WSNET: LEARNING COMPACT AND EFFICIENT NETWORKS WITH WEIGHT SAMPLING

Anonymous authors
Paper under double-blind review

ABSTRACT

We present a novel network architecture for learning compact and efficient deep neural networks, where the weights of convolution filters fully connected layers, instead of learned independently, are sampled from a compact set of parameters that enforces weight sharing in the learning process. Specifically, we consider learning compact and efficient 1D convolutional neural networks for audio classification in this work. We show that our novel way of weight sampling allows not only weight sharing but also computation sharing, and therefore, we can learn much smaller and efficient but yet competitive networks compared to baseline networks with same number of convolution filters. Extensive experiments on multiple audio classification datasets verify the effectiveness of our approach. Combining weight quantization, we demonstrate that we can learn models that are up to 180× smaller than the baselines without noticeable performance drop.

1 INTRODUCTION

Despite deep neural networks (DNNs) have achieved remarkable success in various applications, *e.g.* sound classification, speech recognition, etc., they usually suffer from following two problems caused by their inherent huge parameter space. First, most of state-of-the-art deep architectures are prone to over-fitting even trained on large datasets Simonyan & Zisserman (2015); Szegedy et al. (2015). Secondly, as they usually consume large storage memory and energy Han et al. (2016), DNNs are difficult to embed into devices with limited memory and power (like portable devices or chips). There have been a lot of research on compressing DNNs Han et al. (2015); Anwar et al. (2017); Li et al. (2017); Collins & Kohli (2014). Nevertheless, they either compress the deep models with large sacrifice of classification performance or only achieve relatively small compression ratio (tens of times).

To tackle the above problems, we propose a **W**eight **S**ampling deep neural network (*i.e.* WSNet) to significantly reduce the model size of deep networks, specifically 1D convolution neural networks in this paper, with more than 100× smaller size at negligible loss of performance drop or achieve even better performance than baseline models. In network architecture design, there are two novel components proposed in WSNet: weight sampling and channel sharing. These two components enable the WSNet to learn a compact network with Weight Sharing Filters which are much smaller than that of independently trained filters in baseline model. With weight sampling, all filters that are to perform convolution operations are sampled from the weight sharing filters, and similar weight sampling is also applied to fully connected layer weights. With channel sharing, we first learn the weight sharing filters which have less channels than needed and repeat the channels of extracted filters until reaching the number of channels in actual filters. These two components make it possible to learn highly compact and efficient WSNet. We also propose a filter augmentation approach to make up the accuracy loss caused by compressing weights. Combined with weight quantization method, WSNet is able to further reduce the model size during inferencing without sacrificing the loss of accuracy.

For demonstrating the effectiveness of WSNet, we conduct extensive experiments on the challenging acoustic scene classification and music detection tasks. On all tested datasets, including MusicDet200K (a self-collected dataset, as detailed in Section 4), ESC-50 Piczak (2015a), UrbanSound8K J. et al. (2014) and DCASE Stowell et al. (2015), WSNet significantly reduces the model sizes of the baseline models by 100× with comparable or even higher classification accuracy. When

compressing more than 160×, WSNet is only subject to slight accuracy drop. Above experimental results strongly evident the capability of WSNet on learning efficient and compact networks. Although the experiments in this paper are mostly limited to one dimensional convolution neural networks, the same idea can naturally be extended to 2D CNNs.

In addition, we also propose to use integral image method Viola & Jones (2001) to accelerate WSNet in both training and inference. With densely sampled weight sharing filters, integral image method could dramatically reduce the computation time and make WSNet much faster in computation. The integral image method can also enable sampling filters with different size, from the weight sharing filters with little computational overhead. This could provide much more powerful feature representation with the same number of weights and similar computation.

2 Related Works

2.1 Audio classification

In this paper we considered Acoustic scene classification tasks as well as music detection task. Acoustic scene classification (ASC) is a task focusing on classifying the surrounding environment where an audio stream is generated given the audio input Barchiesi et al. (2015). It can be applied in many different applications such as audio tagging Cai et al. (2006), audio collections management Landone et al. (2007), robotic navigation Chu et al. (2006), intelligent wearable interfaces Xu et al. (2008), context adaptive tasks Schilit et al. (1994), and etc. Music detection is a related task to determine whether a small segment of audio is music or not. It is usually treated as a binary classification problem given an audio segment as input, *i.e.*, to classify the segment into two categories, music or non-music.

Like in many other areas, deep learning based approaches has been recently proven to improve the performance over conventional baselines in audio classification tasks. Among the various types of deep learning approaches proposed, convolutional neural networks (CNN) have been the most popular in learning the representation of the audio signal Valenti et al. (2016) Salamon & Bello (2017).

SoundNet Aytar et al. stands out among different CNNs applied for sound classification. One reason is because the SoundNet is trained from very large amounts of unlabeled sound data using visual information as a bridge, compared to many other networks that are trained with much smaller scale labeled dataset. At the same time, the SoundNet directly takes the one dimensional raw wave signals as input so that there is no need to calculate audio specific features like MFCC Pols (1966) Davis & Mermelstein (1980) and spectrogram Flanagan (1972). The network contains all one dimensional convolutional layers. The soundnet yields significant performance improvements over the state-of-the-art results on standard benchmarks for acoustic scene/object classification.

2.2 Deep Model Compression

Early approaches for deep model compression include optimal brain damage LeCun et al. (1989) and optimal brain surgeon Hassibi & Stork (1993) that prune the connections in networks based on the second order information. However, those methods are not feasible for deep networks due to high computational complexity. Some recent works in network compression include Han et al. (2015); Collins & Kohli (2014); Anwar et al. (2017); Lebedev & Lempitsky (2015); Kim et al. (2015); Luo et al. (2017); Li et al. (2017), in which Han et al. (2015); Anwar et al. (2017) pruned connections in a progressively greedy way. Collins & Kohli (2014); Lebedev & Lempitsky (2015); Kim et al. (2015) used sparsity related regularizer. Li et al. (2017); Luo et al. (2017) focused on filter level pruning by using different criterion. Although those works can reduce model size significantly, they also suffer from dramatic performance loss. Sindhwani et al. (2015); Denton et al. (2014) sought a low-rank approximation of the model Han et al. (2016) combined pruning, quantization and Huffman coding techniques to provide rather high compression ratios. However, those methods also suffer from large performance drop. Iandola et al. (2016) proposed to reduce parameters by re-designing the network architectures with "Fire" module. Jin et al. (2016) proposed an iterative hard thresholding method, but only achieve small compression ratios. There are also works trying to compress a model by using binning methods Gong et al. (2014), but they can only be applied over fully connected layers.

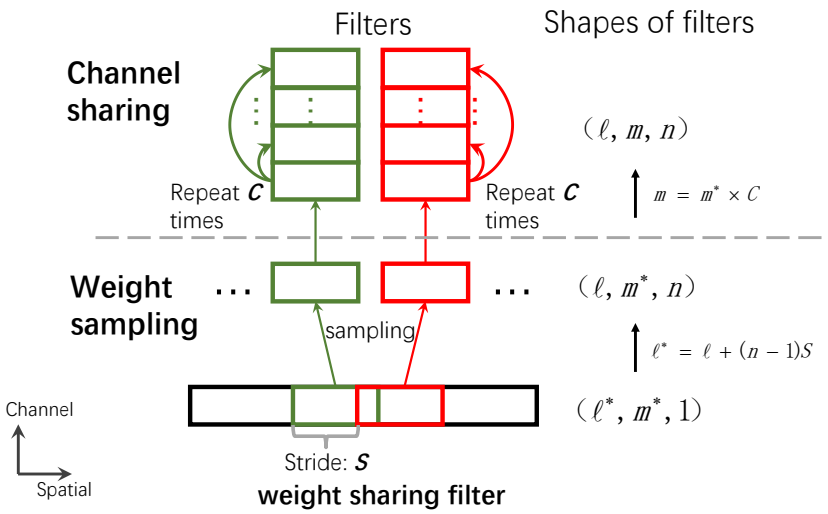

Figure 1: Illustration of two core components in WSNet to compress weights in each layer: weight sampling (the lower panel) and channel sharing (the upper panel). The figure shows the procedure of generating two continuous filters (in green and red respectively). In **weight sampling**, all actual filters are extracted from the weight sharing filter with the stride of S. In **channel sharing**, channels in a filter are shared by repeating C times. Please refer to Section 3 for details.

In contrast, our method can be applied for both convolution layers and fully connected layers. The models trained with WSNet do not only offer large compression ratios compared to the baseline models, but also achieve comparable performance.

3 METHOD

In this section, we describe the technical details of WSNet. Firstly, we present those two most critical components in WSNet: weight sampling and channel sharing. Second, we present a filter augmentation method to compensate the loss caused by the compression. Third, we demonstrate that the weight quantization can be applied on WSNet to further reduce the model size. Finally, we propose an integral image method for accelerating WSNet in both training and inference.

Before diving into the details, we first introduce the notations used in the following text. We use a tuple containing three elements *i.e.* (ℓ, m, n) to represent the shape of weights W in a convolution layer of WSNet. ℓ, m, n denote the filter size, filter channels, and the number of filters that are actually used in convolution. In each layer of WSNet, all filters used in actual convolutions are sampled from a learned weight sharing filter which has shape of (ℓ^*, m^*). The goal of training WSNet is to learn more compact DNNs which satisfy the condition $\ell^* m^* < \ell m n$. We define the compression ratio of a layer as $\frac{\ell m n}{\ell^* m^*}$. In the following, we will show that WSNet reduces the network's size by compressing the weights in two dimensions: the spatial dimension and the channel dimension.

3.1 WEIGHT SAMPLING

During the training phase of traditional DNNs, the filters in a layer are learned independently. There are two disadvantages of this strategy. Firstly, it makes the DNNs have a large number of parameters, which impedes them to be deployed in computation resource constrained platforms. Second, due to the over-parameterization of deep models, the network is prone to overfitting and learns a sub-optimal solution. Therefore, we propose a novel weight sampling method to reuse the weights among filters. Specifically, in WSNet, all filters with the shape (ℓ, m, n) to be used in convolutions are sampled from a weight sharing filter, as illustrated in Figure 1. By scanning the weight sharing

filter with a window size of ℓ and stride of S, we could sample out n filters with filter size of ℓ. Formally, the equation between the filter size of the shared filter and the sampled filters is:

$$\ell^* = \ell + (n-1)S \tag{1}$$

In weight sampling, the compression ratio $\frac{\ell m^* n}{\ell^* m^*} \approx \frac{\ell}{S}$. Note that since the minimal value of S is 1, thus the minimal value of ℓ^* is $\ell + n - 1$ and the maximal compression ratio is therefore ℓ. For the brevity of description, In the following text, S will be used as the value of compression ratio where no confusion is caused. Note we use Eq. 1 to strictly calculate the final compression ratios of WSNet in all tables in the experiments.

3.2 Channel sharing

Although it has been experimentally verified that the weight sampling strategy could compresses deep models with negligible loss of classification accuracy, it has a disadvantage. Just as mentioned in above section 3.1, the maximum compression ratio is near the filter size ℓ. What's more, aggressively reducing the size of filters in the spatial dimension might do harm to the final performance of WSNet due to its learning capability might be largely reduced by forcing sharing large proportions of weights among filters. For seeking more compact networks, we propose a channel sharing strategy to compress DNNs along the channel dimension. As illustrated in Figure 1, each filter after weight sampling is repeated C times to generate the actual filter used in convolution. The equation between the channels of filters before and after channel sharing is:

$$m^* = m \times C \tag{2}$$

In the following text, we use C to denote the compression ratio along the channel dimension. As introduced later in Experiments, we observe that channel sharing is able to significantly reduce the model size while achieving comparable performance to baselines. Note one advantage of channel sharing is that the maximum compression ratio can be as large as m^* which make it possible to more aggressive compression of deep models.

Although above analysis is conducted on convolution layers, the weight sampling and channel sharing can also be conveniently applied to the fully connected layers. For a fully connected layer, we treat its spatial dimension and channel dimension as one dimension, along which the weight sampling is performed to compress the size of weights.

3.3 Filter Augmentation

The performance of WSNet might get adversely affected when performing aggressively model compression (*i.e.* S and C are large). To make up the performance loss, we propose to augment filters for layers with significantly reduced size to enhance their learning capability. Specifically, we sample more filters with stride $\bar{S}$ ($\bar{S} < S$) from the weight sharing filter. In order to ensure the validity of operations in the following layer, we append a 1×1 convolution layer with shape of $(1, \bar{n}, n)$ to reduce the channels of augmented filters. It has been experimentally verified that filter augmentation can effectively boost the performance of WSNet. However since the filter augmentation also brings extra parameters (*i.e.* the weights in the added 1×1 convolution layer, it is generally only used in lower layers where the number of filters (n) is small (so the size of added parameters to WSNet is small). Besides, one can also perform the strategy of channel sharing on the weights of the newly introduced 1×1 convolution layer.

3.4 Weight Quantization

Weight quantization has been used in previous literatures Han et al. (2016) to compresses DNNs by reducing the number of bits required to represent each weight. By making multiple weights share the same weights, the storage space of DNNs can be largely saved. More concretely, in our experiments, the weights in each layer are linearly quantized to q bins where q is a pre-defined number, all the weights in the same bin share the same value, thus for each weight, we then need to store only a small index into a table of shared weights. The size of each bin is calculated as

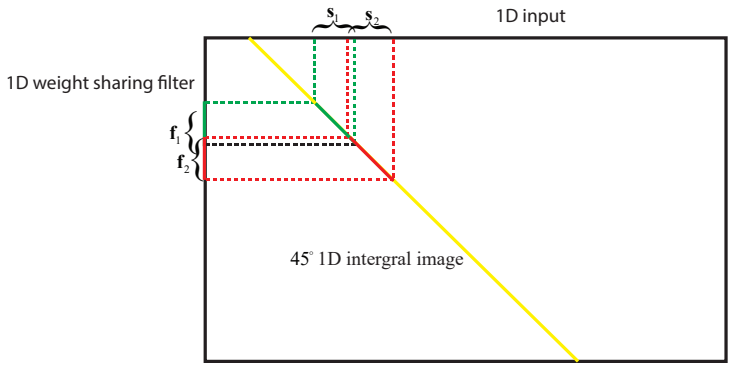

Figure 2: Illustration of Efficient 1D Convolution by 1D Integral Image.

$(\max(W) - \min(W))/q$. To calculate the compression rate by weight quantization, given q bins, we only need $log_2(q)$ bits to encode the index. For the weight sharing filter in WSNet which uses 4 bytes float number (32 bits) to represent a weight, the compression rate of weight quantization can be calculated as $\frac{32\ell^* m^*}{\ell^* m^* log_2(q)+32q}$. Since the condition of $\ell^* m^* \gg q$ generally holds in most layers of WSNet, the compression rate by weight quantization is approximately $\frac{32}{log_2(q)}$. Note different from previous literatures Han et al. (2015) which learns the weight quantization during training, we perform quantization after the training of WSNet is finished. In the experiments, we find that it is sufficient to use our simpler weight quantization method to obtain a significantly compressed WSNet without loss of accuracy.

3.5 Integral Image for Efficient 1D Convolution

The weight sharing scheme makes an accelerated algorithm using 1D integral image for the convolution operation between the 1D weight sharing filter and the input, which is illustrated in Figure 2. We first compute the the outer product of the 1D weight sharing filter and the 1D input, and compute the 1D integral image along all the crossing lines of 45 degree. In the example showed in Figure 2, the convolution result of filter $\mathbf{f}_1$ and the input signal $\mathbf{s}_1$ are the sum of elements in the green segment on the yellow crossing line of 45 degree, and convolution result of filter $\mathbf{f}_2$ and the input signal $\mathbf{s}_2$ are the sum of elements in the red segment on the same yellow line. With the 1D integral image computed for this yellow line, these two convolution results can be computed efficiently by two subtraction operations.

4 Experiments

In this section, we present the details and analysis of the results in our experiments. Extensive ablation studies are conducted to verify the effectiveness of the proposed WSNet on learning compact and efficient networks. On all tested datasets, WSNet is able to improve over baseline networks on classification accuracy while achieves **100x** compression ratio. When compressing up to (*e.g.* **180x**), there is only slight performance drop observed compared to baseline networks.

4.1 Experimental Settings

Datasets We collect a large-scale music detection dataset (MusicDet200K) from public available social media platforms (*e.g* Facebook, Twitter, *etc.*) to conduct experiments with. For fair comparison with previous literatures, we also test WSNet on three standard, publicly available datasets, *i.e* ESC-50, UrbanSound8K and DCASE, the details of which are as follows.

MusicDet200K aims to assign a binary label to each sample to indicate whether it is music or not. MusicDet200K has overall 220,000 annotated sound clips. Each has a time duration of 4 seconds and is resampled to 16000 Hz and normalized. Among all samples, we use 200,000/20,000/18,000

Table 1: Baseline-1: the configuration of the baseline network used on MusicDet200K. After each convolutional layer, we use the nonlinearity layer (*i.e.* ReLU), batch normalization layer and pooling layer which are omitted in the table for brevity. The strides of all pooling layers are 2. The padding strategies adopted for both convolutional layers and fully connected layers are all "size preserving".

Layer	conv1	conv2	conv3	conv4	conv5	conv6	conv7	fc1	fc2
Filter sizes	32	32	16	8	8	8	4	1536	256
#Filters	32	64	128	128	256	512	512	256	128
Stride	2	2	2	2	2	2	2	1	1
#Params	1K	65K	130K	130K	260K	1M	1M	390K	33K
Params(%)	0.03%	2.1%	4.2%	4.2%	8.4%	33.7%	33.7%	12.6%	1.1%

as train/val/test set. The samples belonging to "non-music" count for around 70% of all samples, which means if we trivially assign all samples to be "non-music", the classification accuracy is 70%.

ESC-50 is a collection of 2000 short (5 seconds) environmental recordings comprising 50 equally balanced classes of sound events in 5 major groups (animals, natural soundscapes and water sounds, human non-speech sounds, interior/domestic sounds and exterior/urban noises) divided into 5 folds for comparable cross-validation. Following Aytar et al. (2016), we extract 10 sound clips from each recording with length of 1 second and time step of 0.5s (*i.e.* two neighboring clips have 0.5 seconds of themselves overlapped). Therefore, in each cross-validation, the number of training samples is 16000. In testing, we average over ten clips of each recording for the final classification results.

UrbanSound8K is a collection of 8732 short (around 4 seconds) recordings of various urban sound sources (*air conditioner*, *car horn*, *playing children*, *dog bark*, *drilling*, *engine idling*, *gun shot*, *jackhammer*, *siren*, *street music*). Like in ESC-50, we extract 8 clips with time length of 1 second from each recording. For those that are less than 1 seconds, we pad them with zeros and repeat for 8 times.

DCASE is used in the Detection and Classification of Acoustic Scenes and Events Challenge (DCASE). It contains 10 acoustic scene categories, 10 training examples per category and 100 testing examples. Each sample is a 30 seconds audio recording. During training, we evenly extract 12 sound clips from each recording.

Evaluation criteria To demonstrate that WSNet is capable of effectively compressing CNN at small cost of accuracy losses, two evaluation criteria are used in our experiments: compression ratio and sound classification accuracy.

Baseline networks The network we used on MusicDet200K consists of 7 convolutional layers and 2 fully connected layers, using which we verify the capability of WSNet on effectively learning compact representations for both convolutional layers and fully connected layers. For fair comparison with others literatures, we firstly modify the state-of-the-art SoundNet Aytar et al. (2016) by applying pooling layers to all but the last convolutional layer. As can be seen in Table 5, this modification largely boost the performance of original SoundNet. We apply this modified SoundNet on all three public datasets as the baseline network. The architectures of those two baseline networks are shown in Table 1 and Table 2, respectively.

Implementation details WS-Net is implemented in Tensorflow Abadi et al. (2016). Following Aytar et al. (2016), the Adam Kingma & Ba (2014) optimizer and a fixed learning rate of 0.001 and momentum term of 0.9 and batch size of 64 are used throughout experiments. We initialized all the weights to zero mean Gaussian noise with a standard deviation of 0.01. In WSNet_1, the dropout ratio for the dropout layers Srivastava et al. (2014) after each fully connected layer is set to be 0.8. The overall training is carried on 100,000 iterations.

Table 2: Baseline-2: the configuration of the layers of the baseline network used on ESC-50, UrbanSound8K and DCASE. For brevity, the nonlinearity layer (*i.e.* ReLU), batch normalization layer and pooling layer following each convolutional layer are omitted. The kernel sizes for each pooling layer following conv1-conv4 is 8 and 4 for the rest pooling layers. The strides of all pooling layers are 2.

Layer	conv1	conv2	conv3	conv4	conv5	conv6	conv7	conv8
Filter sizes	64	32	16	8	4	4	4	8
#Filters	16	32	64	128	256	512	1024	1401
Stride	2	2	2	2	2	2	2	2
#Params	1K	16K	32K	65K	130K	520K	2M	11M
Params(%)	0.01%	0.11%	0.22%	0.45%	0.90%	3.63%	14.55%	79.61%

4.2 Results and analysis

4.2.1 MusicDet200K

Ablation analysis Through extensive control experiments, we investigate the effects of each component in WSNet on the networks' compression ratios and classification accuracies. The comparative study results of different settings of WSNet are listed in Table 3.

(1) Weight sampling We test the performance of WSNet when using compression ratios of 2, 4 and 8 respectively. As can be seen from Table 3, S_2 and S_4 slightly outperforms the classification accuracy of the baseline model, possibly by reducing the overfitting. When compressing 8 times, the classification accuracy of S_8 only drops by 0.4%. Note that the maximum compression ratio along the spatial dimension is equal to the filter size, thus for the layer "conv7" which has filter size of 4, its compression ratio is still 4 in S_8. Above results demonstrate that benefiting from the weight-sharing strategy used by WSNet can achieve comparative performance by compressing the network in the spatial dimension.

(2) Channel sharing Three compression ratios: 2, 4 and 8 are used to compress the network along the channel dimension. It can be observed from Table 3 that C_2 and C_4 and C_8 achieves linear compression ratios while incurring negligible loss of accuracy. Actually, C_2 can even boost the accuracy upon baseline models, demonstrating the high efficiency of sharing channels in WSNet. In the next, we compress in both the spatial dimension and channel dimension (*i.e.* $S_4C_4SC_4^\dagger$ and $S_8C_8SC_8^\dagger$), it reveals that WSNet is capable of obtaining much higher compression ratios while leading to small performance decrease (less than 1%).

(3) Filter augmentation Filter augmentation is used to enhance the learning capability of WSNet in the case of aggressive compression (*i.e.* S and C are large) and make up the performance loss caused by compression. As shown in Table 3, after extracting 2 times more filters in conv1, conv2 and conv3, $S_8C_8SC_8^\dagger A_2$ significantly outperforms the $S_8C_8SC_8^\dagger$. Above results demonstrate that effectiveness of filter augmentation to boost the performance of WSNet.

(4) Weight quantization Weight quantization is used to further compress WSNet. It can be observed from Table 3 that after using $q = 256$ quantization bins to represent each weight by one byte (*i.e.* 8bits), $S_8C_8SC_{15}^\dagger A_2Q_4$) is able to compress the baseline network at 168$\times$ while incurring only 0.1% accuracy loss. Above result demonstrates that the weight quantization method is complementary with WSNet on jointly compressing deep networks.

Visualization To show that WSNets learn more compact parameters compared to baselines by using weight sampling and channel sharing, we visualize the learned filters in conv1 (has shape of $(32, 1, 32)$) from the baseline model and WSNet (*i.e.* $S_4C_4SC_4^\dagger$ in Table 3) in Figure 3. For easy comparison, we firstly reshape the filters in the baseline model to have the same length of the learned weight sharing filters in WSNet. More concretely: the filters in the baseline model are assigned to different segments of equal length by a stride of 4, and the overlapping weights are averaged. As evidenced in Figure 3, the learned weight sharing filters in WSNet demonstrate wider spectrum between yellow and dark. This could be explained as followed: since the filters in WSNet has to be

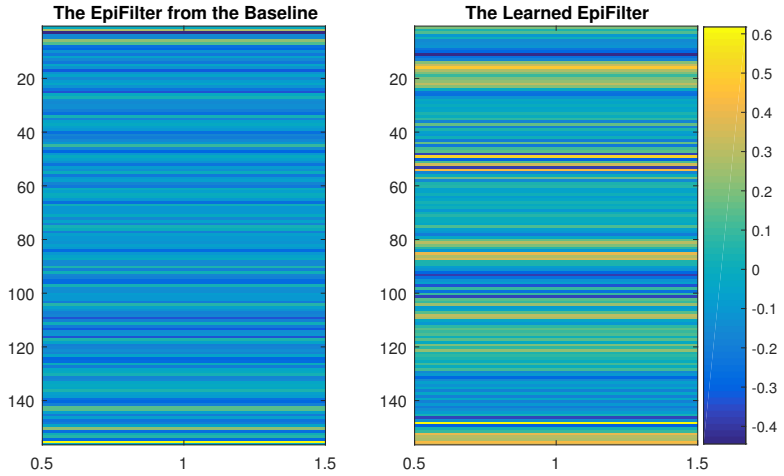

Figure 3: Visualization of the learned filters in conv1 layer from the baseline model and WSNet, all of which are training on MusicDet200K. **Left**: 32 1D filters of size 32 in conv1 of the baseline network are aggregated to form a filter of size 156. The right: the learned filters in WSNet with filter size of 156. Please refer to Section 4.2 for detailed analysis.

Table 3: Comparative study of effects of different settings of WSNet on the compression ratio and classification accuracy on MusicDet200K. For clear description, we represent the names of WSNets with different settings by the combination of symbols S/C/SC†/Q. "S" denotes weight sampling; "C" denotes channel sharing. "A" denotes filter augmentation. "SC†" denotes the overall compression ratio of fully connected layers. "Q" denotes weight quantization. With a symbol occurred in the name, the corresponding component is used in WSNet. The numbers in subscripts of S/C/A/SC/Q denotes the maximum compression ratio on spatial/channel dimension in all layers, the increased times of filters, the compression ratio of fully connected layers and the compression ratio of weight quantization, respectively.

WSNet's settings	conv{1-3}			conv4			conv5			conv6			conv7			fc1	fc2	Acc.	Comp. ratio
	S	C	A	S	C	A	S	C	A	S	C	A	S	C	A	SC†	SC†		
Baseline	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	88.9	3M (1×)
S_2	2	1	1	2	1	1	2	1	1	2	1	1	2	1	1	2	2	88.8	2×
S_4	4	1	1	4	1	1	4	1	1	4	1	1	4	1	1	4	4	88.8	4×
S_8	8	1	1	8	1	1	8	1	1	8	1	1	4	1	1	8	8	88.3	5.7×
C_2	1	2	1	1	2	1	1	2	1	1	2	1	1	2	1	2	2	88.8	2×
C_4	1	4	1	1	4	1	1	4	1	1	4	1	1	4	1	4	4	88.7	4×
C_8	1	8	1	1	8	1	1	8	1	1	8	1	1	8	1	8	8	88.6	8×
$S_4C_4SC_4^\dagger$	4	4	1	4	4	1	4	4	1	4	4	1	4	4	1	4	4	88.7	11.1×
$S_8C_8SC_8^\dagger$	8	8	1	8	8	1	8	8	1	8	8	1	4	8	1	8	8	88.4	23×
$S_8C_8SC_8^\dagger A_2$	8	8	2	8	8	1	8	8	1	8	8	1	4	8	1	8	8	89.2	20×
$S_8C_8SC_{15}^\dagger A_2$	8	8	2	8	8	1	8	8	1	8	8	1	8	8	1	15	15	88.6	42×
$S_8C_8SC_{15}^\dagger A_2Q_4$	8	8	2	8	8	1	8	8	1	8	8	1	8	8	1	15	15	88.5	168×

Table 4: The configurations of the WSNet used on ESC-50, UrbanSound8K and DCASE. Please refer the denotation of symbols in the "Settings" column to Table 3.

WSNet's settings	conv{1-4}			conv5			conv6			conv7			conv8			Comp. ratio
	S	C	A	S	C	A	S	C	A	S	C	A	S	C	A	
$S_8C_4A_2$	4	4	2	4	4	1	4	4	1	4	4	1	8	4	1	25×
$S_8C_8A_2$	4	4	2	4	4	1	4	4	1	4	8	1	8	8	1	45×

Table 5: Comparison with state-of-the-arts on ESC-50. All results of WSNet are obtained by 10-folder validation. Please refer the denotation of symbols in the "Settings" column to Table 3. The baseline used here is a simple modification of SoundNet with 8 convolution layers (refer to Section 4.1 for details), thus they have the same parameter size.

Model	Settings	Acc. (%)	Comp. ratio
baseline	-	66.0	13.7M (1×)
WSNet	$S_8C_4A_2$	66.5	25×
WSNet	$S_8C_4A_2Q_4$	**66.25**	**100**×
WSNet	$S_8C_8A_2$	66.1	45×
WSNet	$S_8C_8A_2Q_4$	**65.8**	**180**×
Piczak ConvNet	-	64.5	-
SoundNet (Aytar et al., 2016)	8 layer, Scratch Init.	51.1	1×
SoundNet (Aytar et al., 2016)	8 layer, Unlabeled Video	72.9	1×

Table 6: Comparison with state-of-the-arts on UrbanSound8K. All results of WSNet are obtained by 5-folder validation. Please refer the denotation of symbols in the "Settings" column to Table 3.

Model	Settings	Acc. (%)	Comp. ratio
baseline	-	70.39	13.7M (1×)
WSNet	$S_8C_4A_2$	70.76	25×
WSNet	$S_8C_4A_2Q_4$	**70.61**	**100**×
WSNet	$S_8C_8A_2$	70.02	45×
WSNet	$S_8C_8A_2Q_4$	**69.79**	**180**×
Piczak ConvNet (Piczak, 2015b)	-	73.1	-
Salamon & Bello (Salamon & Bello, 2015)	-	73.6	1×

learned in a much smaller parameter space, each weight is encouraged to capture different patterns in the input to be compact, therefore having the larger variations in filters from WSNet than that from the baseline network.

4.2.2 ESC-50

The comparison of WSNet with other state-of-the-arts on ESC-50 is listed in Table 5. The settings of WSNet used on ESC-50, UrbanSound8K and DCASE are listed in Table 4. Compared with the baseline model, WSNet is able to significantly reduce the model size by 25 times and 45 times, while at the same time improve the accuracy of baseline model by 0.5% and 0.1% respectively. Such promising results again demonstrates the effectiveness of WSNet on compressing networks. After applying weight quantization to WSNet, the compression ratio reaches 180 while the accuracy only slightly drops by 0.2%. Benefited from a transfer learning approach, the SoundNet Aytar et al. (2016) that is trained using a large number of unlabeled videos achieves better accuracy than our network. However, since the training method is orthogonal to WSNet, it is expectable that WSNet can also achieve higher accuracies by training similarly as SoundNet Aytar et al. (2016) on unlabeled videos.

4.2.3 URBANSOUND8K

We report the comparsion results of WSNet with state-of-the-arts in Table 6. It can be again observed that WSNet significantly reduces the model size of baseline while obtaining comparative results. Both Piczak (2015b) and Salamon & Bello (2015) uses log-scaled mel-spectrograms extracted from recordings during training. In comparison, the proposed WSNet simply takes the raw wave of recordings as input, saving the time of extracting complex features as done in Piczak (2015b); Salamon & Bello (2015).

Table 7: Comparison with state-of-the-arts on DCASE. Note there are only 100 samples in testing set. Please refer the denotation of symbols in the "Settings" column to Table 3.

Model	Settings	Acc. (%)	Comp. ratio
baseline	-	85	1×
WSNet	$S_8C_4A_2$	86	25×
WSNet	$S_8C_4A_2Q_4$	**86**	**100**×
WSNet	$S_8C_8A_2$	84	45×
WSNet	$S_8C_8A_2Q_4$	**84**	**180**×
Piczak ConvNet (Piczak, 2015b)	-	73.1	-
Salamon & Bello (Salamon & Bello, 2015)	-	73.6	1×

4.2.4 DCASE

As one can observe in Table 7, WSNet outperforms the baseline model by 1% with only 1% size of the baseline model. When compressing at a more aggressive ratio, *i.e.* 180×, the classification accuracy of WSNet is only one percentage lower than the baseline model, demonstrating the effectiveness of the proposed weight sampling strategy in WSNet.

5 Conclusion

In this paper, we present a **W**eight **S**ampling networks (WSNet) to learn efficient and compact networks. There are two novel components in WSNet: weight sampling and channel sharing, through which the weights in each layer is compressed in the spatial dimension and channel dimension respectively. In weight sampling, all filters taking part in convolution are sampled from the learned weight sharing filters. In channel sharing, channels in a filter are shared through repeating. Extensive experiments on four audio classification datasets including MusicDet200K, ESC-50, UrbanSound8K and DCASE demonstrates the effectiveness of WSNet. It is demonstrated that WSNet is able to achieve a compression ratio as high as 180 without significant performance drop.

References

Martín Abadi, Ashish Agarwal, Paul Barham, Eugene Brevdo, Zhifeng Chen, Craig Citro, Greg S Corrado, Andy Davis, Jeffrey Dean, Matthieu Devin, et al. Tensorflow: Large-scale machine learning on heterogeneous distributed systems. *arXiv preprint arXiv:1603.04467*, 2016.

Sajid Anwar, Kyuyeon Hwang, and Wonyong Sung. Structured pruning of deep convolutional neural networks. *J. Emerg. Technol. Comput. Syst.*, 13(3):32:1–32:18, February 2017. ISSN 1550-4832. doi: 10.1145/3005348. URL `http://doi.acm.org/10.1145/3005348`.

Yusuf Aytar, Carl Vondrick, and Antonio Torralba. Soundnet: Learning sound representations from unlabeled video. *NIPS 2016*.

Yusuf Aytar, Carl Vondrick, and Antonio Torralba. Soundnet: Learning sound representations from unlabeled video. In *Advances in Neural Information Processing Systems*, pp. 892–900, 2016.

D. Barchiesi, D. Giannoulis, D. Stowell, and M.D. Plumbley. Acoustic scene classification: Classifying environments from the sounds they produce. *IEEE Signal Processing Magazine*, 32(3): 16–34, 2015.

R. Cai, L. Lu, A. Hanjalic, H.J. Zhang, and Cai L.H. A flexible framework for key audio effects detection and auditory context inference. *IEEE Transactions on audio, speech, and language processing*, 14(3):1026–1039, 2006.

S. Chu, S. Narayanan, C.C. Kuo, and M.J. Mataric. Where am i? scene recognition for mobile robots using audio features. *IEEE International Conference on Multimedia and Expo*, pp. 885–888, 2006.

Maxwell D Collins and Pushmeet Kohli. Memory bounded deep convolutional networks. *arXiv preprint arXiv:1412.1442*, 2014.

S.B. Davis and P. Mermelstein. Comparison of parametric representation for monosyllabic word recognition in continuously spoken sentences. *IEEE Trans. ASSP*, Aug. 1980.

Emily L Denton, Wojciech Zaremba, Joan Bruna, Yann LeCun, and Rob Fergus. Exploiting linear structure within convolutional networks for efficient evaluation. In *NIPS*, pp. 1269–1277, 2014.

J.L. Flanagan. Speech analysis, synthesis and perception. *Springer- Verlag*, 1972.

Yunchao Gong, Liu Liu, Ming Yang, and Lubomir Bourdev. Compressing deep convolutional networks using vector quantization. *arXiv preprint arXiv:1412.6115*, 2014.

Song Han, Jeff Pool, John Tran, and William Dally. Learning both weights and connections for efficient neural network. In *NIPS*, pp. 1135–1143, 2015.

Song Han, Huizi Mao, and William J. Dally. Deep compression: Compressing deep neural network with pruning, trained quantization and huffman coding. In *ICLR*, 2016.

Babak Hassibi and David G Stork. *Second order derivatives for network pruning: Optimal brain surgeon*. Morgan Kaufmann, 1993.

Forrest N Iandola, Song Han, Matthew W Moskewicz, Khalid Ashraf, William J Dally, and Kurt Keutzer. Squeezenet: Alexnet-level accuracy with 50x fewer parameters and¡ 0.5 mb model size. *arXiv preprint arXiv:1602.07360*, 2016.

Salamon J., C. Jacoby, and J. P. Bello. *ACM Multimedia*, 2014.

Xiaojie Jin, Xiaotong Yuan, Jiashi Feng, and Shuicheng Yan. Training skinny deep neural networks with iterative hard thresholding methods. *arXiv preprint arXiv:1607.05423*, 2016.

Yong-Deok Kim, Eunhyeok Park, Sungjoo Yoo, Taelim Choi, Lu Yang, and Dongjun Shin. Compression of deep convolutional neural networks for fast and low power mobile applications. *arXiv preprint arXiv:1511.06530*, 2015.

Diederik Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

C. Landone, J. Harrop, and J. Reiss. Enabling access to sound archives through integration, enrichment and retrieval: the easaier project. *ISMIR*, pp. 159–160, 2007.

Vadim Lebedev and Victor Lempitsky. Fast convnets using group-wise brain damage. *arXiv preprint arXiv:1506.02515*, 2015.

Yann LeCun, John S Denker, Sara A Solla, Richard E Howard, and Lawrence D Jackel. Optimal brain damage. In *NIPS*, volume 89, 1989.

Hao Li, Asim Kadav, Igor Durdanovic, Hanan Samet, and Hans Peter Graf. Pruning filters for efficient convnets. In *ICLR*, 2017.

Jian-Hao Luo, Jianxin Wu, and Weiyao Lin. Thinet: A filter level pruning method for deep neural network compression. In *ICCV*, 2017.

Karol J Piczak. Esc: Dataset for environmental sound classification. *ACM Multimedia*, 2015a.

Karol J Piczak. Environmental sound classification with convolutional neural networks. In *Machine Learning for Signal Processing (MLSP), 2015 IEEE 25th International Workshop on*, pp. 1–6. IEEE, 2015b.

L.C.W. Pols. Spectral analysis and identification of dutch vowels in monosyllabic words. *dissertation*, 1966.

J. Salamon and J.P. Bello. Deep convolutional neural networks and data augmentation for environmental sound classification. *IEEE Signal Processing Letters*, 24(3):279–283, 2017.

Justin Salamon and Juan Pablo Bello. Unsupervised feature learning for urban sound classification. In *Acoustics, Speech and Signal Processing (ICASSP), 2015 IEEE International Conference on*, pp. 171–175. IEEE, 2015.

B. Schilit, N. Adams, and R. Want. Context-aware computing applications. *IEEE Mobile Computing Systems and Applications*, pp. 85–90, 1994.

Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. In *ICLR*, 2015.

Vikas Sindhwani, Tara Sainath, and Sanjiv Kumar. Structured transforms for small-footprint deep learning. In *Advances in Neural Information Processing Systems*, pp. 3088–3096, 2015.

Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Dropout: A simple way to prevent neural networks from overfitting. *JMLR*, 15(1):1929–1958, 2014.

Dan Stowell, Dimitrios Giannoulis, Emmanuouil Benetos, Mathieu Lagrange, and Mark D. Plumbley. Detection and classification of acoustic scenes and events. *TM*, 2015.

Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. Going deeper with convolutions. In *CVPR*, 2015.

M. Valenti, A. Diment, G. Parascandolo, S. Squartini, and T. Virtanen. Dcase2016 acoustic scene classification using convolutional neural networks. *Workshop of Detection and classification of Acoustic Scenes Events*, pp. 95–99, 2016.

P. Viola and M. Jones. Robust real time object detection. *IJCV*, 2001.

Y. Xu, W.J. Li, and K.K. Lee. Intelligent wearable interfaces. *John Wiley & Sons*, 2008.