Proc. 7th IEEE Int. Workshop on Mobile Multimedia Computing (MMC2020) at the IEEE Int. Conf. on Multimedia and Expo (ICME), July 2020. Author's accepted version. The final publication is available at https://doi.org/10.1109/ICMEW46912.2020.9105979

FRACTIONAL STEP DISCRIMINANT PRUNING: A FILTER PRUNING FRAMEWORK FOR DEEP CONVOLUTIONAL NEURAL NETWORKS

Nikolaos Gkalelis, Vasileios Mezaris

CERTH-ITI, 6th Km Charilaou-Thermi Road, P.O. BOX 60361 Thessaloniki, Greece
{gkalelis, bmezaris}@iti.gr

ABSTRACT

In this paper, a novel pruning framework is introduced to compress noisy or less discriminant filters in small fractional steps, in deep convolutional networks. The proposed framework utilizes a class-separability criterion that can exploit effectively the labeling information in annotated training sets. Additionally, an asymptotic schedule for the pruning rate and scaling factor is adopted so that the selected filters' weights collapse gradually to zero, providing improved robustness. Experimental results on the CIFAR-10, Google speech commands (GSC) and ImageNet32 (a downsampled version of ILSVRC-2012) show the efficacy of the proposed approach[1].

Index Terms— Deep convolutional neural networks, asymptotic filter pruning, class-separability criteria

1. INTRODUCTION

Deep convolutional neural networks (DCNNs) have shown outstanding classification performance in a variety of application domains. However, limitations in the computational capabilities of resource-limited devices such as IoT and mobile devices, inhibit the use of top-performing DCNNs in these areas. Many approaches have been proposed to reduce the space requirements and accelerate large DCNNs, including quantization, low-rank approximations, and other [1, 2].

Pruning is recently getting increasing attention due to the fact that it can be used in combination with most of the other compression and acceleration methods. It typically consists of a filter importance estimation criterion and a pruning strategy. For instance, in [3], the l_1 norm filter selection criterion is utilized with different pruning rates per layer. In [4], a Taylor expansion-based criterion is proposed to approximate the accuracy loss of pruned feature maps. In [5], a LASSO regression framework is utilized. In [6], low-cost collaborative layers are used to prune filters with zero responses after the ReLU activation. A hybrid algorithm is introduced in [7], combining low rank approximation, quantization and pruning. In [8], filter- and shape-wise scaling factors are used for filter weakening and pruning. In [9], filters are pruned iteratively along epochs using an l_2 norm-based criterion. The above method is extended in [10] using an asymptotic filter pruning schedule. In [11], a trained DCNN and a k-means-based criterion are utilized for filter representation and pruning. In [12], the geometric median (GM) is used to design a criterion for selecting the filters with the most replaceable contribution.

As described above, most pruning approaches in the literature utilize energy preserving-based criteria for filter selection, which may be suboptimal from the perspective of classification [13, 14]. To this end, instead of energy preservation measures, we consider the between-class scatter matrix, which is a popular class-separability measure utilized for extracting the most effective features for classification, and combine it with the GM-based criterion presented in [12]. Furthermore, inspired from [14] we adapt and extend the approach of [10] so that both the pruning rate and filter weights' scaling factor asymptotically approximate their target values. In this way, selected filters are not pruned immediately, but instead are "squeezed" towards zero in small fractional steps, preserving more effectively the capacity of the network and yielding a more stable procedure.

The rest of the paper is structured as follows: The problem is formulated in Section 2. The proposed pruning framework is presented in Section 3. Experiments are described in Section 4 and conclusions are drawn in Section 5.

2. FORMULATION

Consider a CNN $\mathcal{W}$ with l convolutional layers and c filters, each filter denoted as:

$$\mathcal{V}^{(i,j)} \in \mathbb{R}^{k\times k\times c_{i-1}}, i = 1,\ldots,l,\ j = 1,\ldots,c_i, \quad (1)$$

where i is the layer index, j and c_i are the filter index[2] and number of filters in the ith layer, respectively, k is the kernel size and $c = \sum_{i=1}^{l} c_i$. Suppose an annotated training dataset $\mathfrak{D}$ of n observations belonging to m disjoint classes

$$\mathfrak{D} = \{(\mathcal{X}_1^{(0)}, \mathbf{y}_1), \ldots, (\mathcal{X}_n^{(0)}, \mathbf{y}_n)\}, \quad (2)$$

[1]Source code is made publicly available at: `https://github.com/bmezaris/fractional_step_discriminant_pruning_dcnn`

[2]Note that in the following j is also used as channel index of feature maps, which is a common practice in the literature.

Proc. 7th IEEE Int. Workshop on Mobile Multimedia Computing (MMC2020) at the IEEE Int. Conf. on Multimedia and Expo (ICME), July 2020. Author's accepted version. The final publication is available at https://doi.org/10.1109/ICMEW46912.2020.9105979

where, $\mathcal{X}_\kappa^{(0)} = [\mathcal{X}_\kappa^{(0,1)}, \ldots, \mathcal{X}_\kappa^{(0,c_0)}] \in \mathbb{R}^{h_0 \times w_0 \times c_0}$ is the tensor representing the κth observation, h_0, w_0, c_0 are the tensor's height, width and number of channels, respectively, $\mathcal{X}_\kappa^{(0,j)} \in \mathbb{R}^{h_0 \times w_0}$ is the slice of $\mathcal{X}_\kappa^{(0)}$ corresponding to channel j, $\mathbf{y}_\kappa \in \mathbb{R}^m$ is the class indicator vector, i.e. its pth element is one if $\mathcal{X}_\kappa^{(0)}$ belongs to class υ_p and zero otherwise, and υ_p denotes the pth class. In modern CNN architectures, the feature map $\mathcal{X}_\kappa^{(i,j)} \in \mathbb{R}^{h_i \times w_i}$ corresponding to filter (i,j) and observation κ is typically computed using

$$\mathcal{X}_\kappa^{(i,j)} = \mathsf{ReLU}(\mathsf{BN}(\sum_{\tau=1}^{c_{i-1}} \mathcal{X}_\kappa^{(i-1,\tau)} * \mathcal{V}_{:,:,\tau}^{(i,j)})), \tag{3}$$

where, $\mathsf{BN}()$, $\mathsf{ReLU}()$ are the batch normalization and rectified linear unit operators, respectively, h_i, w_i are the height and width of $\mathcal{X}_\kappa^{(i,j)}$, $\mathcal{V}_{:,:,\tau}^{(i,j)}$ is the slice of (i,j) filter corresponding to the τth input channel and $*$ is the two-dimensional convolution operator. Given a target pruning rate θ and an importance estimation mapping η, in most approaches the set $\mathfrak{F}^{(i)}$ of filters to prune in layer i is computed using the following criterion

$$\mathfrak{F}^{(i)} = \mathsf{argmin}(\boldsymbol{\eta}^{(i)}, \theta c_i), \tag{4}$$

where, $\boldsymbol{\eta}^{(i)} = [\eta^{(i,1)}, \ldots, \eta^{(i,c_i)}]^T$ is a vector whose component $\eta^{(i,j)}$ is the output of η associated with the filter (i,j) and the use of the $\mathsf{argmin}(,)$ vector operator above returns the indexes of the θc_i smaller components in $\boldsymbol{\eta}^{(i)}$.

3. PROPOSED METHOD

3.1. Filter importance estimation mapping

3.1.1. Discriminant criterion

Class-separability measures have shown superior performance in comparison to energy preserving criteria in many classification problems [13, 14]. To this end, we resort to the trace of the between-class scatter matrix for designing a suitable criterion for network pruning. For simplicity of notation the indexes (i,j) are dropped in this section as the analysis in the following is valid for any filter in the network. The feature maps, $\mathcal{X}_\kappa \in \mathbb{R}^{h \times w}, \kappa = 1, \ldots, n$, associated with a specific filter can be vectorized and stacked column-wise to form a matrix $\mathbf{X} = [\mathbf{x}_1, \ldots, \mathbf{x}_n] \in \mathbb{R}^{f \times n}$, where $\mathbf{x}_\kappa = \mathsf{vec}(\mathcal{X}_\kappa)$, $\mathbf{x}_\kappa \in \mathbb{R}^f$, $f = hw$, $\mathsf{vec}()$ is the vectorization operator, and h, w are the height and width of the feature maps. The filter discriminant score is then computed using

$$\hat{\eta} = \mathsf{tr}(\mathbf{S}), \tag{5}$$

where, $\mathsf{tr}()$ is the matrix trace operator, $\mathbf{S}$ is a variant of the between-class scatter matrix defined as

$$\mathbf{S} = \sum_{p=1}^{m-1} \sum_{q=p+1}^{m} (\boldsymbol{\mu}_p - \boldsymbol{\mu}_q)(\boldsymbol{\mu}_p - \boldsymbol{\mu}_q)^T, \tag{6}$$

Algorithm 1: Fractional step discriminant pruning

Input: $\mathcal{W}$, $\mathfrak{D}$ (2), δ, ε, θ (10), (11), $\hat{\vartheta}_f$ (13)
Output: Pruned $\mathcal{W}$

```
Compute α, β, γ in (10) using (12)
for ι ← 1 to ε do
    Update CNN parameters using training set 𝔇 (2)
    Compute pruning rate ϑ_ι (10) and scaling ζ_ι (11)
    Compute pruning rates ϑ̂_ι (13), ϑ̃_ι (14) for the
      discriminant and GM-based criterion
    for i ← 1 to l do
        for j ← 1 to c_i do
            Compute discriminant score η̂^(i,j) (5)
        Form set 𝔉^(i) (4) with the indexes of the ϑ̂_ι c_i
          filters with smaller η̂^(i,j)
        for j ← 1 to c_i; (i,j) ∉ 𝔉^(i) do
            Compute GM-based score η̃^(i,j) (9)
        Add to 𝔉^(i) (4) the indexes of the ϑ̃_ι c_i filters
          with smaller η̃^(i,j)
        Scale the weights of the filters in 𝔉^(i) using ζ_ι
Prune the filters in 𝔉^(i) (4) ∀i
```

and $\boldsymbol{\mu}_p = \frac{1}{n_p} \sum_{\mathbf{x}_\kappa \in \upsilon_p} \mathbf{x}_\kappa$, n_p, are the estimated mean vector and cardinality of class υ_p. The computation of $\mathbf{S}$ using (6) is susceptible to memory restrictions and does not fully utilize the parallelization capabilities of modern GPUs. To this end, the matrix $\mathbf{M}$ of class means can be factorized as

$$\mathbf{M} = [\boldsymbol{\mu}_1, \ldots, \boldsymbol{\mu}_m] = \mathbf{XRA}, \tag{7}$$

where, $\mathbf{R} = [\mathbf{y}_1, \ldots, \mathbf{y}_n]^T \in \mathbb{R}^{n \times m}$ is the class indicator matrix and $\mathbf{A} \in \mathbb{R}^{m \times m}$ is a diagonal matrix defined as $\mathbf{A} = \mathsf{diag}(\frac{1}{n_1}, \ldots, \frac{1}{n_m})$. For large-scale datasets it is infeasible to load the whole matrix $\mathbf{X}$ into the memory, and the same is true for $\mathbf{R}$ when the number of classes is large. Instead, assuming that these matrices are partitioned to t blocks, i.e., $\mathbf{X} = [\mathbf{X}_1, \ldots, \mathbf{X}_t]$, $\mathbf{R} = [\mathbf{R}_1^T, \ldots, \mathbf{R}_t^T]^T$, where $\mathbf{R}_\jmath$ is the indicator matrix corresponding to $\mathbf{X}_\jmath$, the matrix product $\mathbf{XR}$ can be computed very efficiently in the GPU using $\mathbf{XR} = \sum_{\jmath=1}^{t} \mathbf{X}_\jmath \mathbf{R}_\jmath$. Finally, $\mathbf{S}$ can be factorized as follows

$$\begin{aligned}
\mathbf{S} &= \sum_{p=1}^{m-1} \sum_{q=p+1}^{m} (\boldsymbol{\mu}_p \boldsymbol{\mu}_p^T + \boldsymbol{\mu}_q \boldsymbol{\mu}_q^T - \boldsymbol{\mu}_p \boldsymbol{\mu}_q^T - \boldsymbol{\mu}_q \boldsymbol{\mu}_p^T) \\
&= (m-1) \sum_{p=1}^{m} \boldsymbol{\mu}_p \boldsymbol{\mu}_p^T - \sum_{p=1}^{m-1} \boldsymbol{\mu}_p (\sum_{q=p+1}^{m} \boldsymbol{\mu}_q^T) \\
&\quad - \sum_{p=1}^{m-1} (\sum_{q=p+1}^{m} \boldsymbol{\mu}_q) \boldsymbol{\mu}_p^T \\
&= (m-1)\mathbf{M}\mathbf{M}^T - \mathbf{M}(\mathbf{J} - \mathbf{I})\mathbf{M}^T \\
&= \mathbf{M}(m\mathbf{I} - \mathbf{J})\mathbf{M}^T,
\end{aligned} \tag{8}$$

Proc. 7th IEEE Int. Workshop on Mobile Multimedia Computing (MMC2020) at the IEEE Int. Conf. on Multimedia and Expo (ICME), July 2020. Author's accepted version. The final publication is available at https://doi.org/10.1109/ICMEW46912.2020.9105979

where, $\mathbf{I}$ and $\mathbf{J}$ are the $m \times m$ identity and all-one matrices respectively. Using the last expression and provided that the matrix of class means has been derived, the between-class scatter matrix can be computed very efficiently in the GPU.

3.1.2. Geometric median-based criterion

In more challenging problems, the application of the discriminant criterion with a relatively large pruning rate may eliminate filters with small but still important discriminant information, harming the classification performance of the network (as for instance we have seen in the experimental evaluation of ImageNet32 in Section 4.3). To this end, we select a fraction of the filters using the discriminant criterion in (5), and from the remaining filters another fraction is selected exploiting a GM-based scoring function defined as [12]

$$\tilde{\eta}^{(i,j)} = \sum_{o=1}^{c_i} ||\mathbf{v}^{(i,j)} - \mathbf{v}^{(i,o)}||_2, \tag{9}$$

where, $\tilde{\eta}^{(i,j)}$ is the importance score for filter (i,j) and $\mathbf{v}^{(i,j)} = \mathsf{vec}(\mathcal{V}^{(i,j)})$.

3.2. Fractional step pruning strategy

Most recent approaches select and prune θc filters at every epoch of the pruning procedure. When the pruning rate is large, following the above strategy will reduce abruptly the network capacity and at the same time discard a large amount of discriminant information [9, 10]. To alleviate these drawbacks, an asymptotic soft filter pruning approach was presented in [10]. Inspired from [14], here we extend the pruning strategy of [10] so that the parameters of the selected filters are not set to zero at every epoch, but on the contrary are multiplied with a scaling factor that decreases from one to zero along the training procedure. Thus, the selected filters are compressed in small fractional steps towards zero and the capacity of the network decreases smoothly. In more detail, assuming that ε, θ are the total epochs and desired pruning rate, respectively, the pruning rate ϑ_ι and scaling factor ζ_ι at epoch ι are computed using the following asymptotic schedule

$$\vartheta_\iota = \alpha \exp(-\beta\iota) + \gamma, \tag{10}$$

$$\zeta_\iota = 1 - \frac{\vartheta_\iota}{\theta}, \tag{11}$$

where, $\vartheta_\varepsilon = \theta$, and α, β, γ are the parameters of the asymptotic function. Similarly to [10], the estimation of these three parameters is performed using the following three points for the epoch and pruning rate tuple $\{\iota, \theta_\iota\}$

$$\{0,0\},\ \{\delta\varepsilon,\ \frac{3}{4}\theta\},\ \{\varepsilon,\theta\}, \tag{12}$$

where $\delta \in (0,1)$. Moreover, the individual pruning rates at each epoch, $\hat{\vartheta}_\iota$, $\tilde{\vartheta}_\iota$ for the discriminant and GM-based criterion, respectively, are computed as follows

$$\hat{\vartheta}_\iota = \mathsf{min}(\vartheta_\iota, \hat{\vartheta}_f), \tag{13}$$

$$\tilde{\vartheta}_\iota = \vartheta_\iota - \hat{\vartheta}_\iota, \tag{14}$$

where $\hat{\vartheta}_f$ is a parameter denoting the final pruning rate associated with the discriminant criterion. We observe that the sum of the two individual rates always equals to the total rate at each epoch ($\hat{\vartheta}_\iota + \tilde{\vartheta}_\iota = \vartheta_\iota$) and the pruning rate $\hat{\vartheta}_\iota$ associated with the discriminant criterion (5) is never larger than $\hat{\vartheta}_f$, ensuring that filters with small but possible significant information are not pruned.

The pseudocode of the proposed method is shown in Algorithm 1. We should note that the only input parameters of the method are δ (12) and $\hat{\vartheta}_f$ (13), which based on the related literature and our experimental evaluation in Section 4.3, are typically set to $\frac{1}{8}$ and 10%, respectively (e.g. see ASFP [10] and FPGM [12]).

4. EXPERIMENTS

4.1. Datasets

For the evaluation of the proposed fractional step discriminant pruning approach, denoted hereafter as FSDP, two image and one speech datasets are used as described in the following: i) CIFAR-10 [15]: It consists of 50000 training and 10000 testing color images of 32×32 resolution, belonging to one of 10 different classes. ii) ImageNet32 [16]: This dataset consists of the annotated images of ILSVRC-2012 resized to 32×32 resolution. It contains 1331167 images in total belonging to one of 1000 classes. It is divided to a training and testing partition of 1281167 and 50000 images, respectively. iii) GSC [17]: This is version 0.01 of the Google speech commands dataset. It consists of 64727 utterances and 12 categories representing short commands such as "No", "Up" and "Go". A training, validation and testing partition is provided consisting of 51094, 6798 and 6835 utterances, respectively.

4.2. Setup

The CIFAR-10 dataset is used to evaluate the proposed FSDP against several top-performing approaches, namely, MIL [6], PFEC [3], CP [5], SFP [9], ASFP [10] and FPGM [12]. Three popular ResNet architectures (ResNet-20, -56, -110) [18] along different pruning rates θ are used in the evaluation, following the experimental setup in [9, 10, 12]. Specifically, each image is normalized to zero mean and unit variance, and data augmentation is applied during training, i.e., 32×32 random cropping and horizontal flipping with 50% probability. The networks are trained using minibatch stochastic gradient descent (SGD) with Nesterov momentum of 0.9, batch size of 128 and weight decay of 0.0005. The total number of epochs is set to $\epsilon = 200$ and the learning rate starts at 0.01 and is divided by 5 at epochs 60, 120 and 160, as in [9, 10, 12].

Proc. 7th IEEE Int. Workshop on Mobile Multimedia Computing (MMC2020) at the IEEE Int. Conf. on Multimedia and Expo (ICME), July 2020. Author's accepted version. The final publication is available at https://doi.org/10.1109/ICMEW46912.2020.9105979

Next, the ImageNet32 dataset is used to evaluate the proposed approach for large-scale image classification. In this experiment, the proposed FSDP is compared against SFP [9] and FPGM [12] using ResNet-56, pruning rates $\theta = 20\%, 50\%$ and number of epochs $\epsilon = 40$. Following [16], each image is normalized to zero mean and the minibatch SGD is used, where momentum, batch size and weight decay are set to 0.9, 128 and 0.0005, respectively. The learning rate starts at 0.01 and is multiplied with 0.1 every 10 epochs.

Finally, the GSC dataset is employed to evaluate SFP [9], FPGM [12], and FSDP for the task of speech command classification using ResNet-56 and pruning rates $\theta = 20\%, 50\%$. The training and validation sets of GSC are used for training, while the testing set is used for the performance evaluation. Log mel-spectrograms (LMSs) are used to represent the speech commands in GSC. More specifically, the one-second long speech recordings are re-sampled to 16 KHz, the STFT with Hamming window of size 1024 and hop length 512 is applied, and subsequently, 32 mel filterbanks and the logarithmic operator are used to map the power spectra to the log-mel space, and retrieve a 32×32 LMS for each recording. Moreover, following [19], the training dataset is augmented using time-stretching, pitch-shifting and mixing with background noise. The network is trained using minibatch SGD, number of epochs $\epsilon = 70$, and with momentum, weight decay and batch size set to 0.9, 0.0005 and 96, respectively. The initial learning rate starts at 0.01 and is divided by 10 at epoch 50.

In the various experiments, the following input parameters for FSDP (Algorithm 1) are used: i) the asymptotic schedule parameter δ for the estimation of α, β and γ in (10) is set to $\delta = \frac{1}{8}$ as in [10], ii) in order to gain insight into the effect of the target pruning rate associated with the discriminant criterion $\hat{\theta}_f$ (13), we test different values of it, i.e., 10%, 40% on CIFAR-10 (see Tables 1, 2), and in all other experiments this is set to 10%, as explained in Section 3.2. Moreover, the between-class scatter matrix $\mathbf{S}$ (6) in the discriminant criterion (5) is computed using the whole training set for CIFAR-10, 20% of the training set for ImageNet32 and the validation set (which is part of the overall training set) for GSC.

FSDP is implemented in PyTorch, while for SFP and FPGM the implementations in [9, 12] are used. The experiments are performed in an Intel i7 3770K@3.5GHz PC with 32 GB RAM and Nvidia GeForce GTX 1080Ti GPU.

4.3. Results

The evaluation results in terms of accuracy rates along different methods on CIFAR-10, three ResNet architectures (ResNet-20, ResNet-56, ResNet-110) and pruning rate 40% are shown in Table 1, while, Table 2 depicts the accuracy rates of the proposed FSDP and FPGM under the same settings and pruning rate 50%. In Figure 1, we vary the pruning rate from 10 to 90 in order to study its effect in the performance of the proposed method along different network depths. Finally, the evaluation results on ImageNet32 and GSC using ResNet-56 and pruning rates 20% and 50% are shown in Tables 3 and 4. From the obtained results we conclude the following:

i) The proposed FSDP achieves the best performance in all experiments. For instance, with 50% pruning an accuracy improvement of more than 1% of FSDP over FPGM is observed for ResNet-110 in the CIFAR-10 (Table 2) and for ResNet-56 in the GSC dataset (Table 4). Similarly, in Ima-

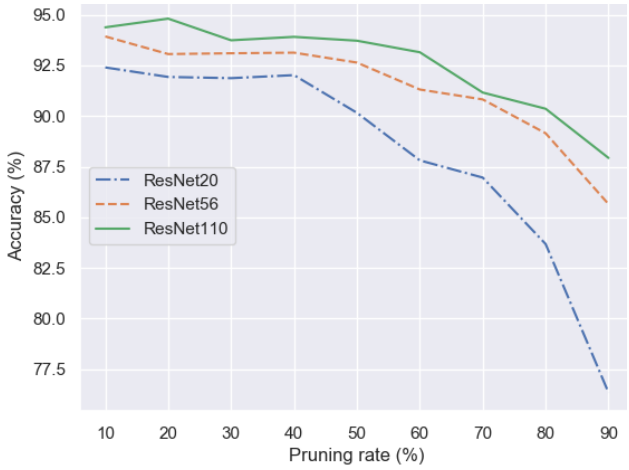

Fig. 1. Accuracy rates of FSDP with $\hat{\theta}_f = 10\%$ on CIFAR-10 along different ResNet architectures and pruning rates.

Table 1. Accuracy rates on CIFAR-10 along different ResNet architectures and pruning rate 40%.

	ResNet-20	*ResNet-56*	*ResNet-110*
no pruning	92.2%	93.59%	93.68%
MIL [6]	91.43%	–	93.44%
PFEC [3]	–	91.31%	92.94%
CP [5]	–	90.90%	–
SFP [9]	90.83%	92.26%	93.38%
ASFP [10]	–	92.44%	93.20%
FPGM [12]	91.99%	92.89%	93.85%
FSDP ($\hat{\theta}_f = 10\%$)	92.02%	**93.13%**	93.91%
FSDP ($\hat{\theta}_f = 40\%$)	**92.09%**	93.1%	**93.99%**

Table 2. Accuracy rates on CIFAR-10 along different ResNet architectures and pruning rate 50%.

	ResNet-20	*ResNet-56*	*ResNet-110*
FPGM [12]	89.73%	91.79%	92.51%
FSDP ($\hat{\theta}_f = 10\%$)	**90.16%**	**92.64%**	**93.72%**

Proc. 7th IEEE Int. Workshop on Mobile Multimedia Computing (MMC2020) at the IEEE Int. Conf. on Multimedia and Expo (ICME), July 2020. Author's accepted version. The final publication is available at https://doi.org/10.1109/ICMEW46912.2020.9105979

Table 3. Accuracy rates in ImageNet32 and GSC with ResNet-56 and pruning rate 20%.

	ImageNet32	*GSC*
no pruning	40.79%	97.47%
SFP [9]	29.92%	94.57%
FPGM [12]	37.23%	95.64%
FSDP ($\hat{\theta}_f = 10\%$)	**38.3%**	**96.22%**

Table 4. Accuracy rates of FSDP and FPGM in ImageNet32 and GSC with ResNet-56 and pruning rate 50%.

	ImageNet32	*GSC*
FPGM [12]	32.32%	92.89%
FSDP ($\hat{\theta}_f = 10\%$)	**33.23%**	**94.66%**

geNet32 (Table 3), the proposed FSDP with $\hat{\theta}_f = 10\%$ outperforms the other two methods. This is attributed to the stability of the fractional pruning procedure and the ability of the proposed discriminant criterion to identify the filters with negligible discriminant information. We also see that that the application of FSDP to ResNet-110 in the CIFAR-10 experiments (Tables 1 and 2) yields an increase in performance over the baseline model without pruning. Therefore, we may conclude that the use of FSDP to reduce the capacity of large networks has a regularization effect. This is not observed in the experiments with the smaller networks (ResNet-20, -56) where as expected pruning reduces slightly the performance. Another interesting observation is that SFP appears to have a more than 10% performance drop in ImageNet32 (Table 3), which is due to the much more challenging problem, where a percentage of the pruned filters selected using the l_2 norm criterion still carry important discriminant information.

ii) As shown in Fig. 1, FSDP provides a quite high robustness for pruning rates less than 40%. We also observe that ResNet-110 exhibits the more stable behavior, with even an increase in performance for pruning rates less than 25%, only a small performance drop for pruning rates 40% and 50%, and accuracy of more than 90% even with 80% pruning rate. Finally, we see that the performance of ResNet-20 degrades rapidly with pruning rates higher than 40%, which indicates that network depth is an important parameter concerning the robustness of a network against pruning.

iii) Concerning training times, an overhead of several seconds to a few minutes at epoch level (depending on the dataset) is observed when FSDP is used. For instance, on CIFAR-10 and ResNet-20 this time is increased from 17 to 42 seconds, while for ResNet-110 an overhead of 1.5 minutes, i.e. going up from 1.7 to 3.2 minutes, is observed. However, concerning that the training is performed off-line, its duration is less than a day in all experiments, and that the computation of the discriminant criterion (5) can be accelerated significantly using a more powerful GPU, this time overhead is considered insignificant.

4.4. Visualization of the proposed approach

In order to gain further insight into the proposed framework, Fig. 2 illustrates the discriminant scores of each filter along different epochs during the training of ResNet-20 with FSDP on CIFAR-10 and pruning rates $\theta = \hat{\theta}_f = 20\%$ (i.e. only the discriminant criterion (5) is used for filter selection and pruning in this experiment). The heatmaps on the left, middle and right correspond to the state of the network at epochs 1, 40 and 200, while, the x-, y-axis at each heatmap represent the filter and convolutional layer number, respectively. A cool-to-warm color spectrum is used to represent the filter discriminant score at logarithmic-scale, i.e., darker squares represent less important filters while lighter ones correspond to filters with a high discriminant score. We can observe the following:

i) Filters at layers closer to the network input seem to attain a relatively higher discriminant score. This phenomenon seems to be stronger during the first stages of the training procedure and becoming less emphatic as the network converges to its steady-state condition. We also observe a possible correlation between filter's discriminant score and layer's width, i.e., the discriminant power of a filter seems to decrease with the number of filters in the layer it belongs to. For instance, we see that filters of layers 1 to 7 attain in general a higher discriminant score in comparison to the filters of layers 8 to 13. These conclusions are in agreement with similar ones in other literature works, e.g. [4], where it is stated that "global importance seems to decrease with depth".

ii) The last convolutional layer (layer 19) is a clear exception to the above observation, where we see that after a certain number of epochs the majority of its filters attain a large discriminant score. Similarly, another exception are the filters of the second convolutional layers in residual blocks, which also attain a relatively high discriminant score. For instance, this can be seen more clearly at the final network (right heatmap in Fig. 2) for the filters in layers 11, 13, 15 and 17. This conclusion is again in agreement with similar ones in the literature (e.g. see Section 4.3 in [3]).

iii) At epoch 40, most discriminant information has been already concentrated in a rather small portion of the filters. From this, we can conclude that there is a quite high redundancy in the network for the specified problem, and that the proposed approach can effectively discover a more compact network structure already at the early stages of the training.

Proc. 7th IEEE Int. Workshop on Mobile Multimedia Computing (MMC2020) at the IEEE Int. Conf. on Multimedia and Expo (ICME), July 2020. Author's accepted version. The final publication is available at https://doi.org/10.1109/ICMEW46912.2020.9105979

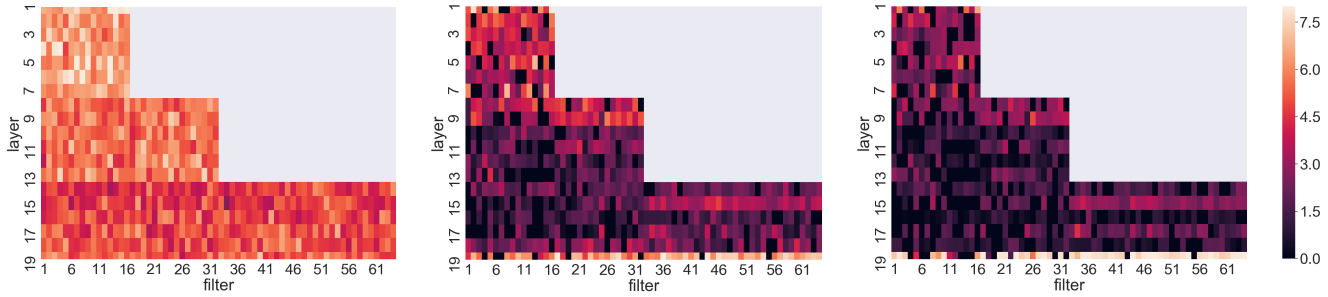

Fig. 2. Heatmaps depicting discriminant scores of ResNet-20 filters computed using the discriminant criterion (5) during the application of FSDP on CIFAR-10 with $\theta = \hat{\theta}_f = 20\%$; the heatmaps (left to right) correspond to epochs 1, 40 and 200.

5. CONCLUSIONS

In this paper, a fractional step discriminant pruning framework was proposed and evaluated on three datasets for the tasks of image and speech classification, providing competitive performance at high pruning rates. A promising future work direction is the investigation of variable pruning rates using the discriminant scores at layer-level (e.g. as indicated in Fig. 2), similarly to globally-comparing criteria in [3, 4].

6. ACKNOWLEDGMENTS

This work was supported by the EUs Horizon 2020 research and innovation programme under grant agreement H2020-780656 ReTV.

7. REFERENCES

[1] K. Ota et al., "Deep learning for mobile multimedia: A survey," *ACM Trans. Multimedia Comput. Commun. Appl.*, vol. 13, no. 3s, pp. 34:1–34:22, June 2017.

[2] Y. Cheng et al., "Model compression and acceleration for deep neural networks: The principles, progress, and challenges," *IEEE Signal Process. Mag.*, vol. 35, no. 1, pp. 126–136, Jan. 2018.

[3] H. Li et al., "Pruning filters for efficient convnets," in *Proc. ICLR 2017*, Toulon, France, Apr. 2017.

[4] P. Molchanov et al., "Pruning convolutional neural networks for resource efficient inference," in *Proc. ICLR 2017*, Toulon, France, Apr. 2017.

[5] Y. He, X. Zhang, and J. Sun, "Channel pruning for accelerating very deep neural networks," in *Proc. ICCV 2017*, Venice, Italy, Oct. 2017, pp. 1398–1406.

[6] X. Dong et al., "More is less: A more complicated network with less inference complexity," in *Proc. CVPR 2017*, Honolulu, HI, USA, July 2017, pp. 1895–1903.

[7] S. Ge et al., "Compressing deep neural networks for efficient visual inference," in *Proc. ICME 2017*, Hong Kong, China, July 2017, pp. 667–672.

[8] Z. Zhou et al., "Online filter weakening and pruning for efficient convnets," in *Proc. ICME 2018*, San Diego, CA, USA, July 2018, pp. 1–6.

[9] Y. He et al., "Soft filter pruning for accelerating deep convolutional neural networks," in *Proc. IJCAI 2018*, Stockholm, Sweden, July 2018, pp. 2234–2240.

[10] Y. He et al., "Asymptotic soft filter pruning for deep convolutional neural networks," *IEEE Trans. Cybern.*, pp. 1–11, Aug. 2019.

[11] L. Li, J. Zhu, and M. Sun, "Deep learning based method for pruning deep neural networks," in *Proc. ICMEW 2019*, Shanghai, China, July 2019, pp. 312–317.

[12] Y. He et al., "Filter pruning via geometric median for deep convolutional neural networks acceleration," in *Proc. CVPR 2019*, Long Beach, CA, USA, June 2019.

[13] K. Fukunaga, *Introduction to statistical pattern recognition (2nd ed.)*, Academic Press Professional, Inc., San Diego, CA, USA, 1990.

[14] N. Gkalelis et al., "Mixture subclass discriminant analysis link to restricted Gaussian model and other generalizations," *IEEE Trans. Neural Netw. and Learning Syst.*, vol. 24, no. 1, pp. 8–21, Jan. 2013.

[15] A. Krizhevsky, "Learning multiple layers of features from tiny images," Tech. Rep., Department of Computer Science, University of Toronto, 2009.

[16] P. Chrabaszcz, I. Loshchilov, and F. Hutter, "A downsampled variant of ImageNet as an alternative to the CIFAR datasets," *CoRR*, vol. abs/1707.08819, 2017.

[17] P. Warden, "Speech commands: A dataset for limited-vocabulary speech recognition," *CoRR*, vol. abs/1804.03209, 2018.

[18] K. He et al., "Deep residual learning for image recognition," in *Proc. CVPR 2016*, Las Vegas, NV, USA, June 2016, pp. 770–778.

[19] J. Salamon and J. P. Bello, "Deep convolutional neural networks and data augmentation for environmental sound classification," *IEEE Signal Process. Lett.*, vol. 24, no. 3, pp. 279–283, Mar. 2017.