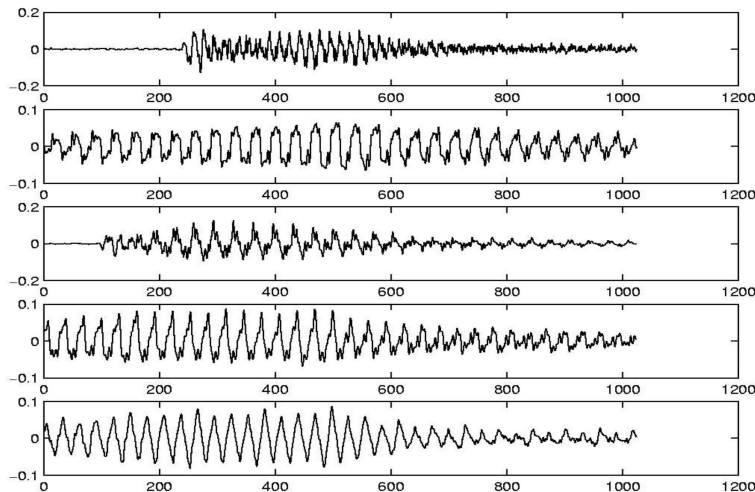
Historical perspective: waveform-based models?



Historical perspective: waveform-based models?



waveform-based ICA



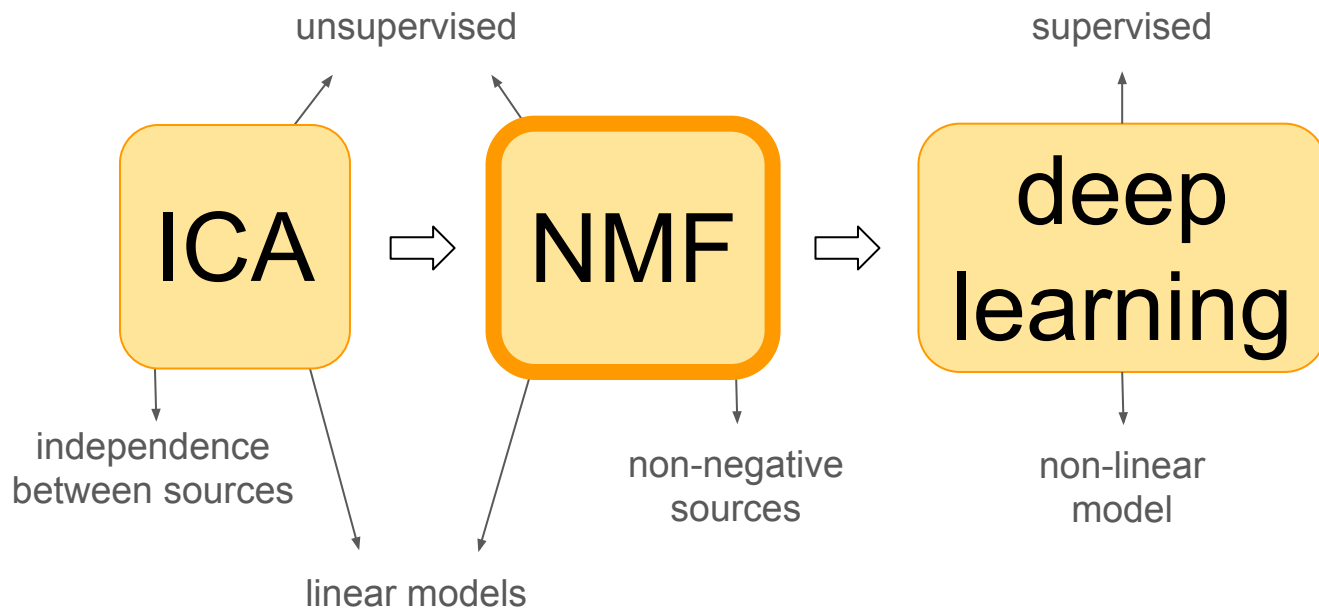
$$X \approx \hat{X} = \sum_k w_k h_k = WH$$

bases

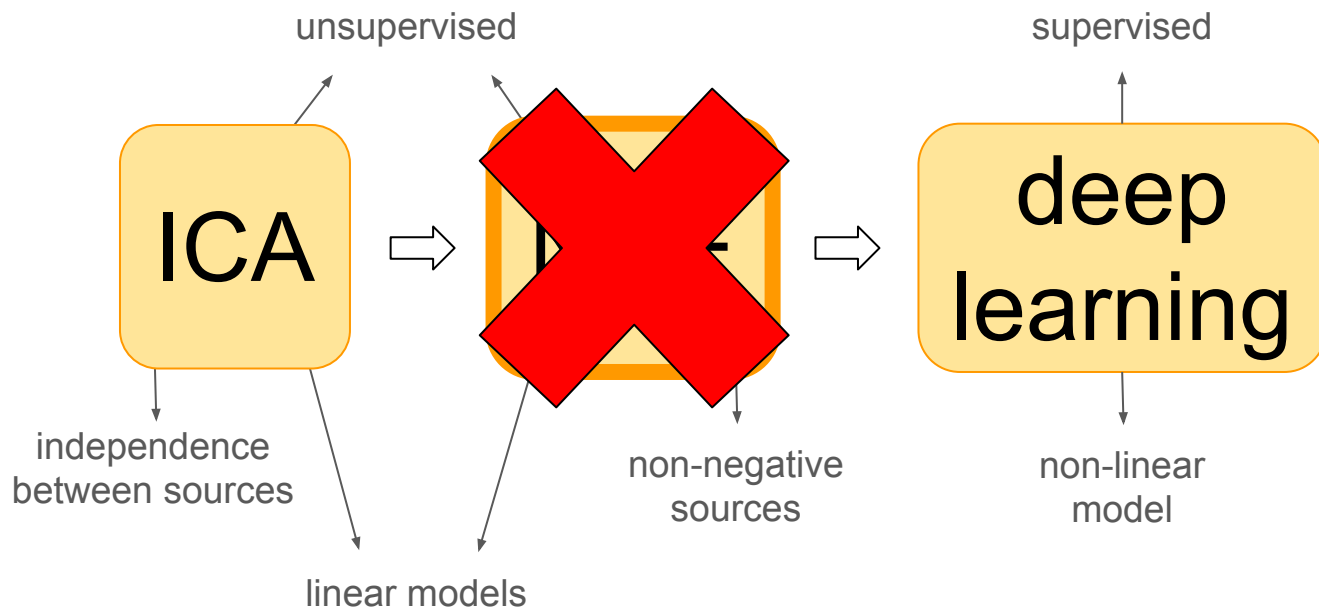
activations

Problem 1: phase sensitive basis
Problem 2: simplicity of the linear model

Historical perspective: waveform-based models?

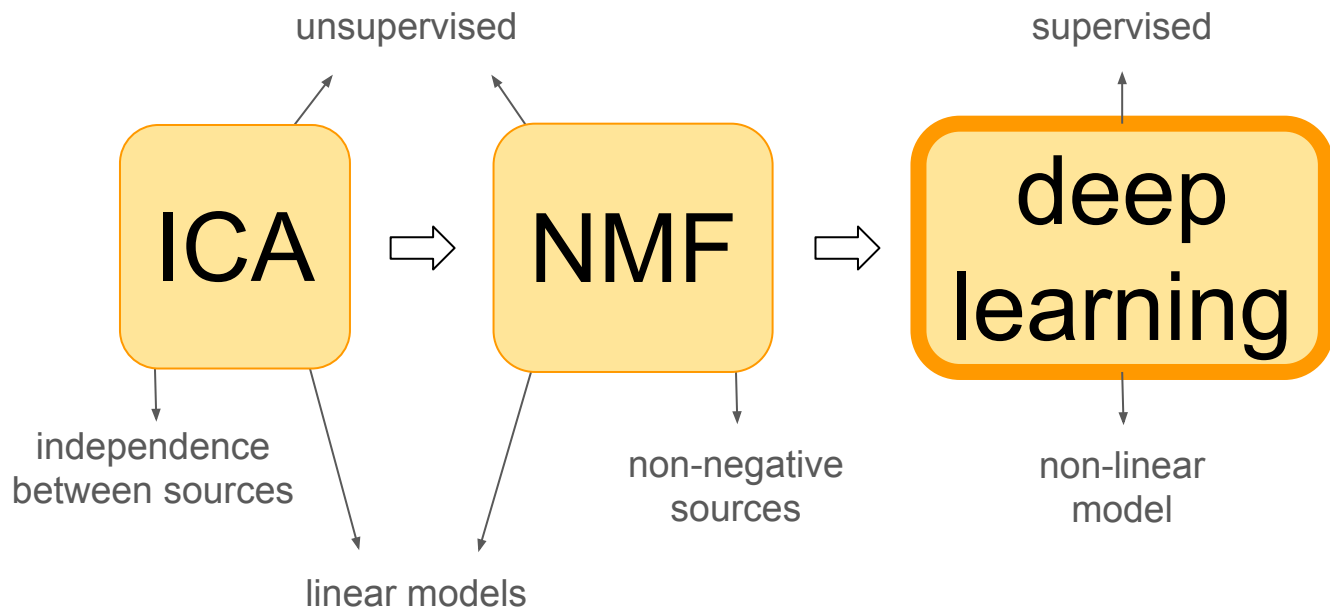


Historical perspective: waveform-based models?



NMF cannot be used with waveforms due to its non-negative constraint! (waveforms range from -1 to 1)

Historical perspective: waveform-based models?



A widely-used set of tools:

filtering spectrograms

linear models

unsupervised learning

audio domain knowledge

..maybe we could try another toolset?

~~filtering~~ → synthesis?

~~linear models~~ → non-linear models?

~~unsupervised learning~~ → supervised learning?

~~audio domain knowledge~~ → data driven?

End-to-end music source separation: 9 publications

Stoller et al., 2018. “Wave-u-net: A multi-scale neural network for end-to-end audio source separation” in arXiv.

Grais et al., 2018. “Raw Multi-Channel Audio Source Separation using Multi-Resolution Convolutional Auto-Encoders” in EUSIPCO.

Lluis, et al., 2018. “End-to-end music source separation: is it possible in the waveform domain?” in arXiv.

Slizovskaia et al., 2018. “End-to-end Sound Source Separation Conditioned on Instrument Labels” in arXiv.

Cohen-Hadria et al., 2019. “Improving singing voice separation using Deep U-Net and Wave-U-Net with data augmentation” in arXiv.

Kaspersen, 2019. “HydraNet: A Network For Singing Voice Separation”. Master Thesis.

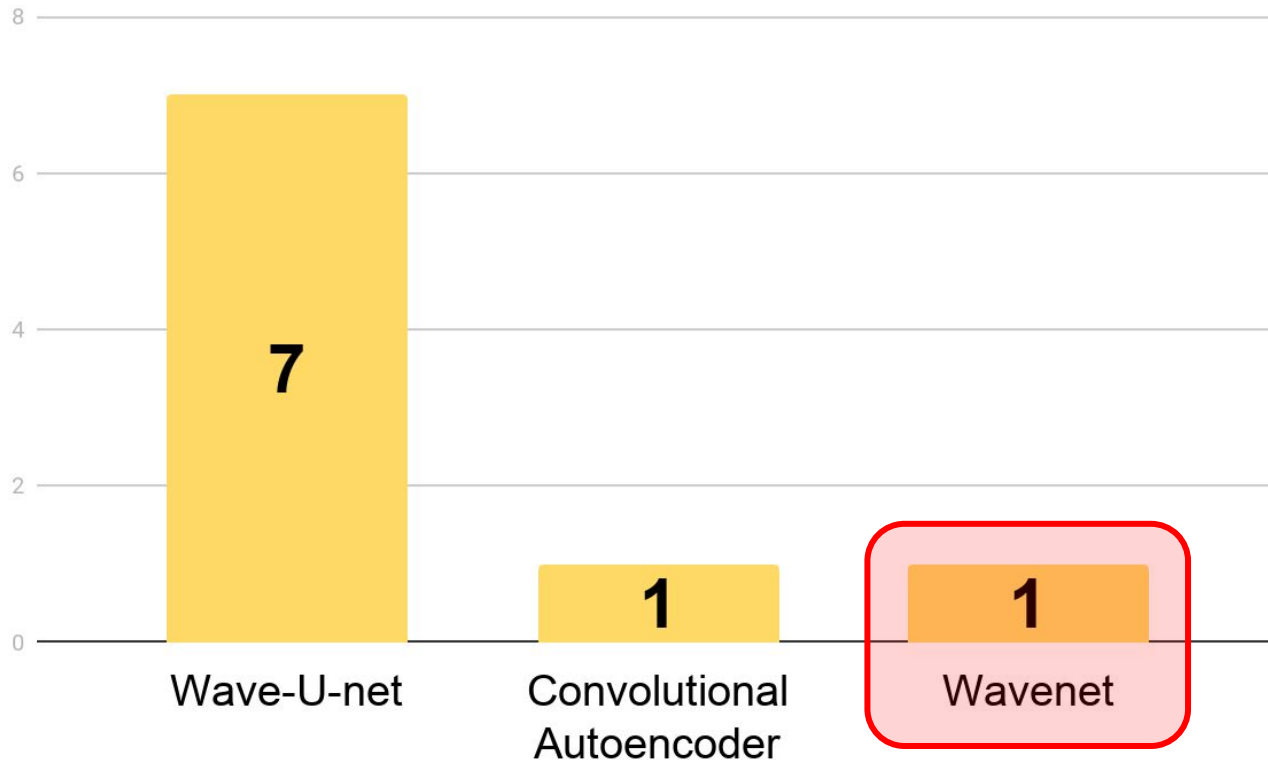
Akhmetov et al., 2019. “Time Domain Source Separation with Spectral Penalties”. Technical Report.

Défossez et al., 2019. “Demucs: Deep Extractor for Music Sources with extra unlabeled data remixed” in arXiv.

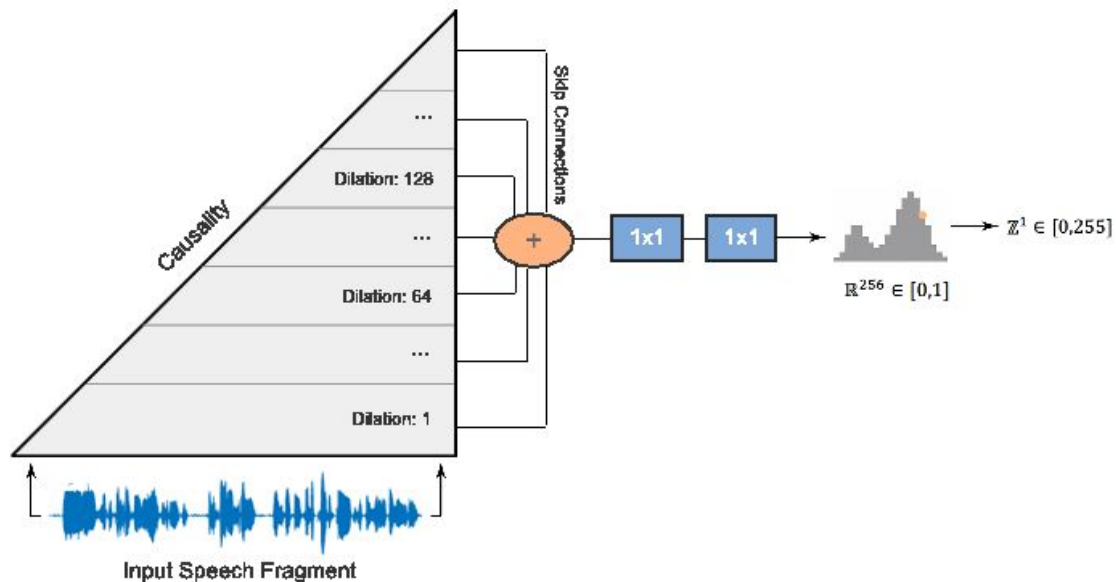
Narayanaswamy et al., 2019. “Audio Source Separation via Multi-Scale Learning with Dilated Dense U-Nets” in arXiv.

ALL THE PUBLICATIONS IN CHRONOLOGICAL ORDER AS OF OCTOBER 2019

End-to-end music source separation: architectures



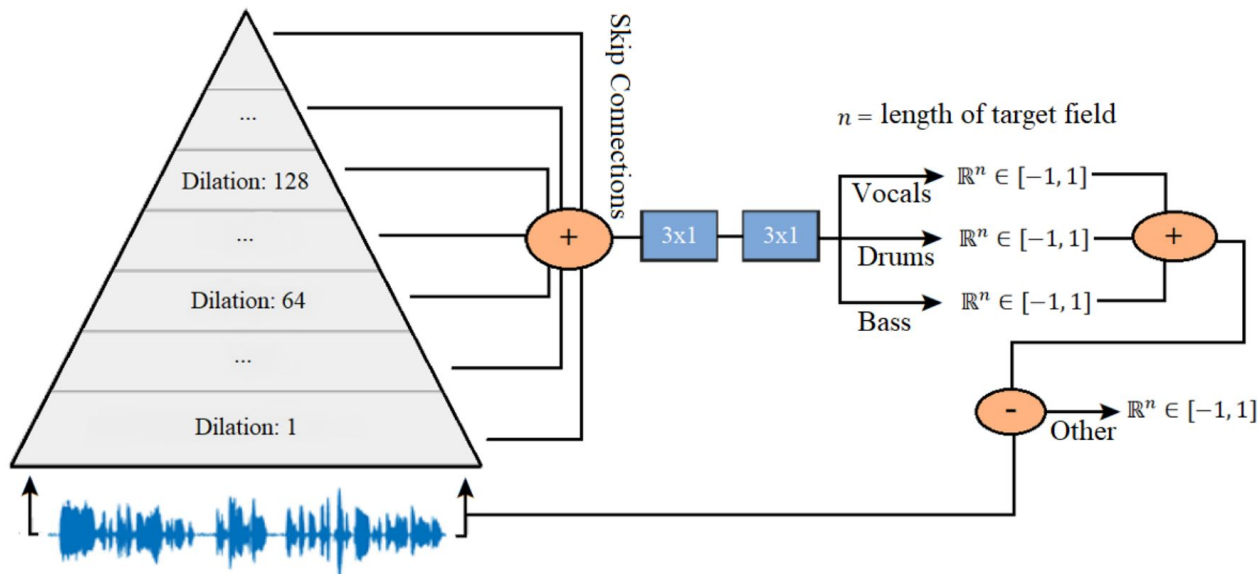
Introduction: the “generative” Wavenet



Causal

Softmax-output
distribution
modeling $p(x)$

A “regression” Wavenet for music source separation

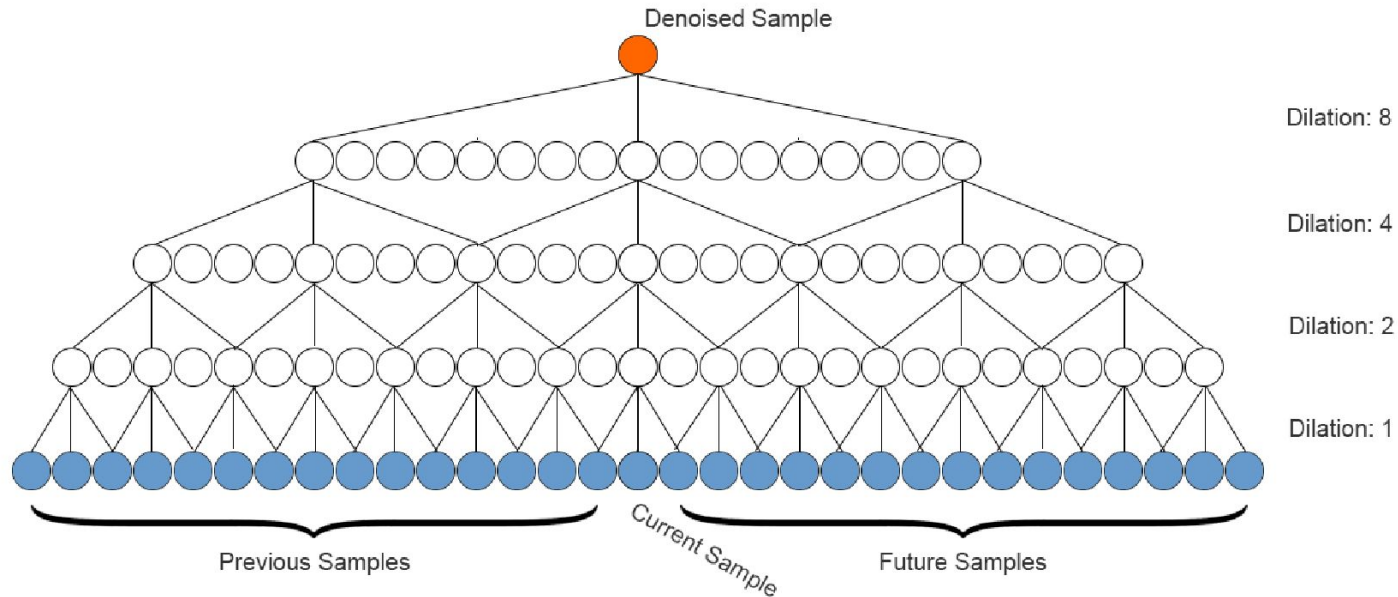


Mixture Fragment

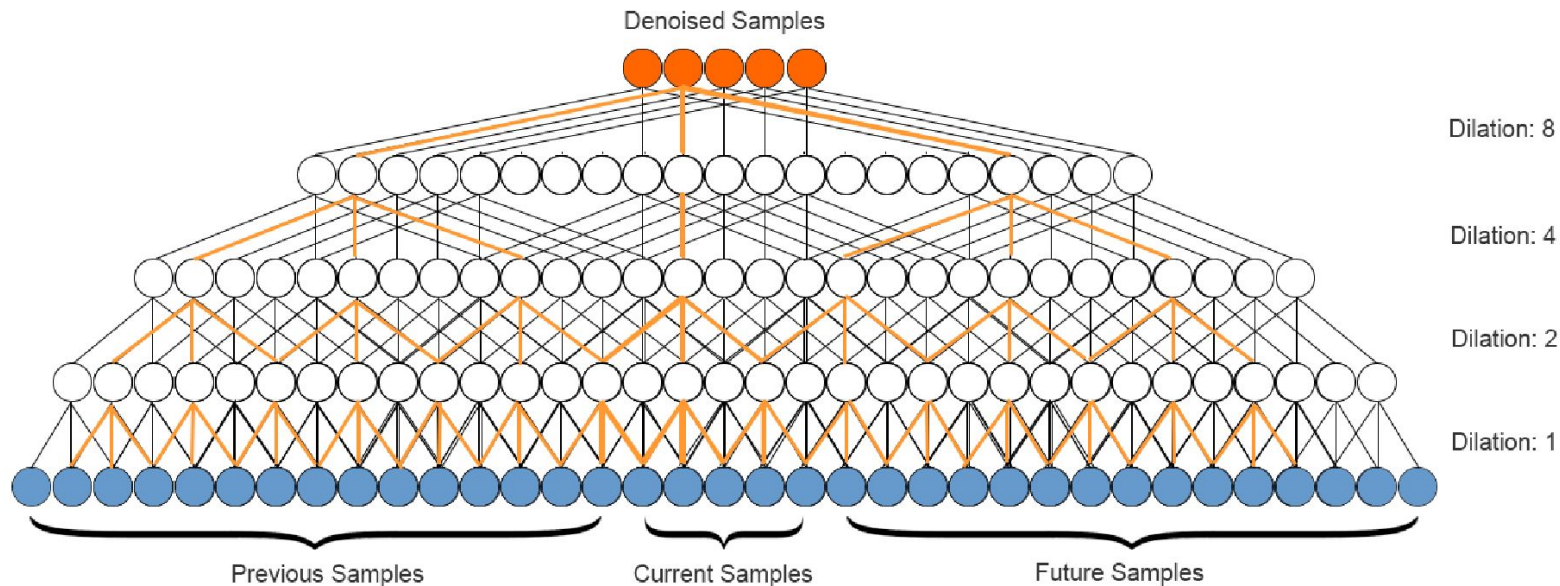
Non-causal

Regression output
 $p(y|x)$ *discriminative*
post-processing!

Fully convolutional & deterministic



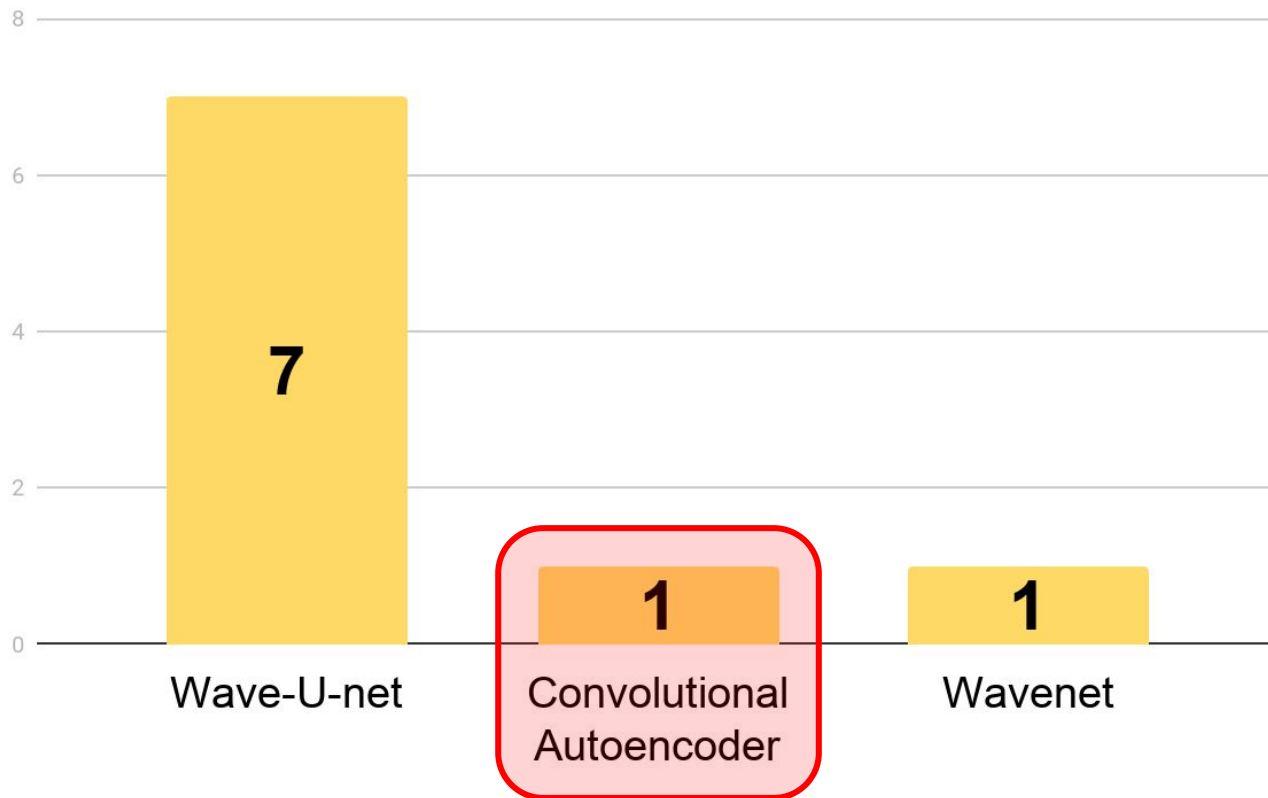
Fully convolutional & deterministic



Real time inference!

1601 samples input \rightarrow denoising time: ≈ 0.56 sec per second of music on GPU!

End-to-end music source separation: architectures



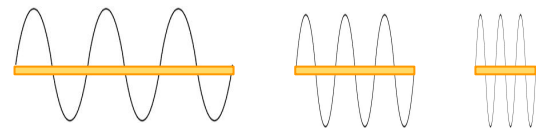
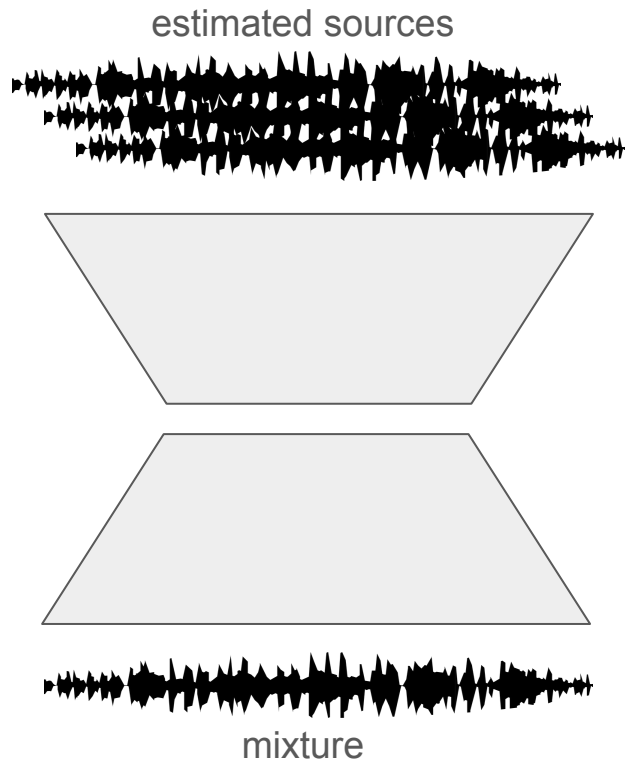
Autoencoders

estimated sources



mixture

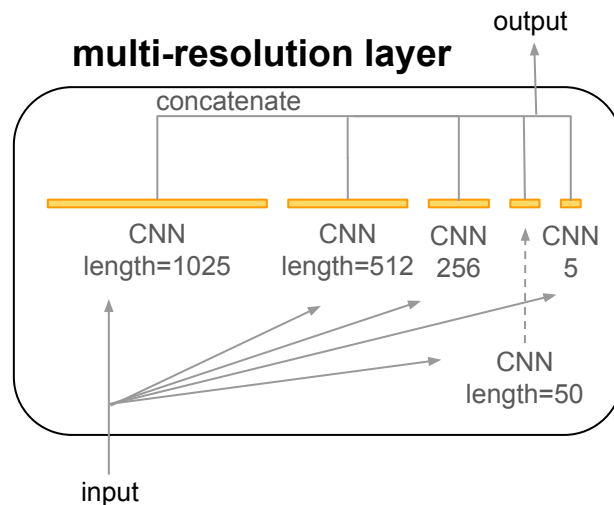
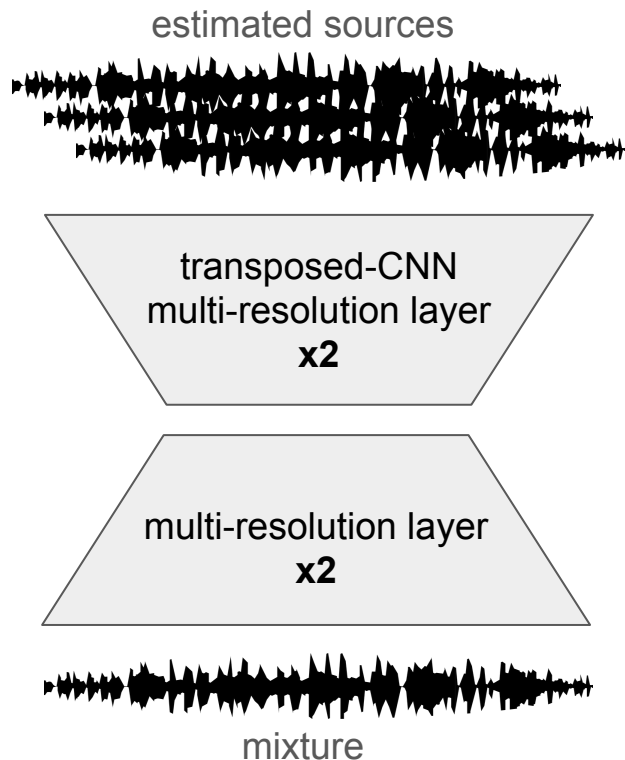
Multi-resolution & Convolutional autoencoder



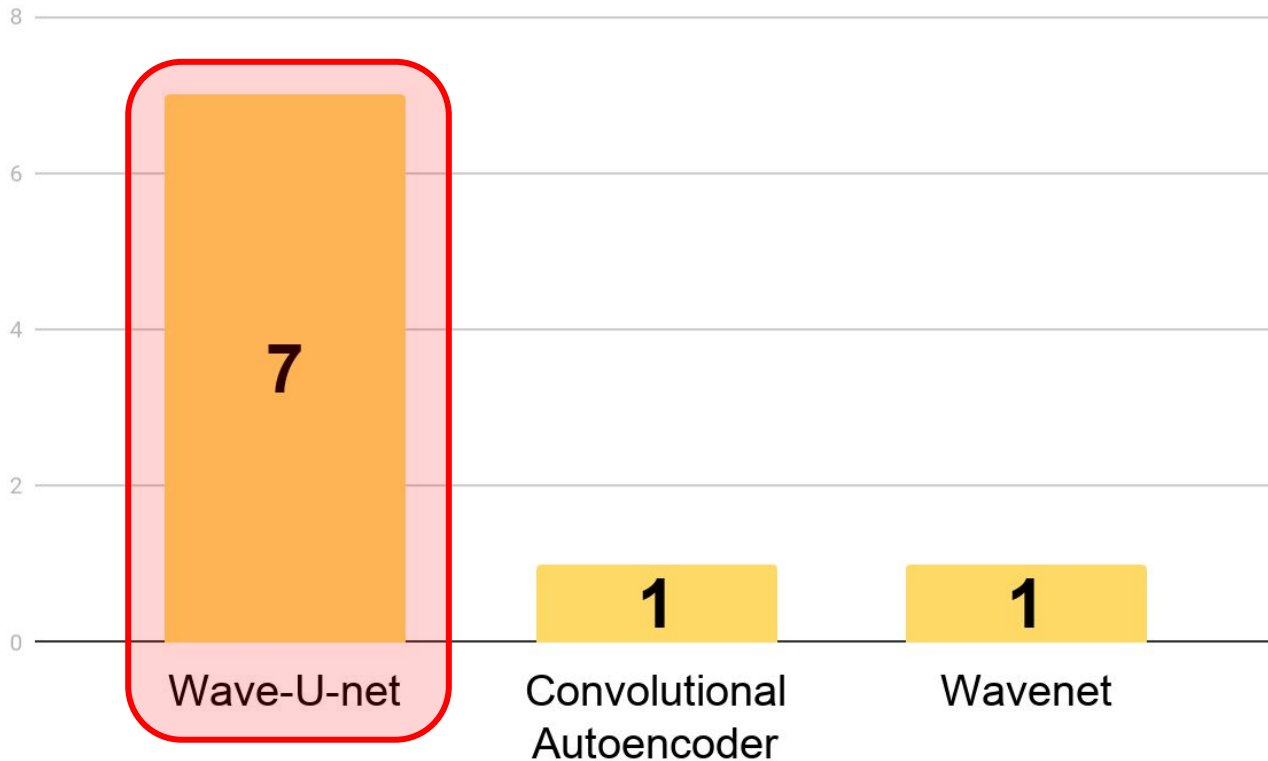
Multi-resolution CNN: efficient way to represent 3 periods!

Multi-resolution CNN = Inception CNN
(different filter shapes in the same CNN layer)

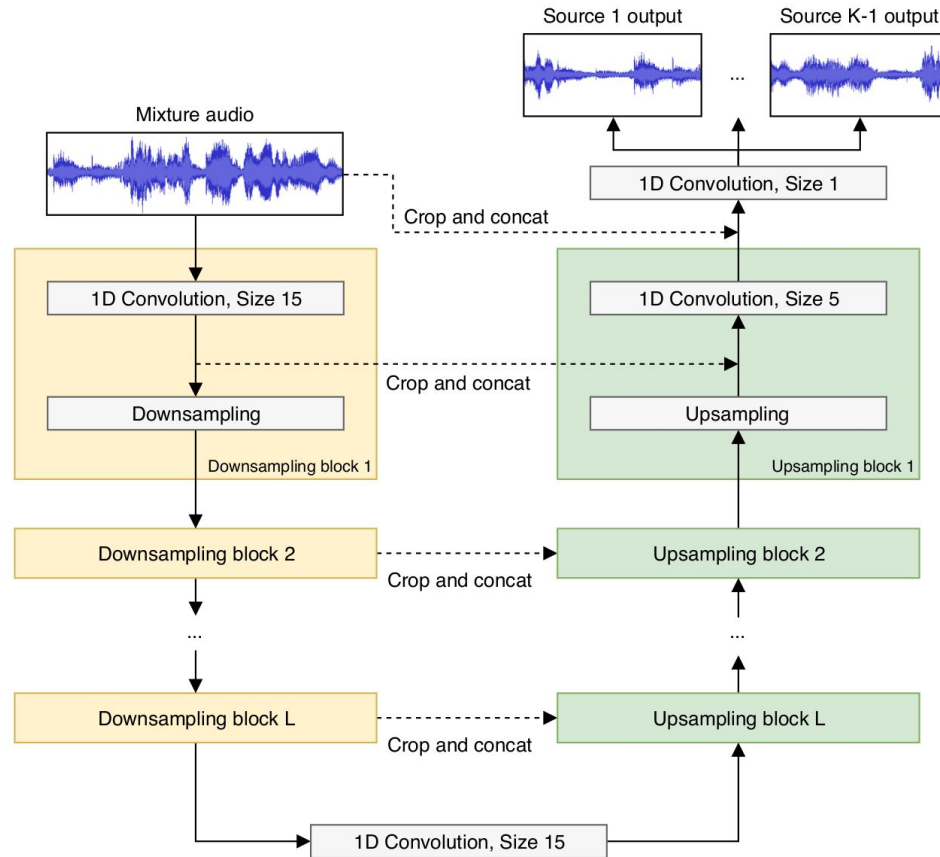
Multi-resolution & Convolutional autoencoder



End-to-end music source separation: architectures



Wave-U-net



Wave-u-net extensions

- Multiplicative conditioning using instrument labels at the bottleneck.

Slizovskaia et al., 2019. “End-to-end Sound Source Separation Conditioned on Instrument Labels” in ICASSP.

- Data augmentation: ≈ 1 dB SDR improvement.

Cohen-Hadria et al., 2019. “Improving singing voice separation using Deep U-Net and Wave-U-Net with data augmentation” in arXiv.

Data augmentation strategies

It is used to **artificially expand the size of a training dataset** by creating modified versions of it.

- Random swapping left/right channel for each source
- Random scaling sources
- Random mixing of sources from different songs
- Pitch-shifting
- Time-stretching

Uhlich et al, 2017. "Improving music source separation based on deep neural networks through data augmentation and network blending" in ICASSP.

Cohen-Hadria et al., 2019. "Improving singing voice separation using Deep U-Net and Wave-U-Net with data augmentation" in arXiv.

Wave-u-net extensions

- **Multiplicative conditioning using instrument labels at the bottleneck.**

Slizovskaia et al., 2019. “End-to-end Sound Source Separation Conditioned on Instrument Labels” in ICASSP.

- **Data augmentation: ≈ 1 dB SDR improvement.**

Cohen-Hadria et al., 2019. “Improving singing voice separation using Deep U-Net and Wave-U-Net with data augmentation” in arXiv.

- **Add BiLSTMs at the bottleneck: ≈ 1 dB SDR improvement.**

Kaspersen, 2019. “HydraNet: A Network For Singing Voice Separation”. Master Thesis.

- **Loss function in the spectral domain.**

Akhmetov et al., 2019. “Time Domain Source Separation with Spectral Penalties”. Technical Report.

- **Use dilated convolutions and dense CNNs.**

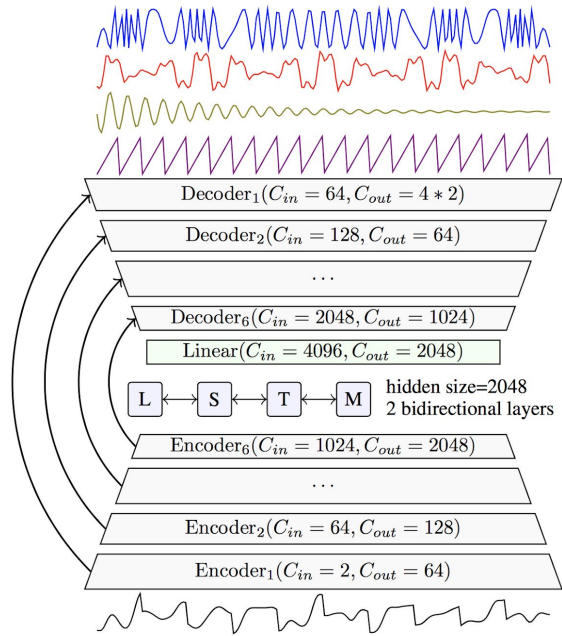
Narayanaswamy et al., 2019. “Audio Source Separation via Multi-Scale Learning with Dilated Dense U-Nets” in arXiv.

- **Achieve comparable results to a spectrogram-based model: Demucs.**

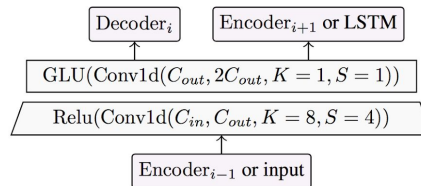
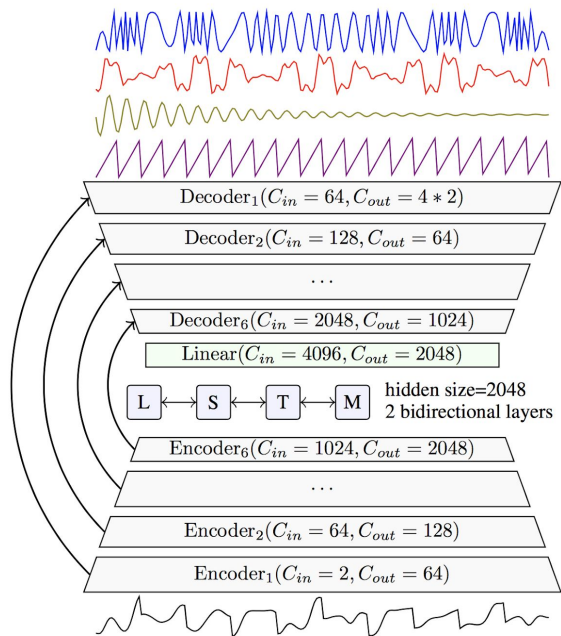
w/ BiLSTMs at the bottleneck, data augmentation, and some additional architectural changes: ≈ 1.6 dB SDR improvement.

Défossez et al., 2019. “Demucs: Deep Extractor for Music Sources with extra unlabeled data remixed” in arXiv.

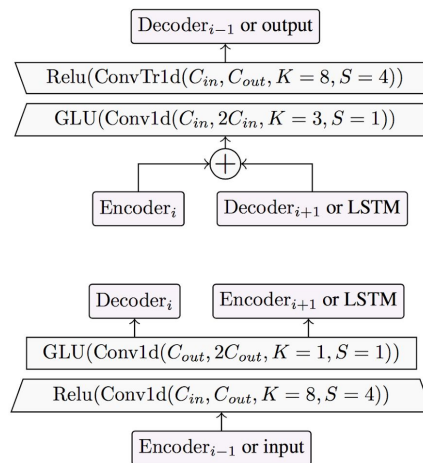
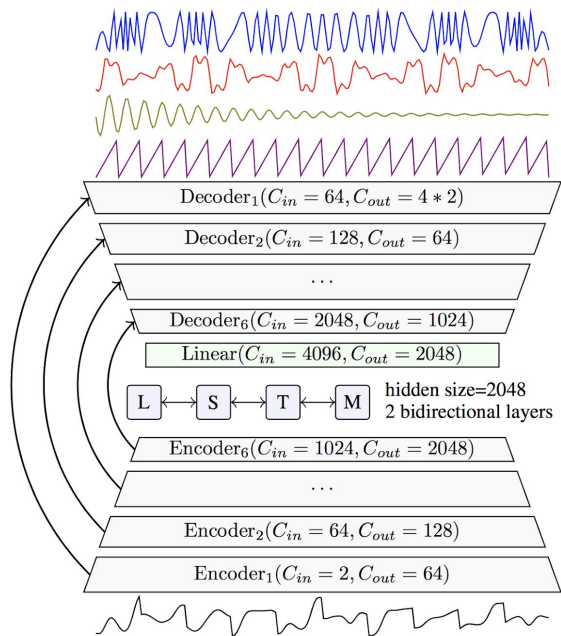
Wave-u-net extensions: Demucs



Wave-u-net extensions: Demucs



Wave-u-net extensions: Demucs

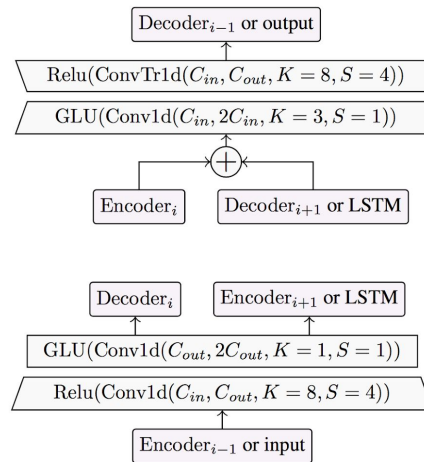


Wave-u-net extensions: Wave-U-net vs. Demucs

Block	Operation	Shape
	Input	(16384, 1)
DS, repeated for $i = 1, \dots, L$	$\text{Conv1D}(F_c \cdot i, f_d)$	
	Decimate	(4, 288)
	$\text{Conv1D}(F_c \cdot (L + 1), f_d)$	(4, 312)
US, repeated for $i = L, \dots, 1$	Upsample	
	Concat(DS block i)	
	$\text{Conv1D}(F_c \cdot i, f_u)$	(16834, 24)
	Concat(Input)	(16834, 25)
	$\text{Conv1D}(K, 1)$	(16834, 2)

Wave-U-net: building blocks

ENCODER



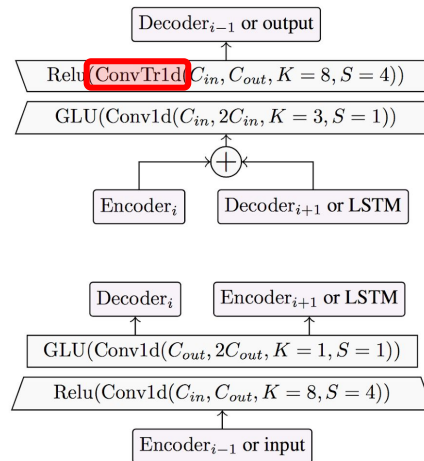
Demucs: building blocks

Wave-u-net extensions: Wave-U-net vs. Demucs

Block	Operation	Shape
	Input	(16384, 1)
DS, repeated for $i = 1, \dots, L$	$\text{Conv1D}(F_c \cdot i, f_d)$	
	Decimate	(4, 288)
	$\text{Conv1D}(F_c \cdot (L + 1), f_d)$	(4, 312)
US, repeated for $i = L, \dots, 1$	Upsample	
	$\text{Concat}(\text{DS block } i)$	
	$\text{Conv1D}(F_c \cdot i, f_u)$	(16834, 24)
	$\text{Concat}(\text{Input})$	(16834, 25)
	$\text{Conv1D}(K, 1)$	(16834, 2)

Wave-U-net: building blocks

DECODER

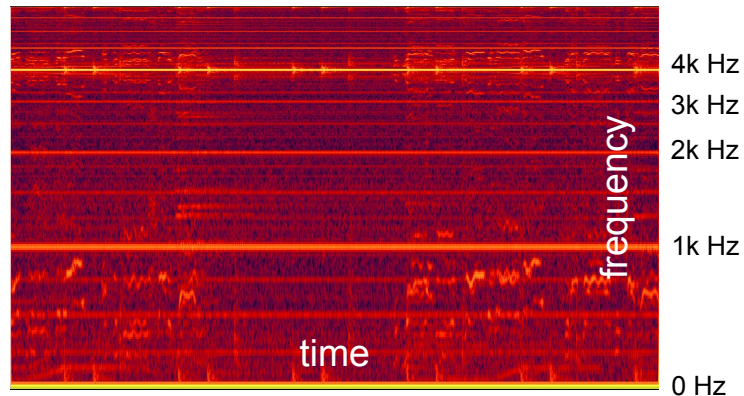


Demucs: building blocks

Deconvolutions and high-frequency artifacts



Checkerboard artifacts in images



High-frequency buzz in audio

Comparison: a perceptual study

Lluis, et al., 2019. "End-to-end music source separation: is it possible in the waveform domain?" in Interspeech.



Mixture



Spectrogram-based
(DeepConvSep)



Waveform-based
(Wavenet)



Waveform-based
(Wave-U-net)

Evaluation metrics: SDR, SIR, SAR

$$\text{SDR} := 10 \log_{10} \frac{\|s_{\text{target}}\|^2}{\|e_{\text{interf}} + e_{\text{noise}} + e_{\text{artif}}\|^2}$$

“overall performance”

$$\text{SIR} := 10 \log_{10} \frac{\|s_{\text{target}}\|^2}{\|e_{\text{interf}}\|^2}$$

“interference from other sources”

$$\text{SAR} := 10 \log_{10} \frac{\|s_{\text{target}} + e_{\text{interf}} + e_{\text{noise}}\|^2}{\|e_{\text{artif}}\|^2}$$

“algorithmic artifacts”

http://craffel.github.io/mir_eval/

<https://github.com/sigsep/sigsep-mus-eval/>

Comparison: a perceptual study

Lluis, et al., 2019. "End-to-end music source separation: is it possible in the waveform domain?" in Interspeech.



Mixture



Spectrogram-based
(DeepConvSep)



Waveform-based
(Wavenet)

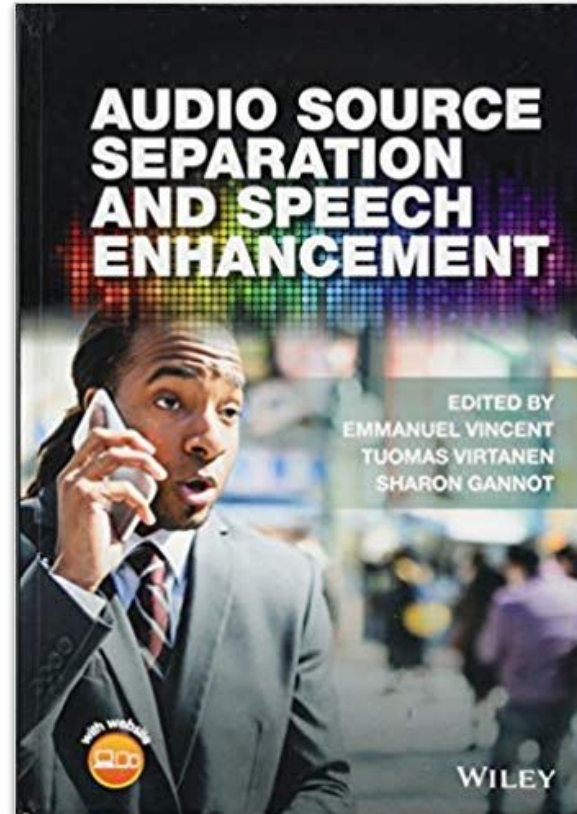
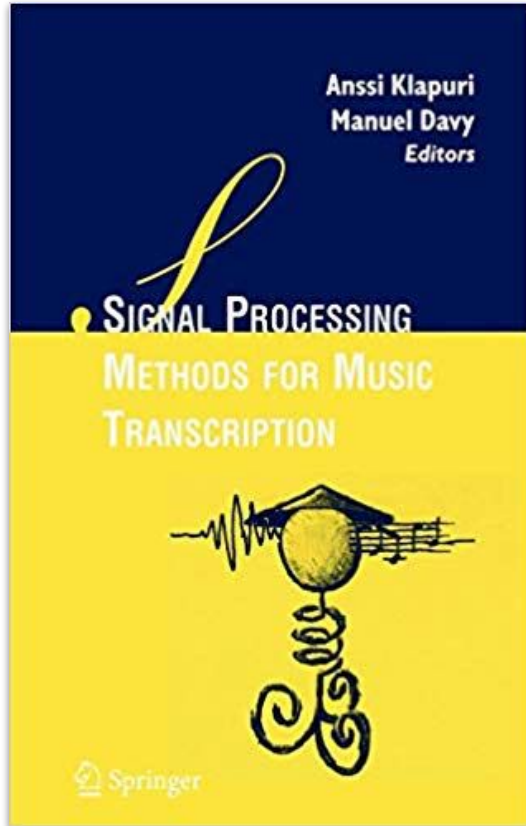


Waveform-based
(Wave-U-net)

MOS	<i>Wavenet-based</i>	<i>Wave-U-Net</i>
<i>Vocals</i>	3.0 ± 1.0	3.3 ± 0.85

(t-test: p-value=0.049, 15 participants)

Additional references



Einarhea Cano, Derry FitzGerald, Anissa Iivikos,
Mark D. Plumbley, and Fabian-Ruben Sotelo

Musical Source Separation

An introduction



Many people listen to recorded music as part of their everyday lives, e.g., from radio or TV programs, compact discs, downloads, or, increasingly, online streaming services. Sometimes we might want to remix the balance within the music, perhaps to make the vocals louder or to suppress an unwanted sound, or we might want to upmix a two-channel stereo recording to a 5.1-channel surround sound system. We might also want to change the spatial location of a musical instrument within the mix. All of these applications are relatively straightforward, provided we have access to separate sound channels (stems) for each musical audio object.

However, if we only have access to the final recording mix, which is usually the case, this is much more challenging. To estimate the original musical sources, which would allow us to remix, suppress, or upmix the sources, we need to perform musical source separation (MSS).

In the general source separation problem, we are given one or more mixture signals that contain different combinations of some original source signals. This is illustrated in Figure 1, where four sources, i.e., vocals, drums, bass, and guitar, are all present in the mixture. The task is to recover one or more of the source signals given the mixtures. In some cases, this is relatively straightforward, e.g., if there are at least as many mixtures as there are sources and if the mixing process is fixed, with no delays, filters, or nonlinear mastering [1].

However, MSS is normally more challenging. Typically, there may be many musical instruments and voices in a two-channel recording, and the sources have often been processed with the addition of filters and reverb/ambience (sometimes nonlinear) in the recording and mixing process. In some cases, the sources may move or the production parameters may change, meaning that the mixture is time varying.

Nevertheless, musical sound sources have particular properties and structures that can help us. For example, musical source signals often have a regular harmonic structure of frequencies at regular intervals and can have frequency contours characteristic of each musical instrument. They may also repeat, in particular, temporal patterns based on the musical structure.

Music Theory Abstracts 16 (2009) 2018-18792
Music Technology Group, Universitat Pompeu Fabra

End-to-end music source separation: is it possible in the waveform domain?

Francesc Llull¹ Jordi Pons¹ Xavier Serra

Music Technology Group, Universitat Pompeu Fabra, Barcelona.

name.surname@upf.edu

Abstract

Most of the currently successful source separation techniques use the magnitude spectrogram as input, and are therefore by default omitting part of the signal: the phase. To avoid omitting potentially useful information, we study the viability of using end-to-end models for music source separation — which take into account all the information available in the raw audio signal, including the phase. Although during the last decades end-to-end music source separation has been considered almost unattainable, our results confirm that waveform-based models can perform similarly (if not better) than spectrogram-based deep learning models. Namely, a WaveNet-based model we propose and Wave-U-Net can outperform DeepConvSep, a recent spectrogram-based deep learning model.

Index Terms: source separation, end-to-end learning.

1. Introduction

When two or more sounds co-exist, they interfere with each other resulting in a novel mixture signal where sounds are superposed (and, sometimes, masked). The source separation task tackles the inverse problem of recovering each individual sound source contribution from an observed mixture signal.

With the recent advances in deep learning, source separation techniques have improved substantially [1]. Interestingly, though, nearly all successful deep learning algorithms use the magnitude spectrogram as input [1, 2, 3] — and are therefore, by default, omitting part of the signal: the phase. Omitting the potentially useful information of the phase entails the risk of finding a sub-optimal solution. In this work, we aim to take full advantage of the acoustic modeling capabilities of deep learning to investigate whether it is possible to approach the problem of music source separation directly in an end-to-end learning fashion. Consequently, our investigation is centered on studying how to separate music sources (e.g., singing voice, bass or drums) directly from the raw waveform music mixture.

During the last two decades, matrix decomposition methods have dominated the field of audio source separation. Several algorithms have been proposed throughout the years, with independent component analysis (ICA) [4], sparse coding [5], or non-negative matrix factorization (NMF) [6] being the most used ones. Given that magnitude or power spectrogram representations are always non-negative, imposing a non-negative constraint (like in NMF) is particularly useful when analyzing these spectrograms — but less appropriate for processing waveforms, which range from -1 to 1 . For that reason, methods like ICA and sparse coding have historically been used to process waveforms [7, 8, 9]. Waveform representations preserve all the information available in the raw signal. However, given the unpredictable behavior of the phase in real-life

sounds, it is rare to find identical waveforms produced by the same sound source. As a result of this variability, a single basis' cannot represent a sound source and therefore, one requires *i)* a large amount of bases, or *ii)* shift-invariant bases to obtain accurate decompositions [8, 10]. Although several matrix decomposition methods have been used for decomposing waveform-based mixtures [7, 8, 9], these have never worked as well as the spectrogram-based ones.

Due to the above mentioned difficulties, the phase of complex time-frequency representations is commonly discarded, assuming that magnitude spectrograms already carry meaningful information about the sound sources to be separated. Phase-related problems disappear when sounds are just represented as magnitude or power spectrograms, since different realizations of the same sound are almost identical in this time-frequency plane. This allows to easily overcome the variability problem found when operating with waveforms.

Most matrix decomposition methods rely on a signal model assuming that sources add linearly in the time domain [10].

However, the addition of signals in the time and frequency domains is not equivalent if phases are discarded. Only in expectation: $E\{|X(k)|^2\} = |E\{X(k)\}|^2 + |E\{Y(k)\}|^2$, where $X(k) = DF^T(x(t))$. This means that we can approximate the time-domain summation in the power spectral domain. For that reason, many approaches utilize power spectrograms as inputs. Although magnitude spectrograms work well in practice [11], there is no similar theoretical justification for such an inconsistency with the signal model when the phases are discarded.

Finally, note that these methods operating on top of spectrograms still need to deliver a waveform signal. To this end, the main practice is to filter the original magnitude or power spectrogram with (predicted) time-frequency masks. Accordingly, the original noisy phase of the mixture is used when synthesizing the waveform of the estimated sources — which might introduce an additional source of error [10]. Notably, many modern spectrogram-based deep learning models are also relying on this same (potentially problematic) approach [2, 12]. To overcome this issue, some tried to consider the phase when separating the sources [13, 14, 15],¹ or some others relied on a sinusoidal signal model at synthesis time [16]. However, in our work, we do not want to rely on any time-frequency transform or any signal model. Instead, we aim to directly approach the problem in the waveform domain.

As seen, many issues still exist around the idea of discarding the phase: are we missing crucial information when performing it? When using the phase of the mixture at synthesis time, are we introducing artifacts that are limiting our model's performance? Or, since magnitude spectrograms (differently from

¹ICA, sparse coding or NMF model the mixture signal as a weighted sum of bases, which represents a source or components of a source. Using the full complex STFT number, instead of utilizing phases representations (either as the input or when applying the masks).

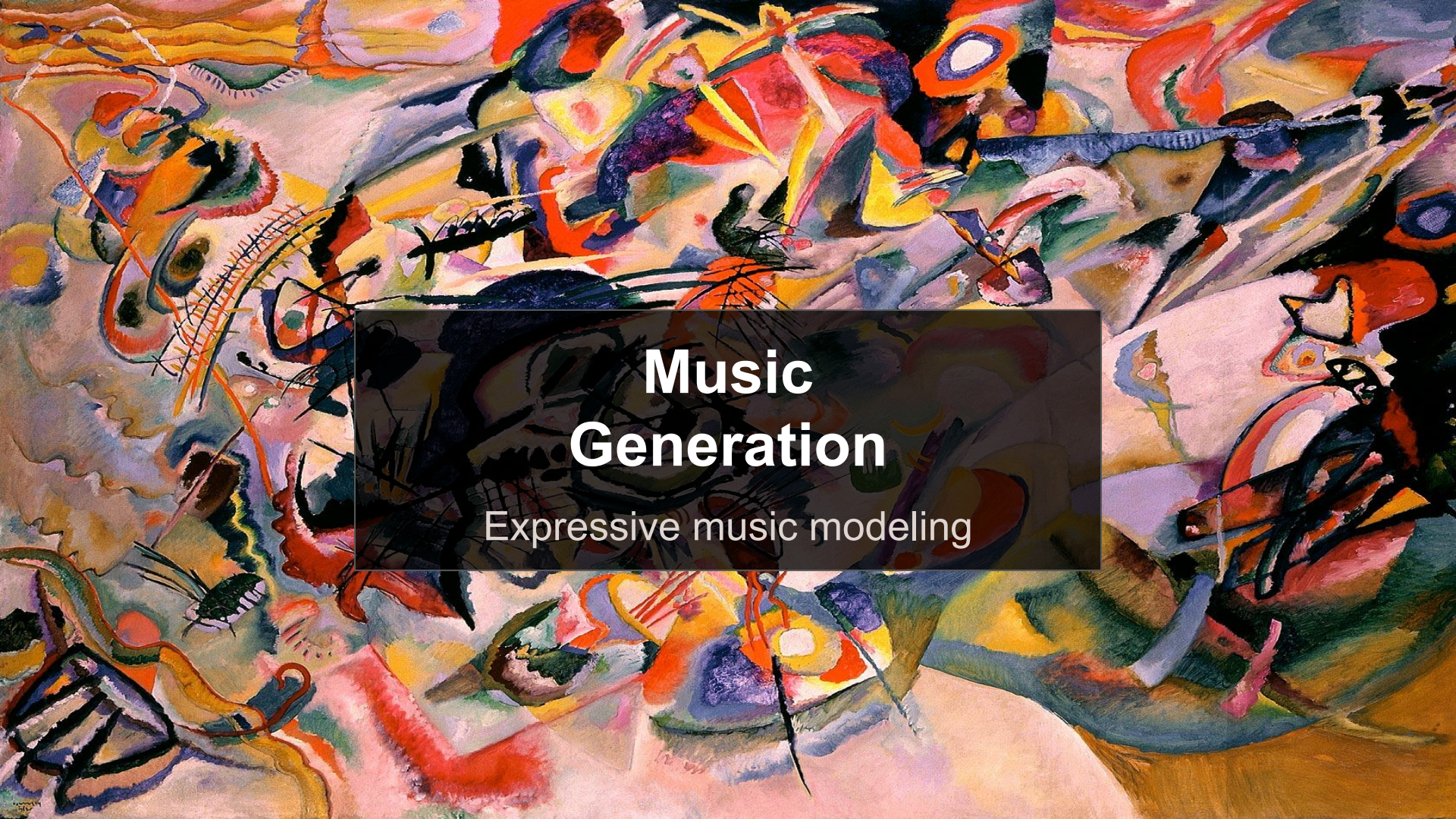
²Continued equally.

arXiv:1810.12187v2 [cs.LG] 28 Jun 2019

S. Dubnov, 2002. "Extracting sound objects by independent subspace analysis" in AES Conference.

Blumensath and Davies, 2004. "Unsupervised learning of sparse and shift-invariant decompositions of polyphonic music," in ICASSP.

Jang and Lee, 2003. "A maximum likelihood approach to single-channel source separation" in Journal of Machine Learning Research.



Music Generation

Expressive music modeling

Overview

Why audio? Why raw audio?

Generative models

Likelihood-based models of raw audio

Adversarial models of raw audio

Summary

Why audio? Why raw audio?

Why audio?

Music generation is typically studied in the symbolic domain

Lento, ma non troppo. (♩ = 100)

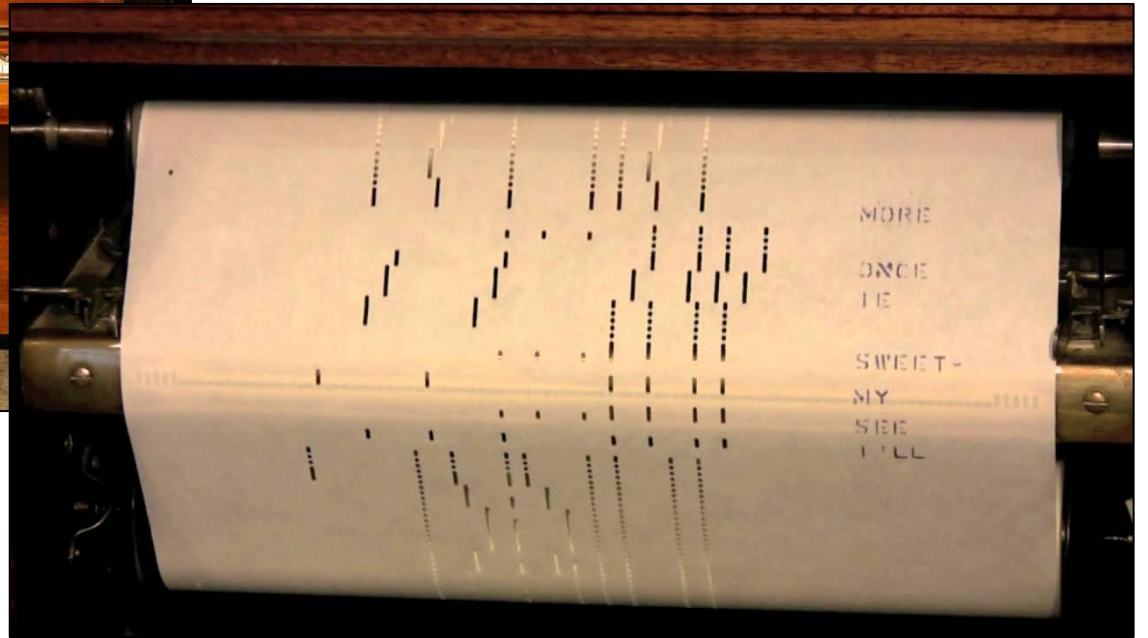
legato *p*

3

ten. *a tempo*

cresc. *stretto* *riten.* *p*

The image displays a musical score for piano, consisting of two systems of music. The first system is marked *Lento, ma non troppo.* (♩ = 100) and *legato*. It features a treble clef with a key signature of two sharps (F# and C#) and a 4/4 time signature. The music is written for the right hand, with a large '3' indicating a triplet. The left hand part is written in the bass clef. The score includes various performance markings such as *p* (piano), *ten.* (tension), and *a tempo*. The second system continues the piece, marked with *cresc.* (crescendo), *stretto*, *riten.* (ritardando), and *p*. The score is heavily annotated with fingerings (numbers 1-5) and slurs, indicating complex technical requirements for the performer.



SONAR X2 1/4

Smart Select Move Edit Draw Erase

Snap TO RY Marks

00:00:05:00

44.1 24 120.00 4/4

Loop 1:01:000 9:01:000

M S PDC FX DIM R!

Mackie Control - 1 Trks 1-16, Bus 1, J...

Main Screenset 1 2 3 4 5 6 7 8 9 10

View Notes Controllers Tracks

Clip Trk PC

C4 G3 C3 C2 C1

4 5 6 7

Why audio?

Many instruments have complex action spaces

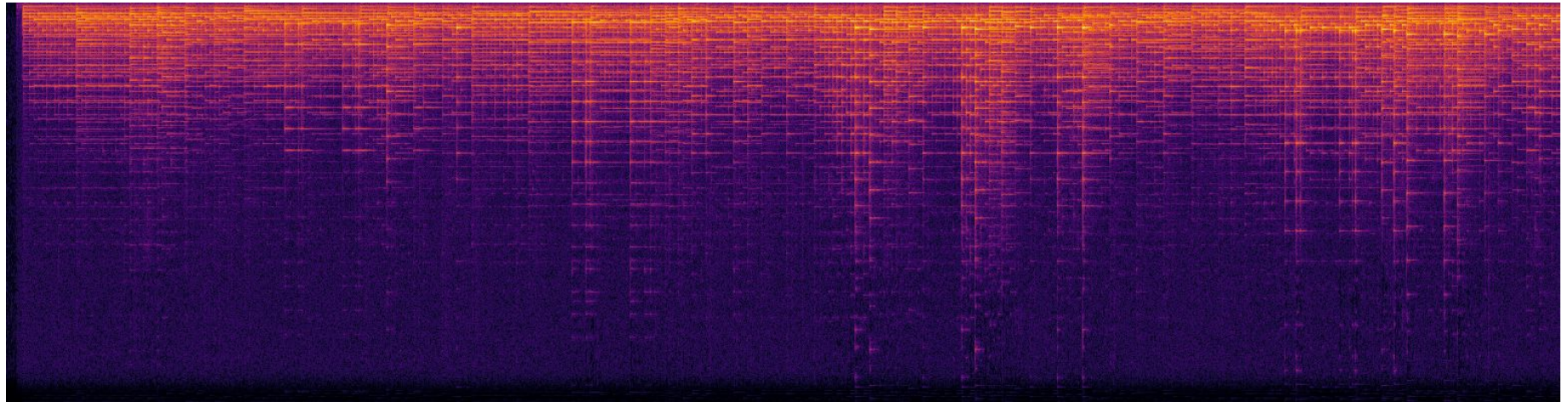
Rich palette of sounds and timbral variations

Guitar

- pick vs. finger
- picking position
- frets
- harmonics
- ...

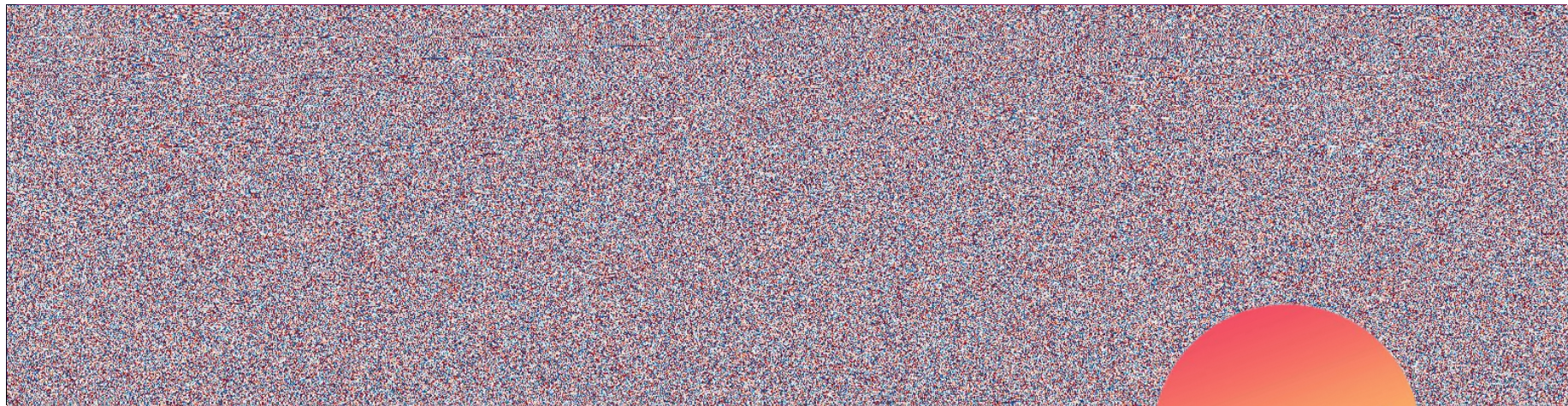


Why raw audio?



Magnitude spectrogram

Why raw audio?



Phase spectrogram



Why raw audio?

Phase is often unimportant in discriminative settings,
but is very important perceptually!



original phase

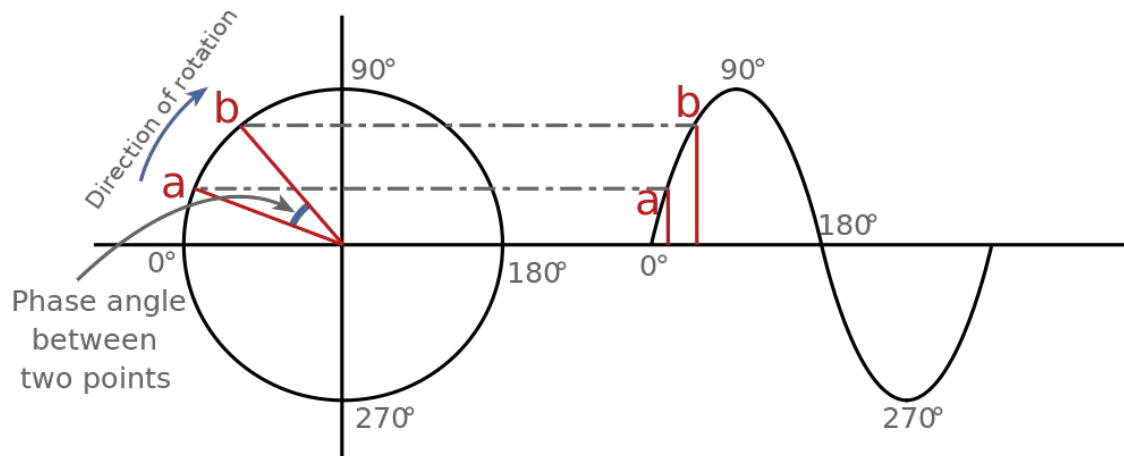
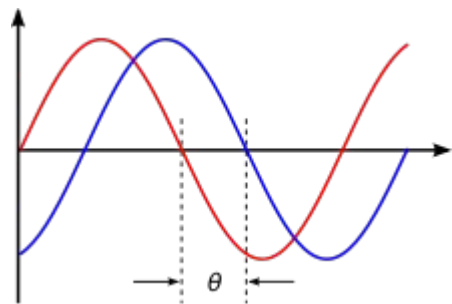


random phase

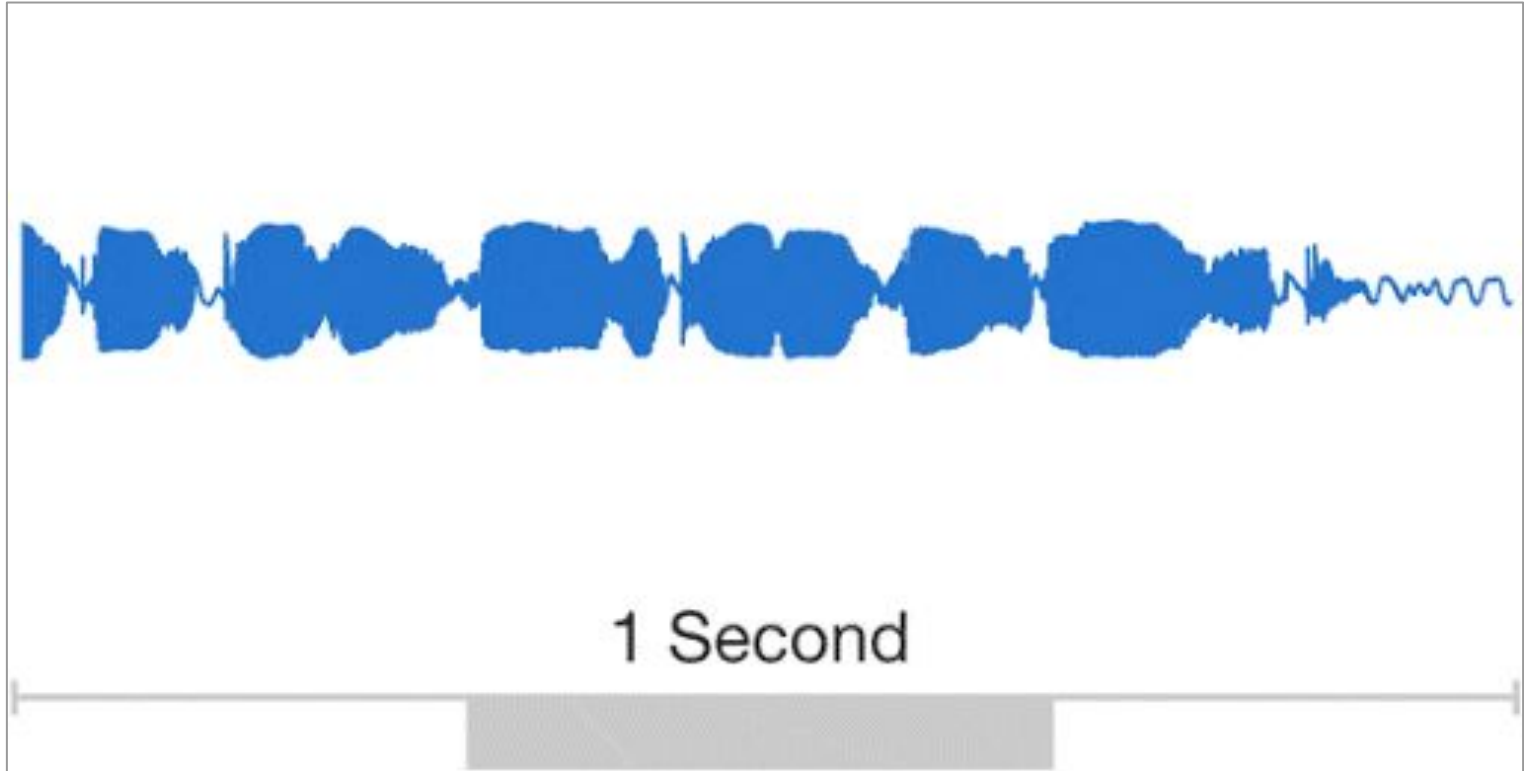
Why raw audio?

Phase is hard to model:

- it is an angle, so it wraps around
- it becomes random as the magnitude tends to 0
- absolute phase is less meaningful, but relative phase differences matter



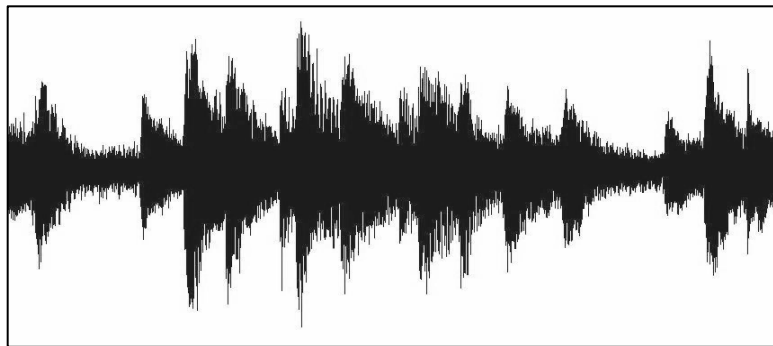
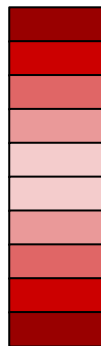
What is “raw audio” anyway?



Discretising audio

- Time
- Amplitude

uniform
quantisation



μ -law
quantisation

Generative models

Generative models

Given a dataset of examples \mathbf{X} drawn from $\mathbf{p}(\mathbf{X})$:
a generative model estimates $\mathbf{p}(\mathbf{X})$

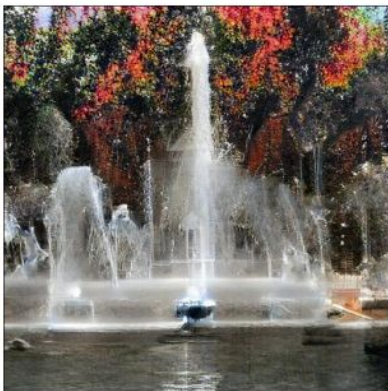
Generative models

Given a dataset of examples \mathbf{X} drawn from $\mathbf{p}(\mathbf{X})$:
a generative model estimates $\mathbf{p}(\mathbf{X})$

Explicit: given $\mathbf{x} \in \mathbf{X}$, model can infer $\mathbf{p}(\mathbf{x})$

Implicit: model can produce new samples $\mathbf{x} \sim \mathbf{p}(\mathbf{X})$

Generative models



Likelihood-based models

Likelihood-based models parameterise $\mathbf{p}(\mathbf{X})$ directly

Objective function: maximise $\sum_{\mathbf{x} \in \mathcal{X}} \log p(\mathbf{x})$

Autoregressive models

Autoregressive models factorise $\mathbf{p}(\mathbf{X})$
into simpler (scalar) distributions

$$\mathbf{x} = (\mathbf{x}_1, \mathbf{x}_2, \mathbf{x}_3, \dots, \mathbf{x}_n)$$

$$\mathbf{p}(\mathbf{x}) = \prod_i \mathbf{p}(\mathbf{x}_i | \mathbf{x}_{<i}) \quad \text{chain rule of probability}$$

We can use the same model $\mathbf{p}(\mathbf{x}_i | \mathbf{x}_{<i})$ for all i !

Flow-based models

Flow-based models transform $\mathbf{p}(\mathbf{X})$ to a simple (factorised) distribution with an invertible mapping

$\mathbf{p}(\mathbf{x}) = \mathbf{p}(\mathbf{z}) \cdot |\det \mathbf{J}|^{-1}$ change of variables theorem

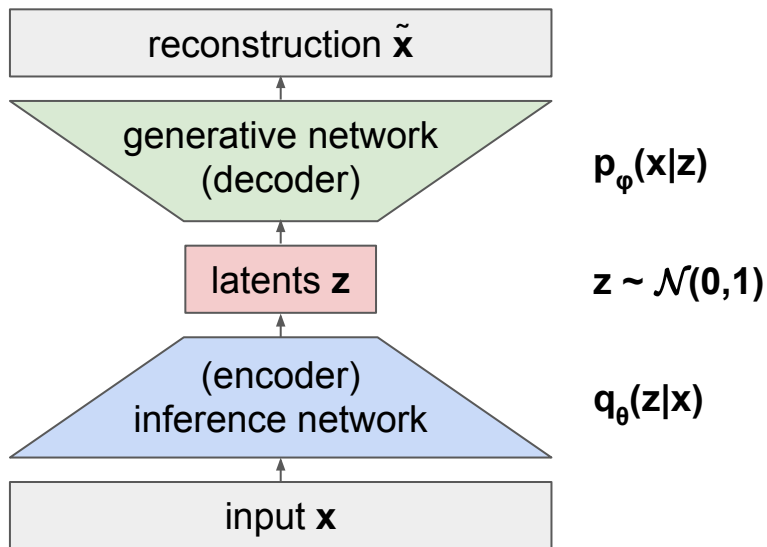
$\mathbf{J} = d(\mathbf{g}(\mathbf{z}))/d\mathbf{z}$ $\mathbf{x} = \mathbf{g}(\mathbf{z})$

Important constraints:

$\mathbf{g}(\mathbf{z})$ must be invertible

$\det \mathbf{J}$ must be tractable

Variational autoencoders (VAEs)



VAE maps latents \mathbf{z} from a simple distribution to \mathbf{x} with a (non-invertible) generative network

The inference network approximates the inverse operation

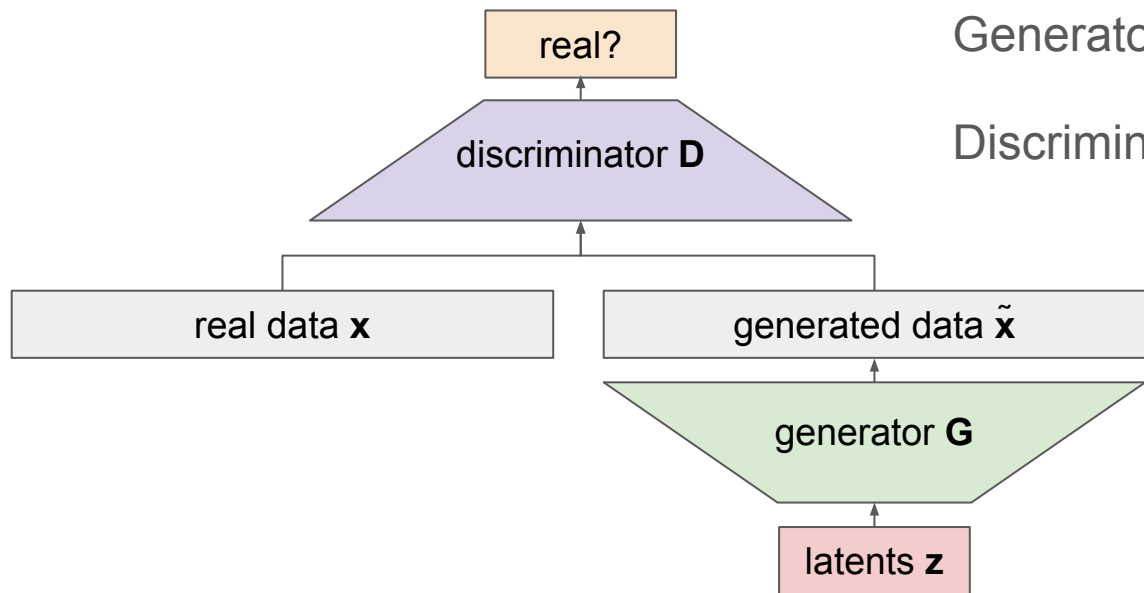
$p(\mathbf{x})$ cannot be computed exactly, the Evidence Lower Bound (ELBO) is maximised instead

Adversarial models

$$\mathcal{L} = \mathbb{E}_x[\log(D(x))] + \mathbb{E}_z[\log(1 - D(G(z)))]$$

Generator minimises \mathcal{L}

Discriminator maximises \mathcal{L}



More exotic flavours

- Implicit quantile networks
- Energy-based models
- Optimal transport (e.g. Wasserstein autoencoders)
- Score-based generative modelling
- ...

Conditional generative models

Conditioning is “side information” which allows for control over the model output

$p(\mathbf{x}|\mathbf{c})$ vs. $p(\mathbf{x})$

Conditional generative models

Conditioning is “side information” which allows for control over the model output

$p(\mathbf{x}|\mathbf{c})$ vs. $p(\mathbf{x})$

image models

sparsely conditioned

densely conditioned

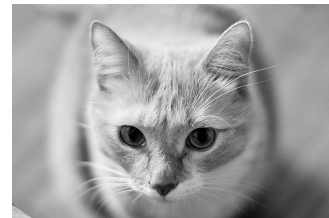
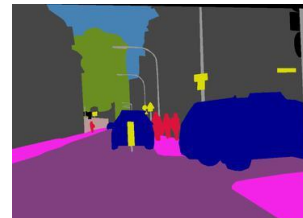
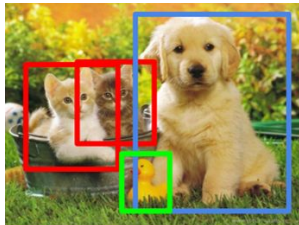
class labels

bounding boxes

segmentation

grayscale image
(colorisation)

$\mathbf{y} = \text{“cat”}$



Conditional generative models

Conditioning is “side information” which allows for control over the model output

$p(\mathbf{x}|\mathbf{c})$ vs. $p(\mathbf{x})$

sparsely conditioned

densely conditioned

music audio models

composer
instrument(s)
tempo
timbre
...

note density
musical form
...

score



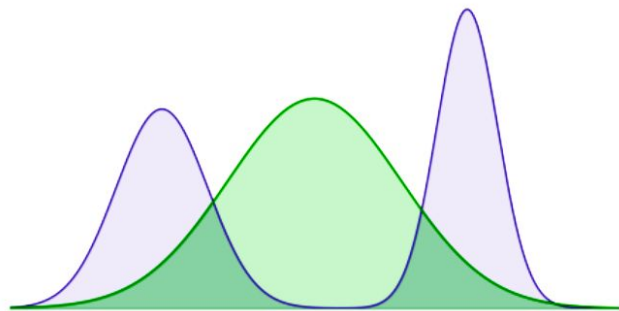
MIDI



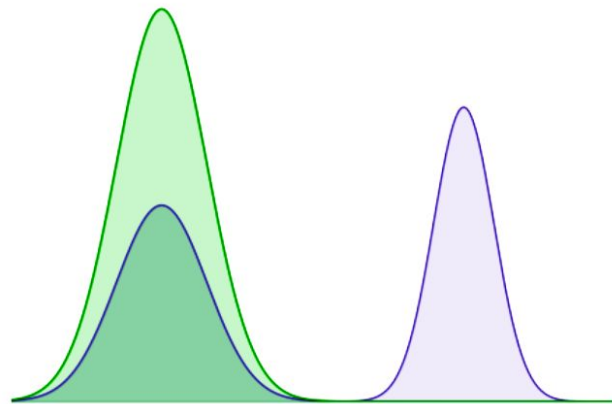
other audio signals



Mode-covering vs. mode-seeking behaviour



mode-covering

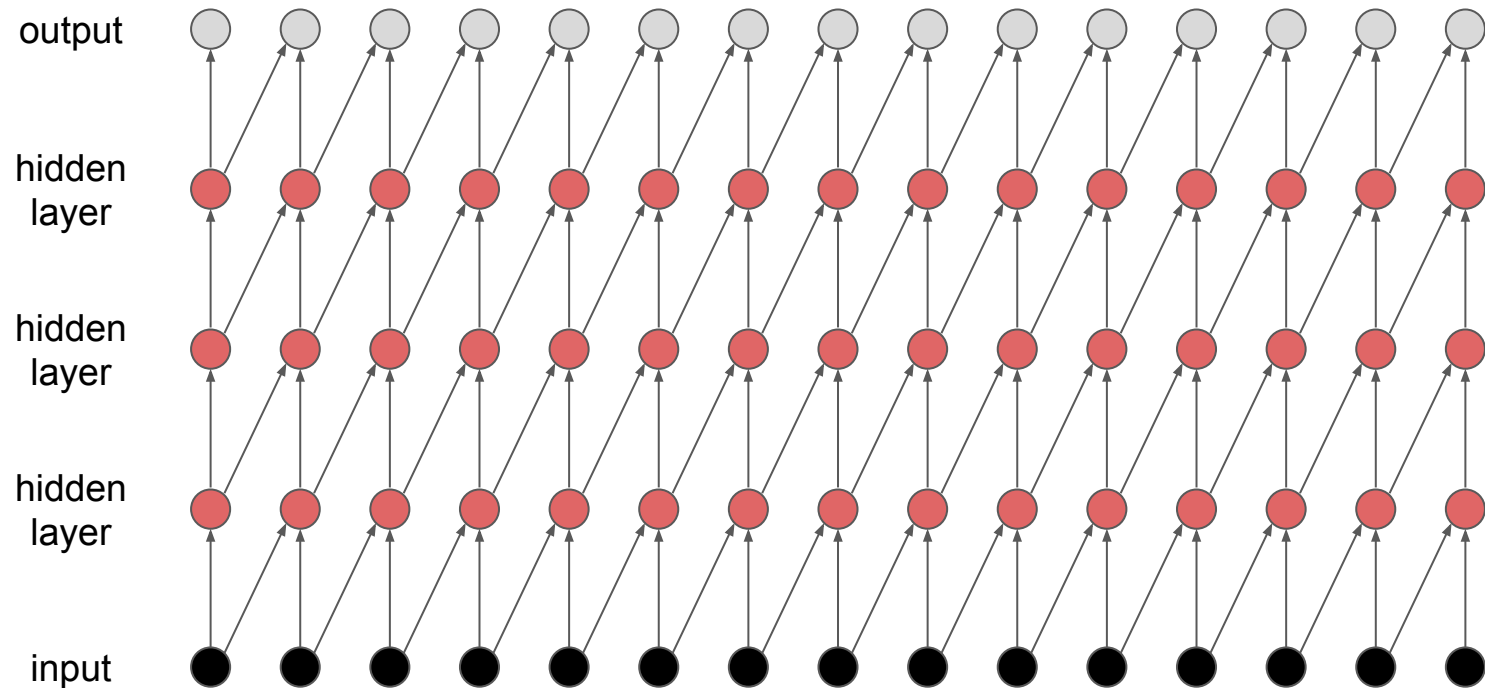


mode-seeking

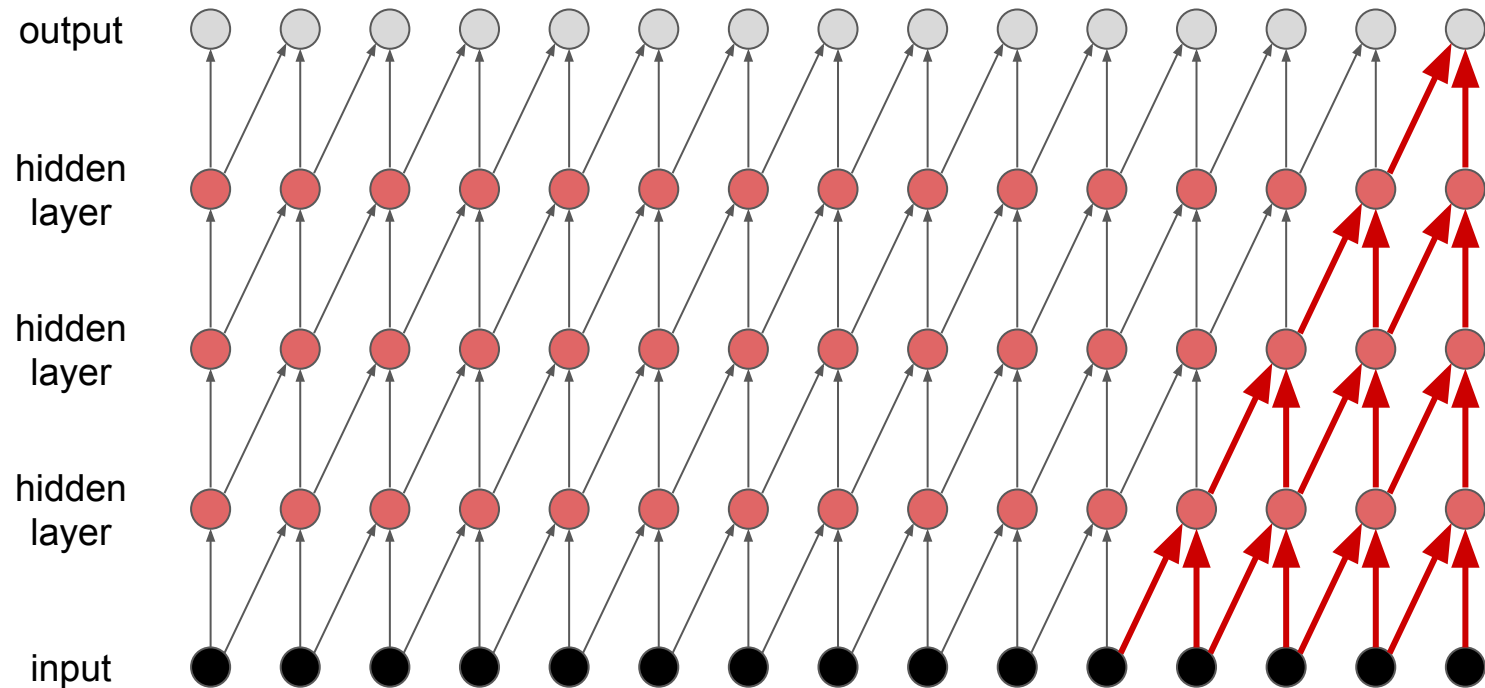
- Likelihood-based models are mode-covering
- Adversarial models are (typically) mode-seeking
- In more densely conditioned settings, we tend to care less about covering all the modes

Likelihood-based models

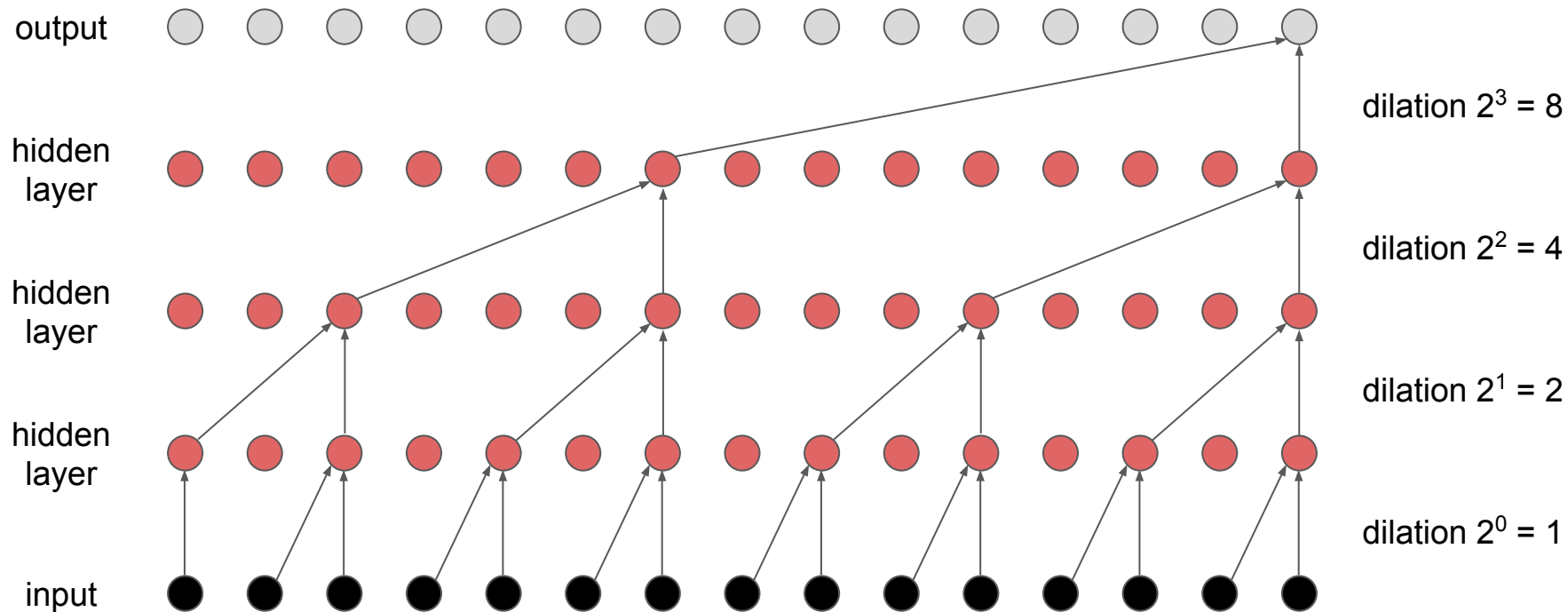
WaveNet



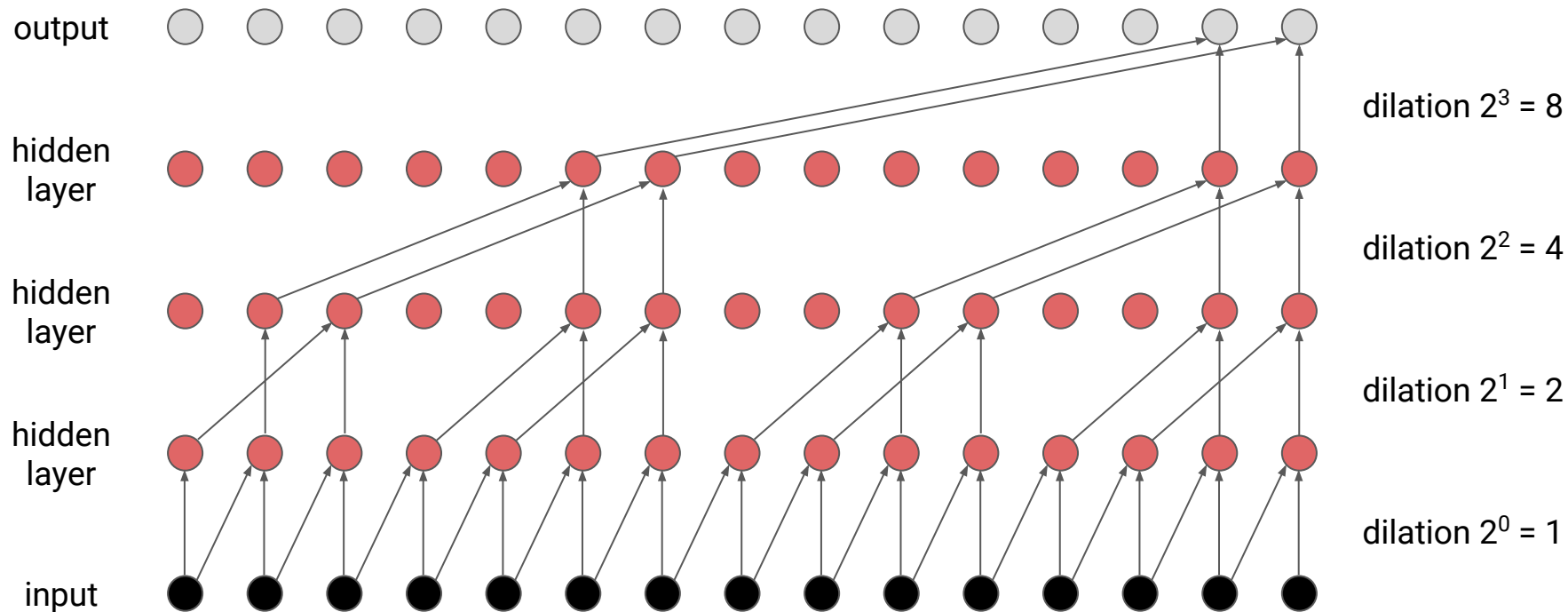
WaveNet

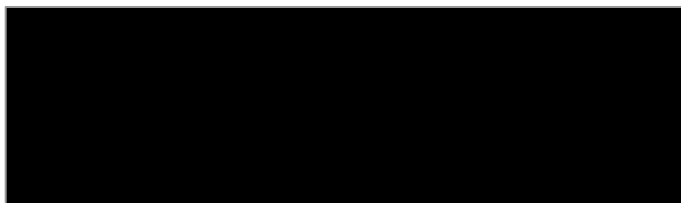
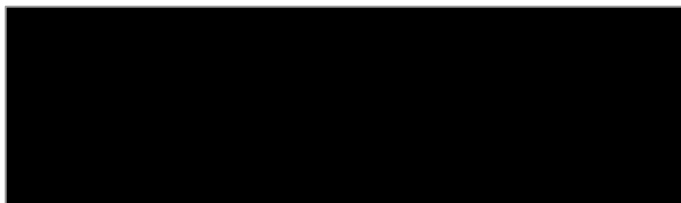


WaveNet: dilated convolutions

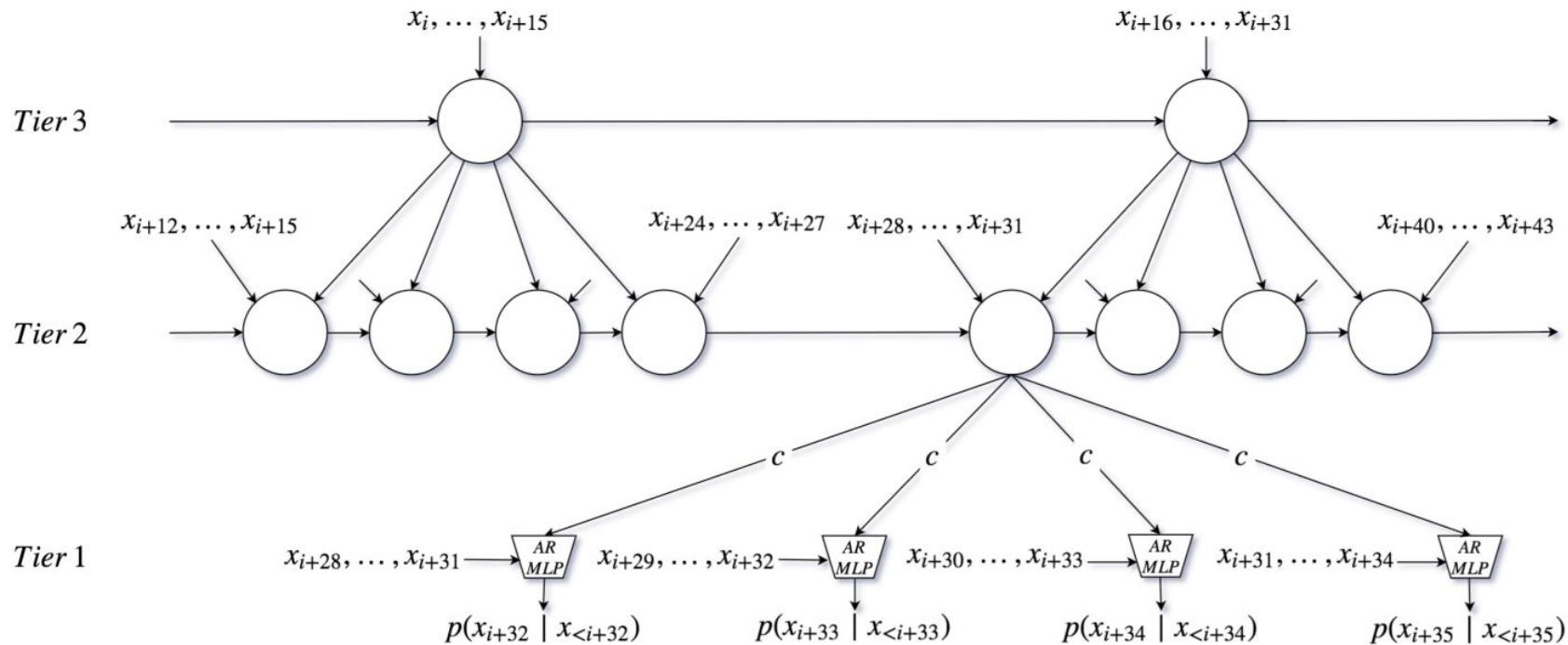


WaveNet: dilated convolutions

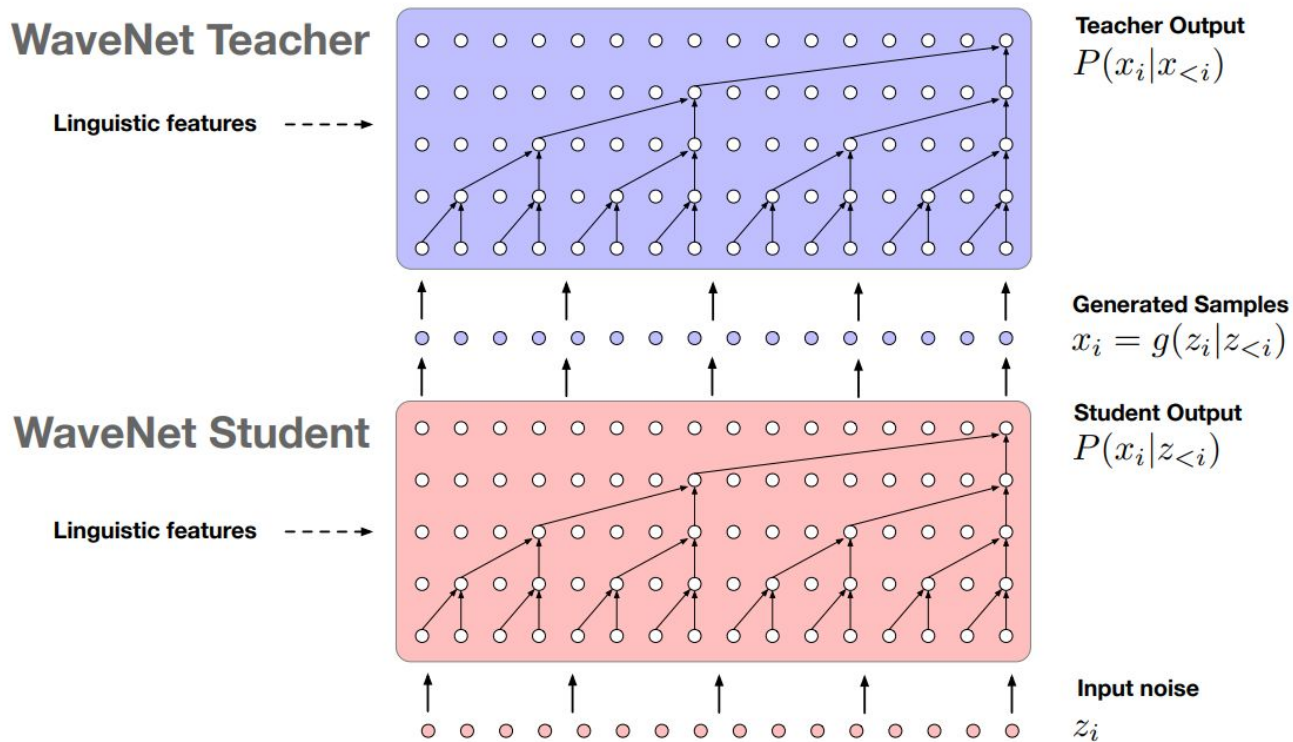




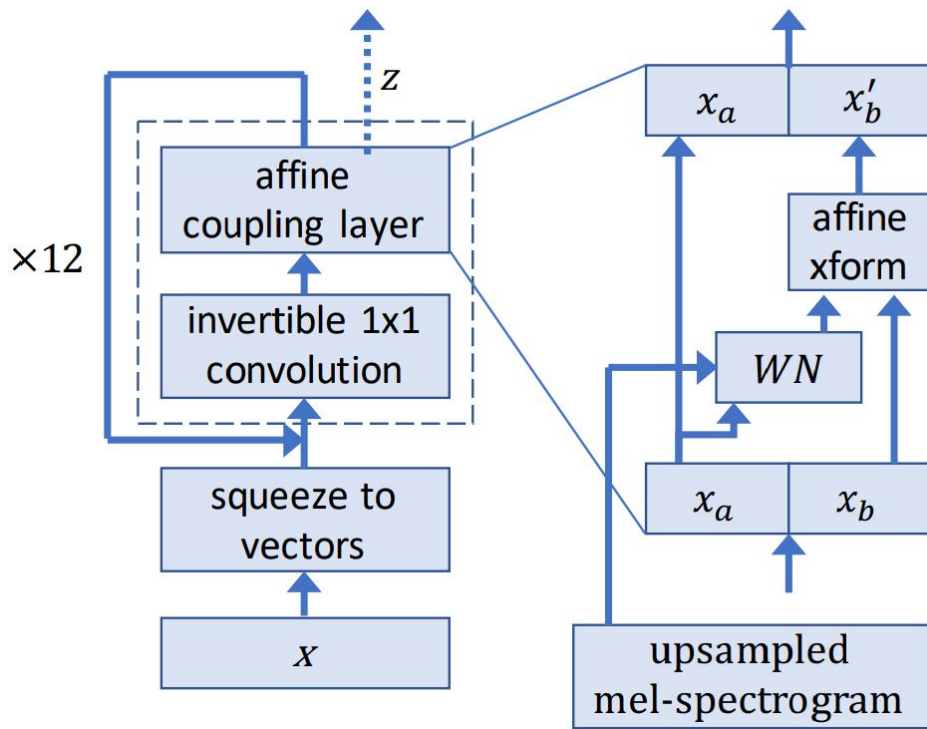
SampleRNN



Parallel WaveNet, ClariNet



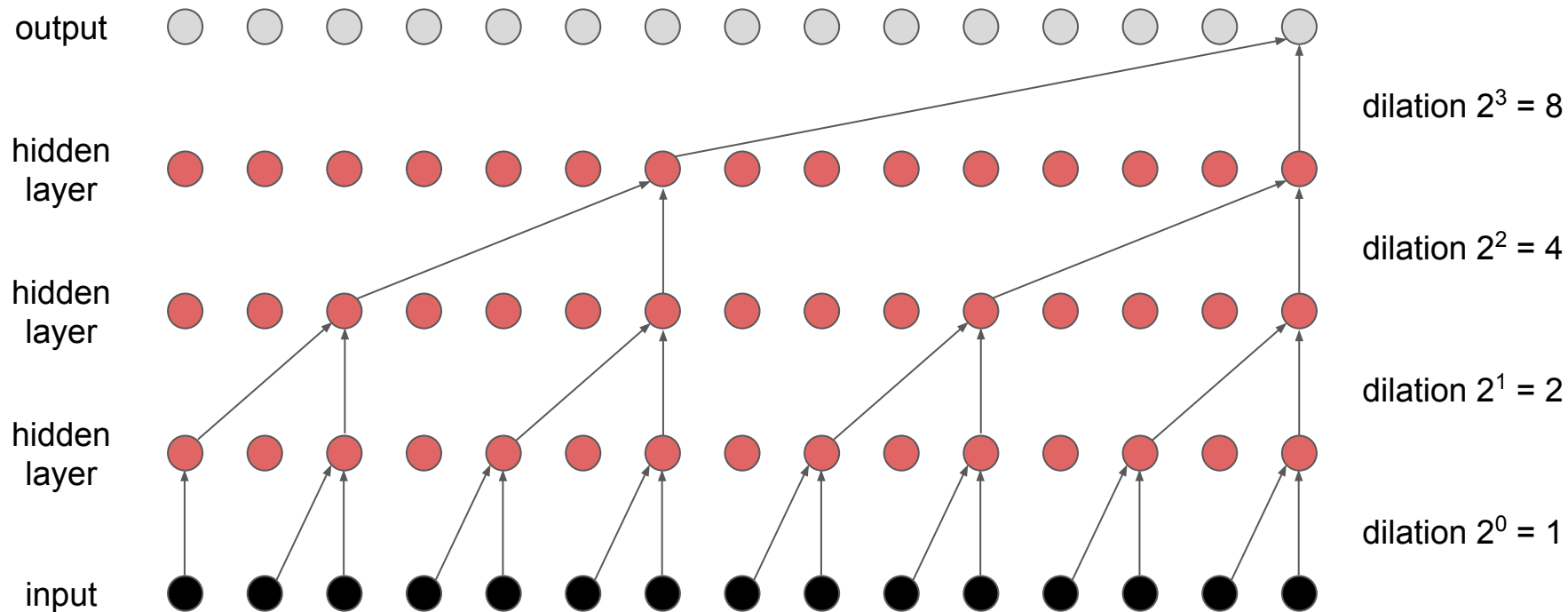
WaveGlow, FloWaveNet



Prenger et al., 2019. "Waveglow: A flow-based generative network for speech synthesis", ICASSP.

Kim et al., 2019. "FloWaveNet: A generative flow for raw audio", ICML.

WaveNet: #layers $\sim \log(\text{receptive field length})$



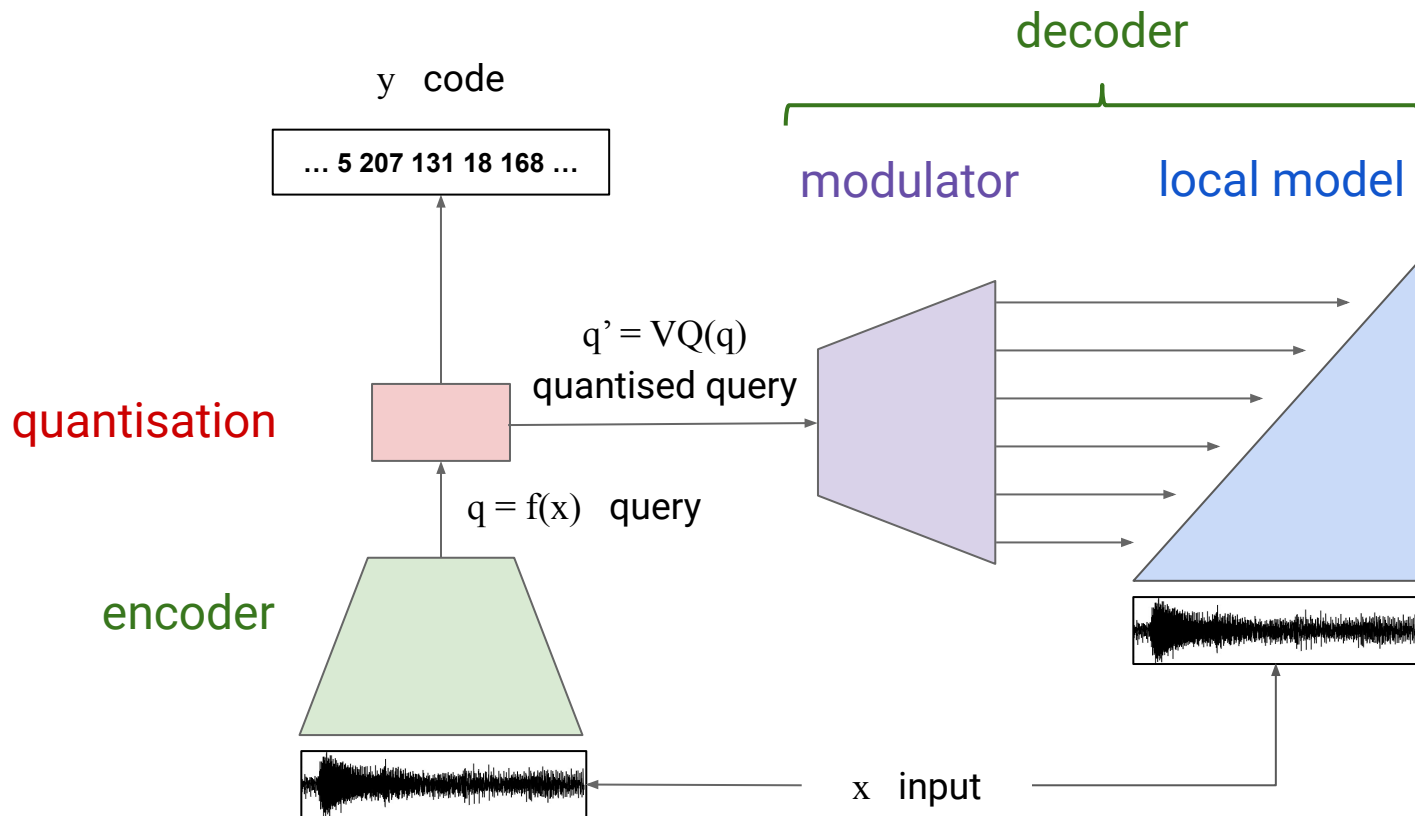
... but memory usage \sim receptive field length

Required model depth is **logarithmic** in the desired receptive field length

Required memory usage during training is still **linear** in the desired receptive field length!

\Rightarrow We cannot scale indefinitely using dilation

Autoregressive discrete autoencoders



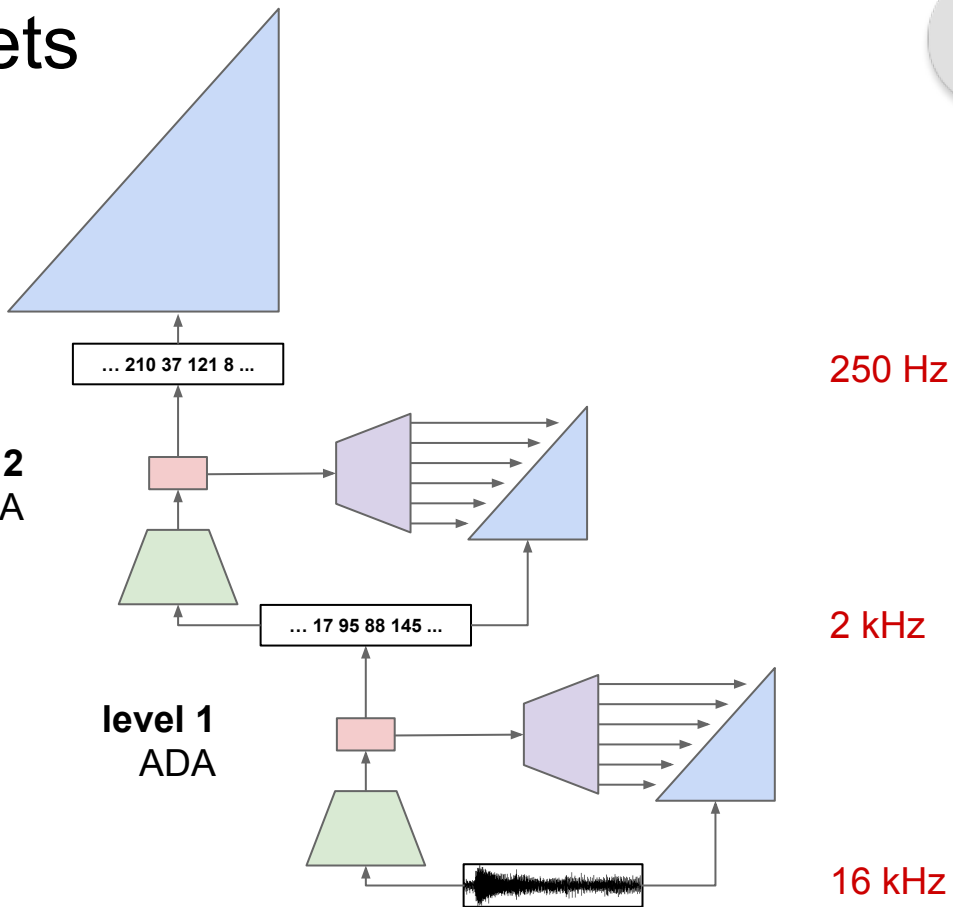
Hierarchical WaveNets



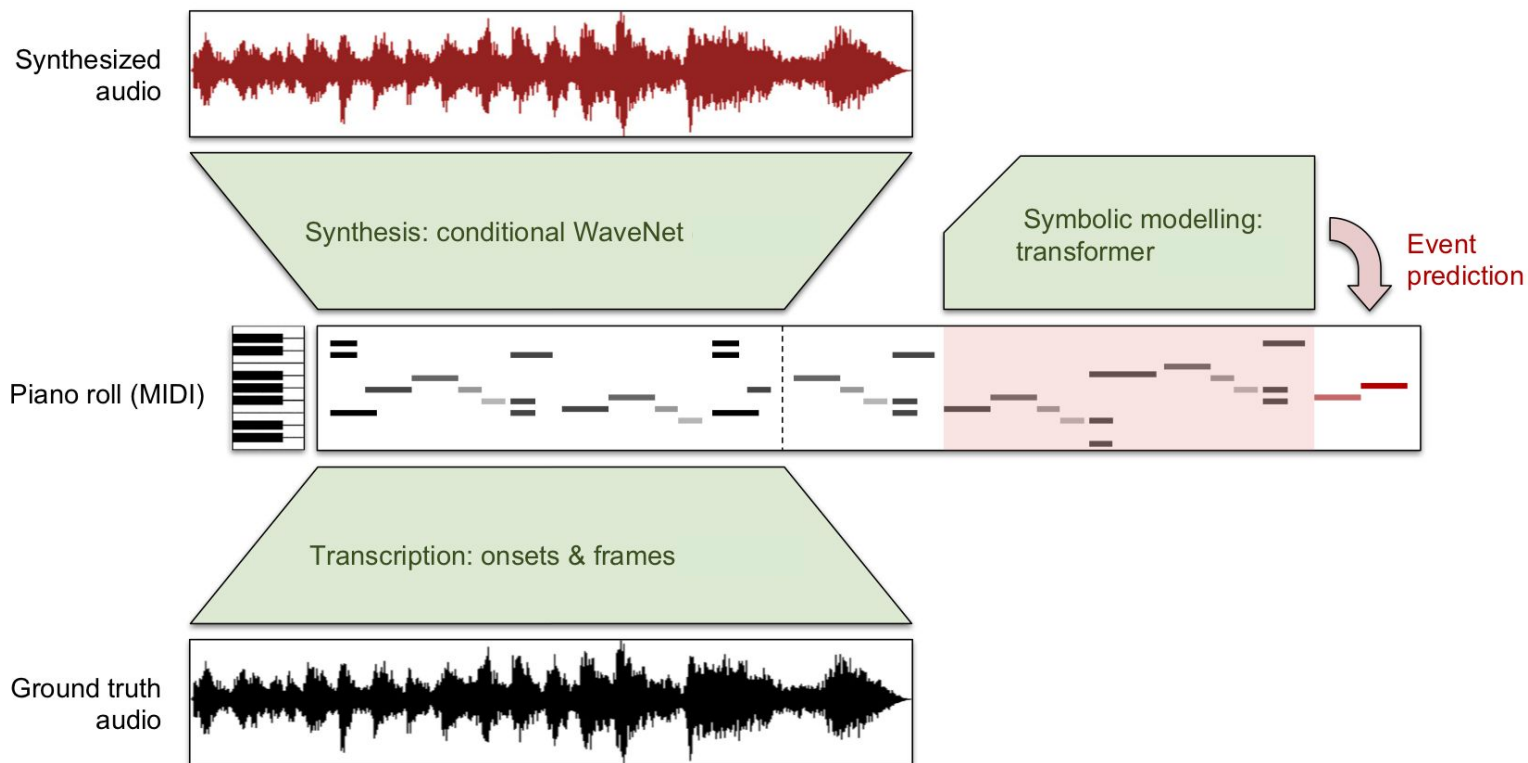
level 3
unconditional model

level 2
ADA

level 1
ADA

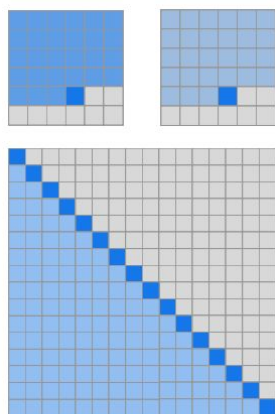


Wave2Midi2Wave and the MAESTRO dataset

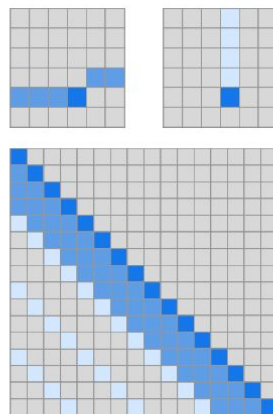


Sparse transformers

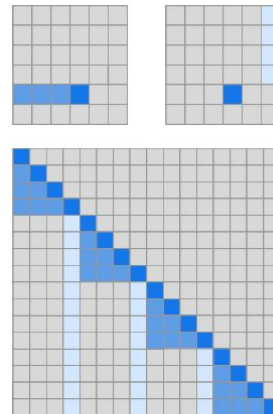
Attention (with sparse masks) instead of recurrence / convolutions.



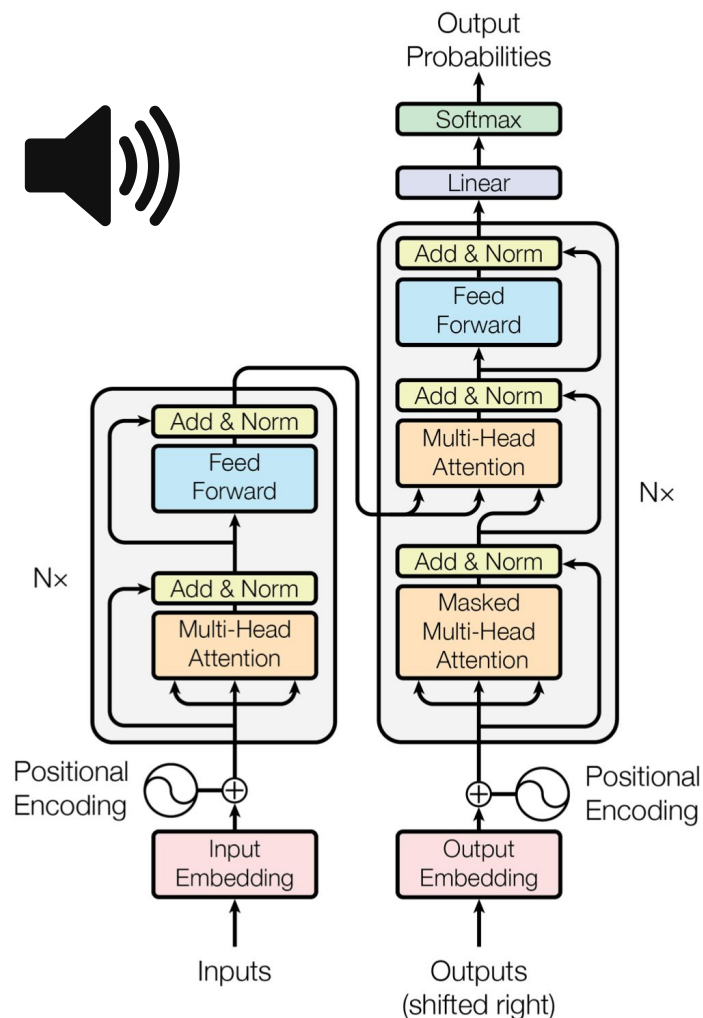
(a) Transformer



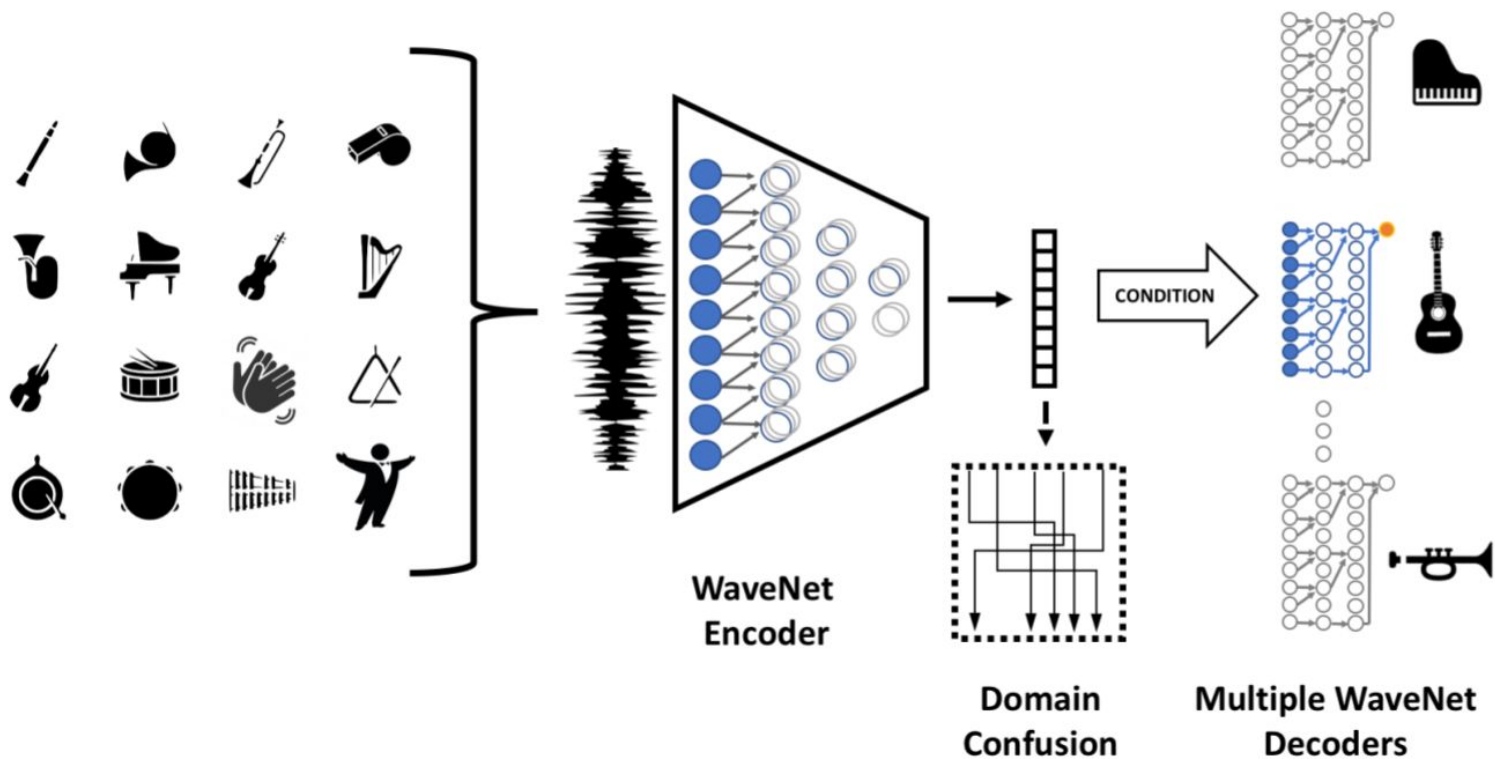
(b) Sparse Transformer (strided)



(c) Sparse Transformer (fixed)



Universal music translation network



<http://dadabots.com/>

Neural networks

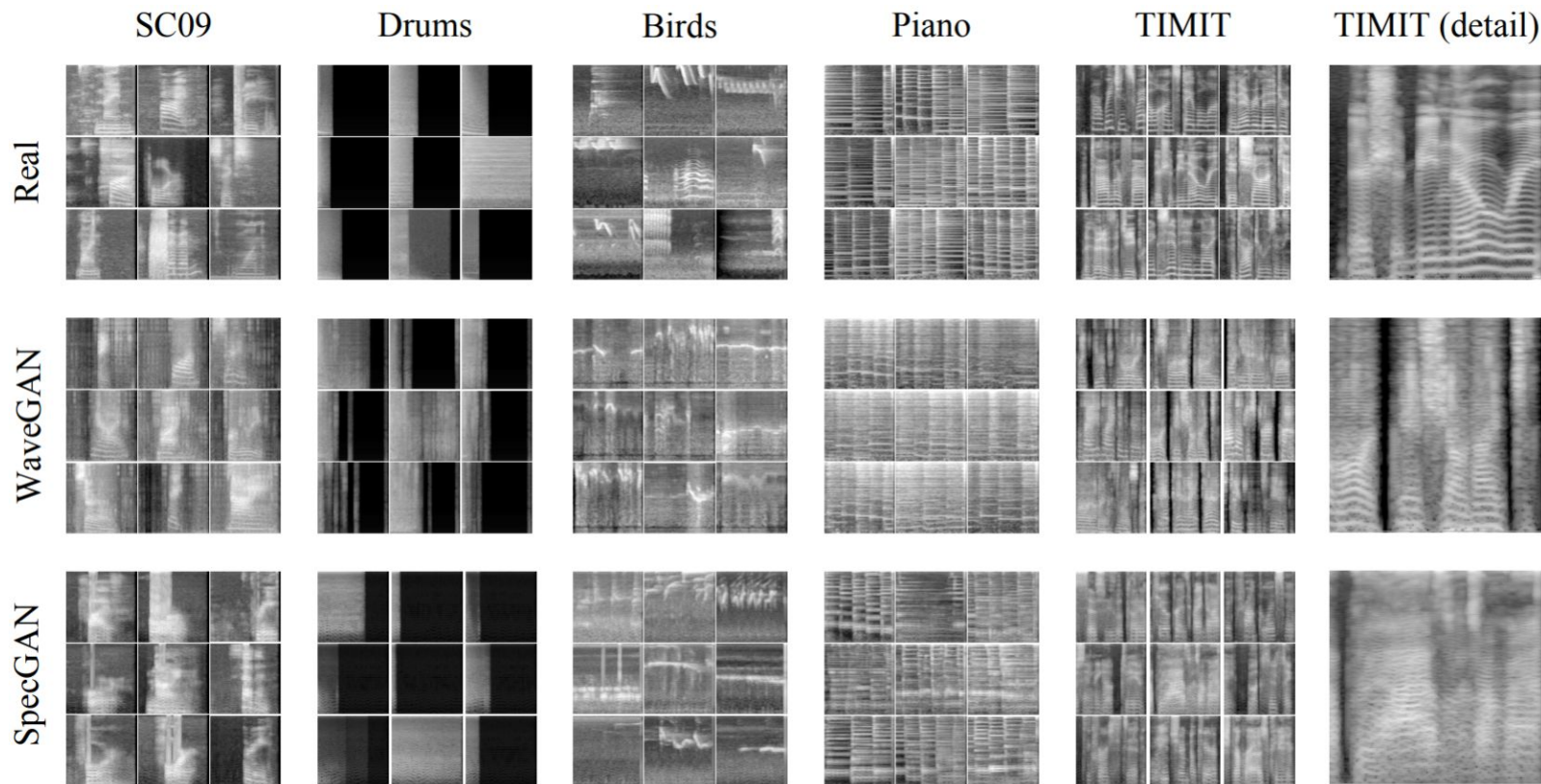
generating death metal
via livestream 24/7 to infinity

We make raw audio neural networks
that can imitate bands

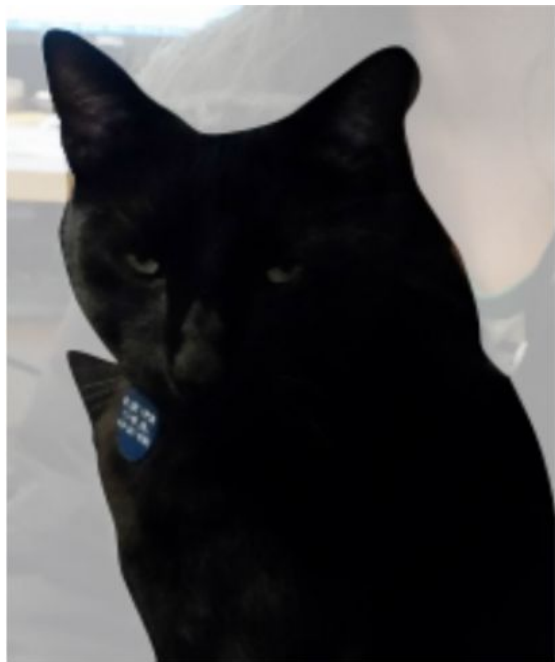


Adversarial models

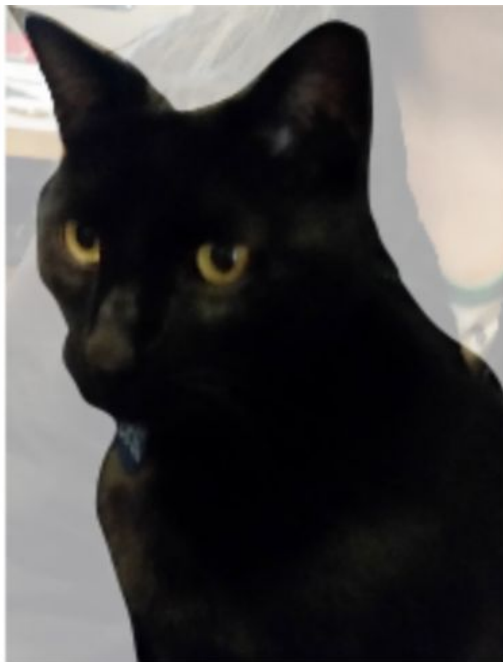
WaveGAN



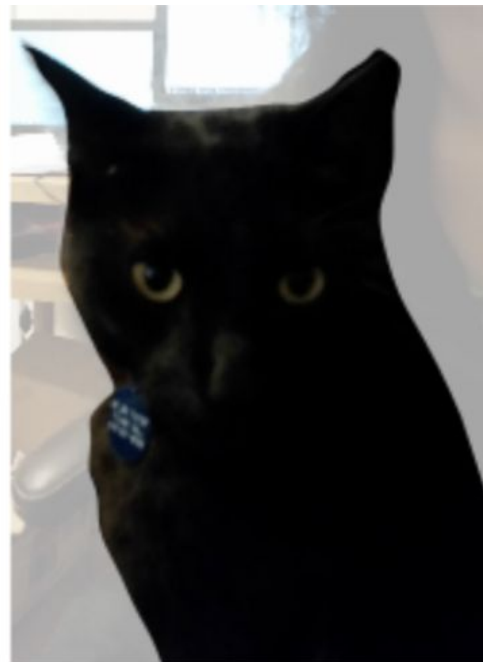
WaveGAN



Non GAN-activated cat

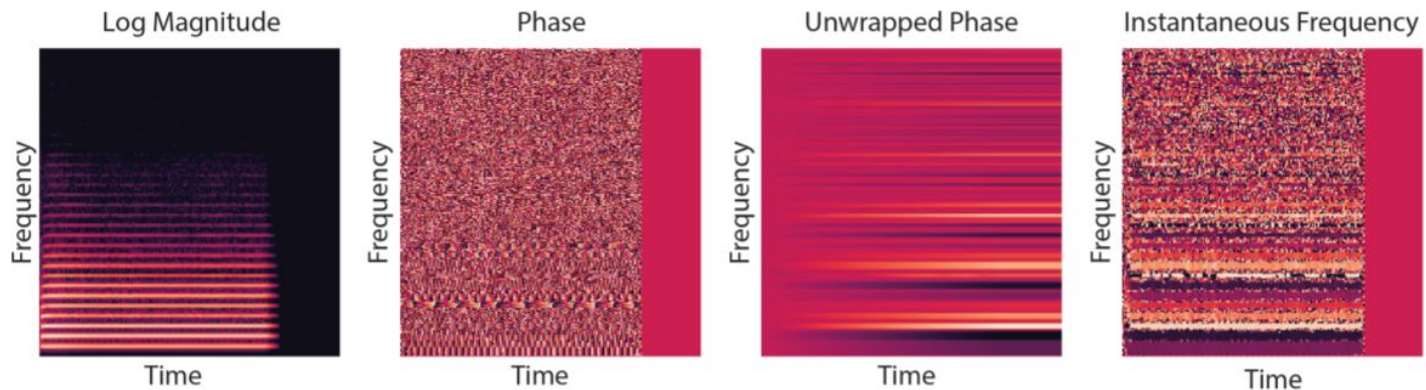
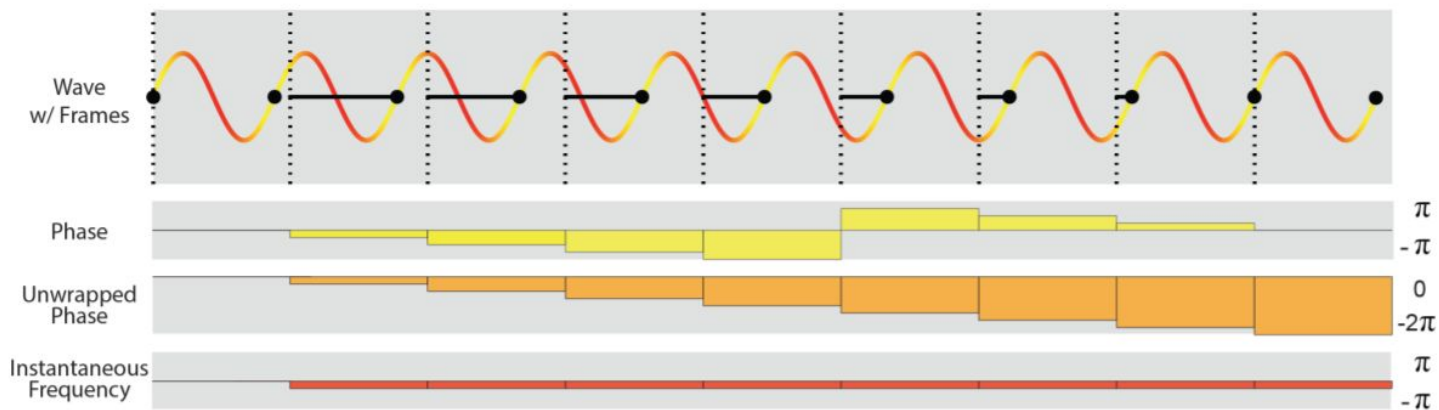


WaveGAN activated cat

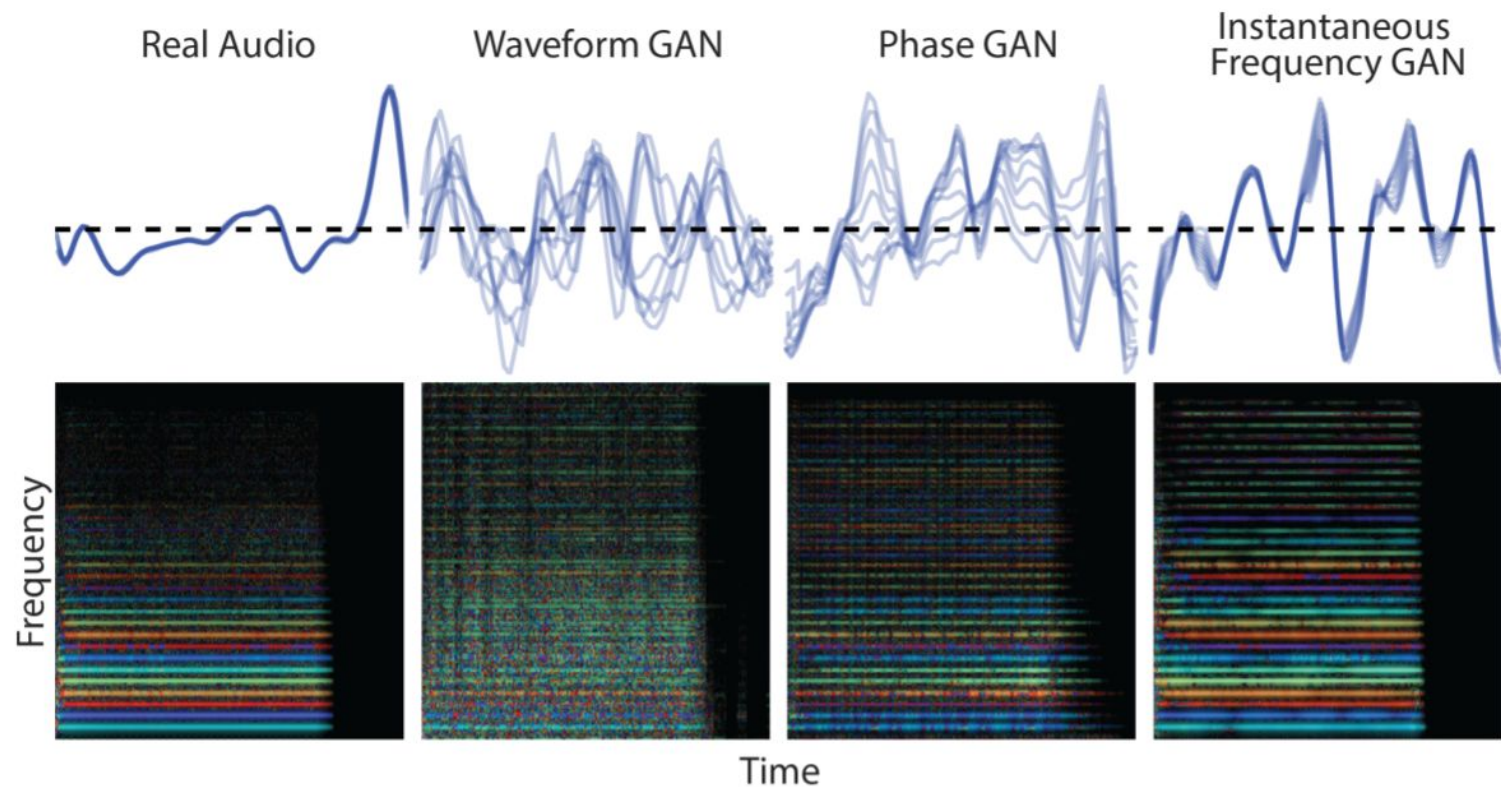


SpecGAN activated cat

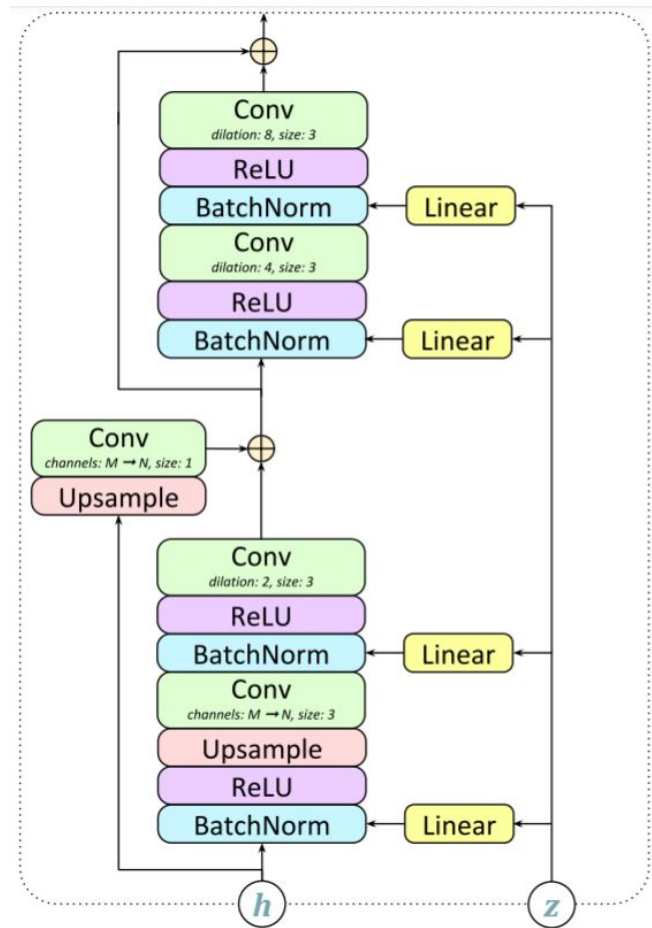
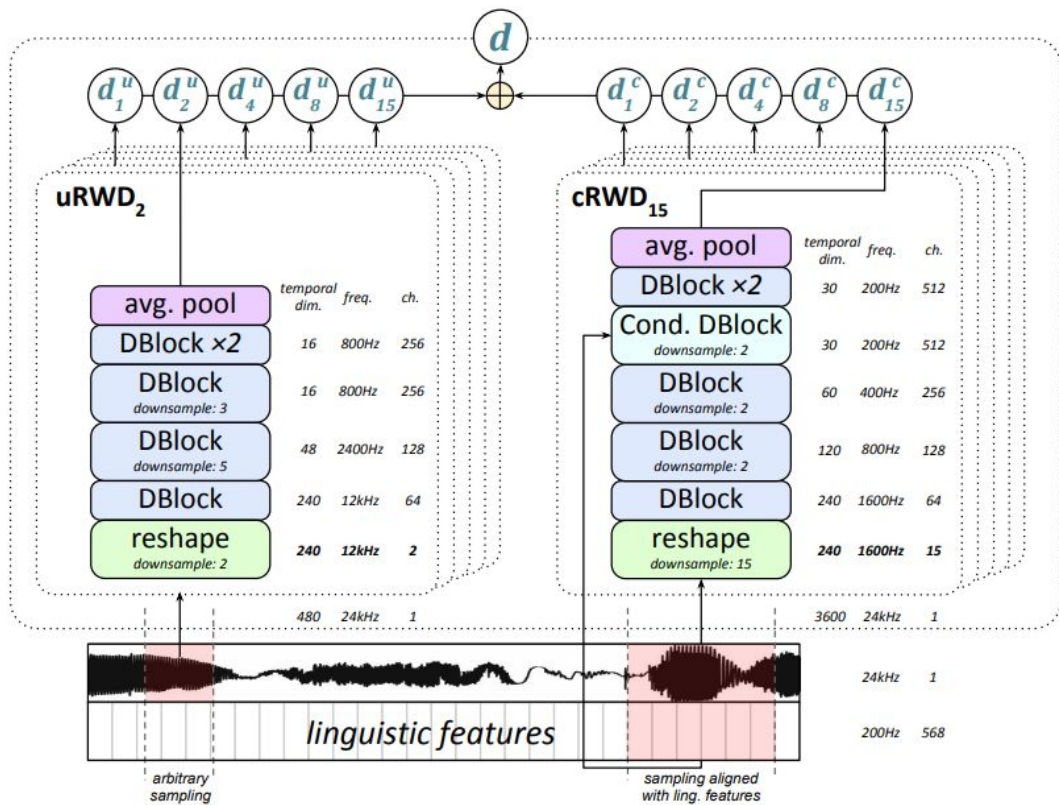
GANSynth



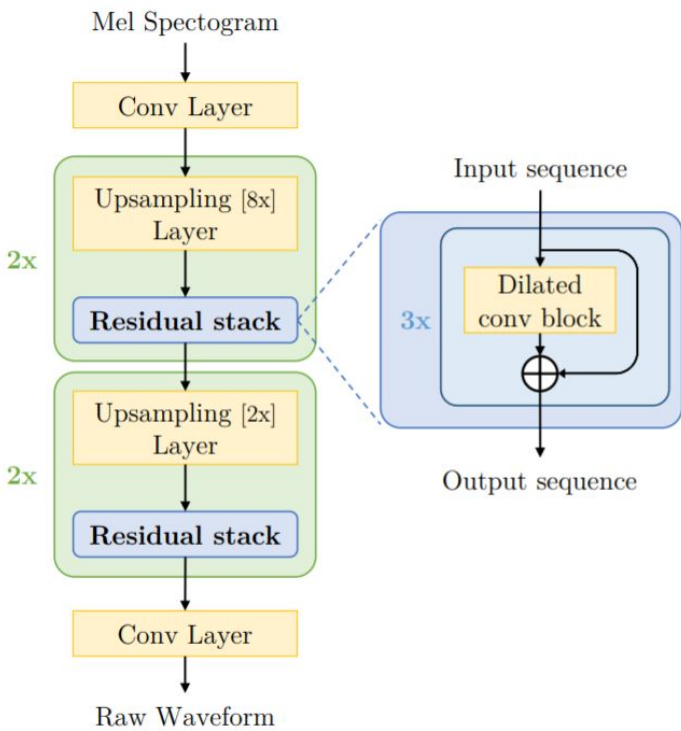
GANSynth



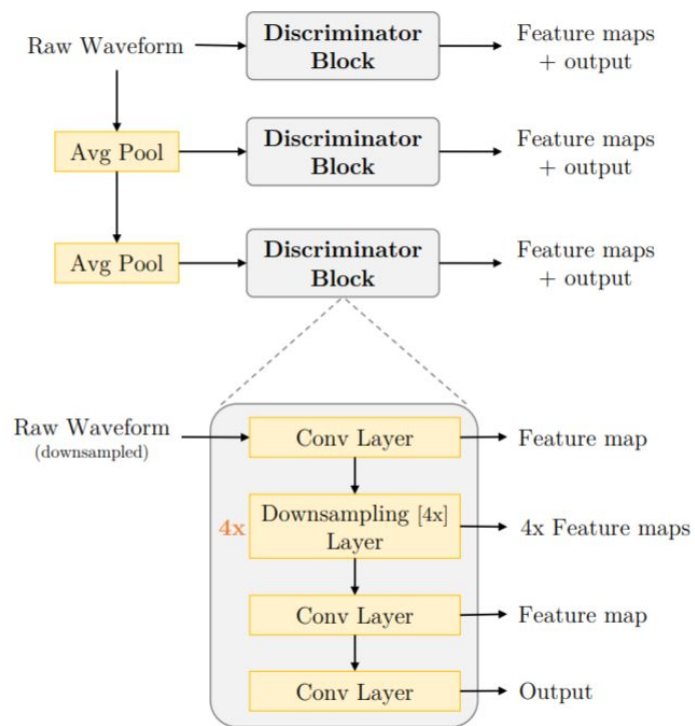
GAN-TTS



MelGAN



(a) Generator



(b) Discriminator

Why the emphasis on likelihood in music modelling?

Most popular generative modelling paradigm:

GANs

Why the emphasis on likelihood in music modelling?

Most popular generative modelling paradigm:

GANs

Most popular generative modelling paradigm for music:

likelihood-based (autoregressive)

Why the emphasis on likelihood in music modelling?

- We are still figuring out the right architectural priors for audio discriminators
 - For images, a stack of convolutions is all you need
 - What do we need for audio? Multiresolution? Dilation? Something phase shift invariant?

Why the emphasis on likelihood in music modelling?

- We are still figuring out the right architectural priors for audio discriminators
 - For images, a stack of convolutions is all you need
 - What do we need for audio? Multiresolution? Dilation? Something phase shift invariant?
- The sparsely-conditioned setting is dominant
 - We care about “creativity” and capturing diversity
 - GANs are worse at this than likelihood-based models

Ulyanov et al., 2018. “Deep Image Prior”, CVPR.

Pons et al., 2019. “Randomly weighted CNNs for (music) audio classification”, ICASSP.

Alternatives to modelling raw audio directly

- Model complex-valued spectrograms and “deal” with phase (GANSynth)
- Model magnitude spectrograms
Use a vocoder or Griffin-Lim to invert
(Tacotron 1 & 2, MelGAN, MelNet, ...)

Wang et al., 2017. “Tacotron: Towards end-to-end speech synthesis”, ISCA.

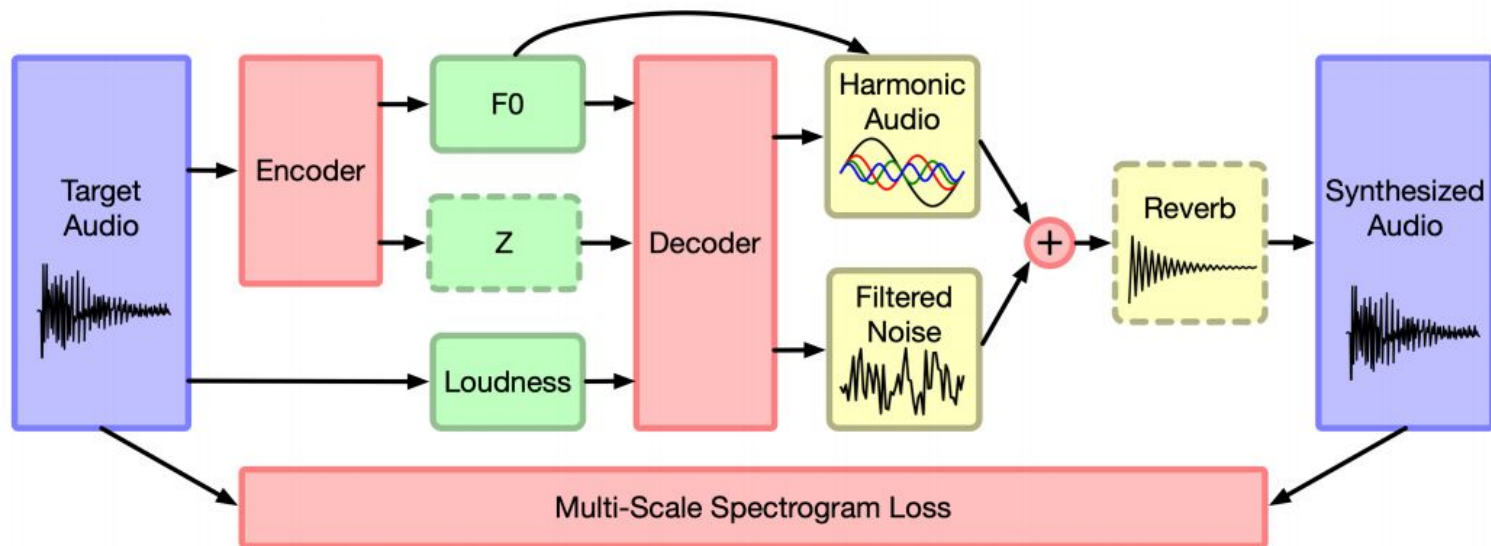
Shen et al., 2018. “Natural TTS Synthesis by Conditioning WaveNet on Mel Spectrogram Predictions”, ICASSP.

Vasquez & Lewis, 2019. “MelNet: A Generative Model for Audio in the Frequency Domain”, arXiv.

Alternatives to modelling raw audio directly

- Differentiable Digital Signal Processing (anonymous authors, ICLR, in review)
Use raw audio input, but put DSP components in the model

<https://openreview.net/forum?id=B1x1ma4tDr>



Summary

- Generative modelling of raw audio is feasible, even in the sparsely conditioned setting
- Likelihood-based models dominate, but GANs are making in-roads in the densely conditioned setting
- Modelling large-scale structure from raw audio is an unsolved problem



Thank you

Sander Dieleman, Jordi Pons, Jongpil Lee