Channel Pruning for Accelerating Convolutional Neural Networks via Wasserstein Metric

Haoran Duan and Hui Li(✉)

Key Laboratory of Wireless-Optical Communications, Chinese Academy of Sciences
School of Information Science and Technology, University of Science and Technology of China, Hefei, China
hrduan@mail.ustc.edu.cn, mythlee@ustc.edu.cn

Abstract. Channel pruning is an effective way to accelerate deep convolutional neural networks. However, it is still a challenge to reduce the computational complexity while preserving the performance of deep models. In this paper, we propose a novel channel pruning method via the Wasserstein metric. First, the output features of a channel are aggregated through the Wasserstein barycenter, which is called the basic response of the channel. Then the *channel discrepancy* based on the Wasserstein distance is introduced to measure channel importance, by considering both the channel's feature representation ability and the substitutability of the basic responses. Finally, channels with the least discrepancies are removed directly, and the loss in accuracy of the pruned model is regained by fine-tuning. Extensive experiments on popular benchmarks and various network architectures demonstrate that the proposed approach outperforms the existing methods.

Keywords: Channel pruning; Wasserstein metric; CNNs

1 Introduction

In recent years, convolutional neural networks (CNNs) have made great achievements in various computer vision tasks [1,2,3,4]. However, the superior performance of CNNs relies on its huge computation and memory cost, which makes these CNNs very difficult to deploy on the devices with limited resources, such as mobile phones or embedded gadgets, etc. To expand the application scope of the CNNs, the research on model compression has attracted great interest.

Recent developments in model compression can be divided into three categories, namely, quantization, low-rank approximation, and network pruning. Network quantization compresses the original network by using few bits to represent each weight. However, the quantization error leads to low training efficiency of the network. Low-rank approximation aims to compress and accelerate the network through tensor decomposition, but these methods often suffer from expensive fine-tuning processes and performance degradation. Network pruning has two branches, *i.e.*, weight pruning and filter/channel pruning. Weight

pruning simply removes weights parameters in filters, which may lead to non-structured sparsity. Thus the specific hardware structure is required for efficient inference. In contrast, channel pruning can significantly reduce memory footprint and boost the inference speeds by directly discarding redundant channels. Moreover, channel pruning is hardware-friendly and it can be easily integrated with other model compression techniques.

In recent work [5], the greedy algorithm is used to remove redundant channels by minimizing the reconstruction error before and after pruning. Regularization-based methods [6,7,8,9] introduce the sparsity penalty to the training objective. For these methods, more training epochs and extra optimization are required to achieve an optimal pruning result, which leads to low efficiency for large networks and datasets. In [10] channel pruning is implemented based on feature reconstruction, which leads to the problem of redundant feature maps due to the absence of the analysis on the relationship among the channels. Wang *et al.* [11] explore the linear relationship among the channels through clustering feature maps. However, the clustering results are only obtained by specific inputs, which do not apply to various input categories. LFC *et al.* [12] prunes one filter of the filter pairs with the largest correlation. LFC only considers the linear relationship of filters while ignoring the nonlinear relationships of filters. Moreover, the extra optimization is needed to increase correlations. To address the above issues, we propose a novel pruning method via the Wasserstein metric.

The concept of the Wasserstein metric or distance derives from optimal transport theory [13], which provides many powerful tools for probability measures and distributions. The Wasserstein distance measures the similarity or discrepancy between distributions by computing the minimal cost of transporting all the mass contained in one distribution to another. The Wasserstein distance satisfies all metric axioms and has many favorable properties, and the details are referred to [13].

In this paper, we aim to identify the most replaceable channels by summarizing the output features of channels, which can reflect the channels' contributions to the layer. Specifically, to prune the most redundant channels in a pre-trained model, the Wasserstein barycenter [14,15] is first utilized to aggregate the output features of a channel. The Wasserstein barycenter is defined as the basic response of the channel, which represents the unique output characteristics of the channel and is independent of the specific input category. This property allows us to identify redundant channels by analyzing the discrepancy between the basic responses. After obtaining the basic response of channels, we further propose *channel discrepancy* to measure the differences between channels and the channel's feature representation ability. Channels with minimal *channel discrepancy* can be substituted by other channels. By directly identifying and removing the channels that contribute the least to each layer, the proposed method avoids the tedious feature reconstruction process. After pruning all redundant channels, the pruned model is fine-tuned to recover performance. Given a pre-trained model, the proposed method can construct a compact model with comparable or even higher performance.

The main contributions of this work can be summarized as follows:

- A novel channel pruning approach via the Wasserstein metric is proposed. The *channel discrepancy* is introduced to measure the contribution of the channel, which provides a new perspective for understanding channel redundancy. And the non-iterative pruning manner makes the proposed method more efficient than previous methods.
- To our knowledge, this is the first work to introduce the Wasserstein metric for pruning deep convolutional neural networks. The Wasserstein metric can be regarded as a new indicator of channel importance, which expands the current evaluation criteria.
- Extensive experiments on popular datasets and various network architectures demonstrate that the proposed pruning approach outperforms the previous methods. On CIFAR-10 [1], the proposed approach achieves about 60% FLOPs reduction of ResNet-56 and ResNet-110 with an accuracy gain of 0.02% and 0.27% respectively. On ImageNet [16], the pruned ResNet-101 achieves a 58.8% FLOPs reduction with a loss of 0.64% in the top-1 accuracy.

2 Related Work

Low-rank approximation. Denton *et al.* [17] introduce tensor decompositions based on SVD to approximate the weight matrix in CNNs. Jaderberg *et al.* [18] achieve a 4.5× speedup with little performance loss by approximating filter banks via low-rank filters. Lebedev *et al.* [19] utilize the Canonical Polyadic decomposition on the kernel tensors, which can be efficiently computed by existing algorithms.

Network quantization. Chen *et al.* [20] propose to hash the network weights into different groups and the weight value is shared in each group. In [21], networks are quantized with binary weights which significantly reduces memory usage. Rastegari *et al.* [22] propose XNOR-Net to approximate convolutions with bitwise operations. Compared to standard networks, the proposed XNOR-Net can achieve a similar performance but more efficient. Zhao *et al.* [23] introduce outlier channel splitting to improve network quantization without retraining. To enhance the representational capability, Liu *et al.* [24] use a identity mapping to propagate the real-valued information before binarization.

Network pruning. Recent work on network pruning can be categorized into two sub-families: weight pruning and channel pruning. In early weight pruning work, LeCun *et al.* [25] and Hassibi *et al.* [26] propose to remove redundant weights based on second-order derivatives of the loss function. Han *et al.* [27] evaluate the importance of weights based on the magnitude and remove unimportant weights with small absolute values. Guo *et al.* [28] integrate pruning and splicing operations, by which incorrect pruned connections could be recovered

if they are found to be important. However, weight pruning can cause irregular connections, which requires specialized hardware to accelerate. Thus channel pruning is more preferred than weight pruning. Channel pruning directly discards unimportant channels without affecting network structure. Li *et al.* [29] utilize ℓ_1-norm to measure the importance of each filter. Based on Lasso regression, He *et al.* [10] and Luo *et al.* [5] prune networks in a layer-by-layer manner by selecting the filters that minimizing the reconstruction error. He *et al.* [30] propose a soft manner pruning approach to preserve the model capacity. In [9], the saliency of each filter is globally evaluated and dynamically updated. Ye *et al.* [8] apply scaling factors to each channel and add the sparse penalty to the training objective. He *et al.* [31] introduce the Geometric Median to prune the fewer contribution filters in the network.

3 Preliminaries

In this section, we give a review of the key concepts used in our proposed method.

3.1 Wasserstein Distance

The p-Wasserstein distance [13], or the Monge-Kantorovich distance of order p, quantifying the discrepancy between two distributions μ_0 and μ_1 is defined as

$$\mathbf{W}_p\left(\mu_0, \mu_1\right) = \left(\inf_{\pi \in \Pi(\mu_0, \mu_1)} \int_{M \times M} d\left(x, y\right)^p \mathrm{d}\pi\left(x, y\right)\right)^{\frac{1}{p}}, \tag{1}$$

where M denotes the compact embedding space and $d : M \times M \to \mathbb{R}_+$ denotes the geodesic distance function. $\Pi(\mu_0, \mu_1)$ is the set of joint distributions of (μ_0, μ_1) having μ_0 and μ_1 as marginals.

The Wasserstein distance satisfies all metric axioms. It seeks a transport plan $\pi \in \Pi(\mu_0, \mu_1)$ with the optimal cost of transporting distribution μ_0 to μ_1, where the cost of moving a unit of mass from x to y is

$$d\left(x, y\right)^p = \|x - y\|_p^p \tag{2}$$

the cost $d\left(x, y\right)$ equals ℓ_1-norm, and $d\left(x, y\right)^2$ is calculated by the squared Euclidean distance.

3.2 Wasserstein Barycenter

Given a set of distributions $\{\mu_1, \mu_2, \ldots, \mu_k\}$, the Wasserstein barycenter μ with respect to the Wasserstein metric, is defined as the following minimization problem:

$$\mu = arg\min_{\mu} \sum_{i=1}^{k} \alpha_i \mathbf{W}_2^2\left(\mu, \mu_i\right), \tag{3}$$

where $(\alpha_1, \alpha_2, \ldots, \alpha_k)$ are non-negative weights summing to 1, and $\mathbf{W}_2^2$ denotes the squared 2-Wasserstein distance.

The Wasserstein barycenter tries to summarize the collection of distributions, which can describe the geometry characteristics better than the Euclidean average.

3.3 Smoothed Wasserstein Distance

The computation of the normal version of the Wasserstein distance has high time complexity, which scaling super-cubically in the size of the domain. Adding an entropic regularization term to the original distance makes the problem strictly convex. Specifically, the squared 2-Wasserstein distance with an entropic regularization term is defined as

$$\mathbf{W}_{2,\gamma}^2(\mu_0, \mu_1) = \inf_{\pi \in \Pi(\mu_0, \mu_1)} \Big[\int_{M \times M} d(x, y)^2 \, \pi(x, y) \, \mathrm{d}x \, \mathrm{d}y - \gamma \mathbf{H}(\pi) \Big], \quad (4)$$

where $\mathbf{H}(\pi)$ denotes the differential entropy of π on $M \times M$ and $\gamma > 0$ is the regularization parameter.

The smoothed version of the Wasserstein distance can be solved iteratively by Sinkhorn-Knopp algorithm [32], with a linear convergence rate. Another good feature of the smoothed Wasserstein distance is that the computation can be carried out simultaneously using elementary linear algebra operations, which could take advantage of parallel GPGPU architectures to get further acceleration.

According to (3) and (4) , the original barycenter problem can be modified as follows

$$\mu = arg \min_{\mu} \sum_{i=1}^{k} \alpha_i \mathbf{W}_{2,\gamma}^2(\mu, \mu_i). \quad (5)$$

There has been a lot of excellent work aiming to solve (5). Our proposed method employs the fast convolution [15], which enhances Sinkhorn's efficiency by avoiding explicit pairwise distance computation and via the pre-factored diffusion operator.

4 Method

4.1 Notions

Let N denote a part of the samples of the training set. Considering the l-th layer of an L-layer CNN model, let n^l $(l \in [1, 2, \ldots, L])$ denote the number of input channels, k denote the spatial size of filters and h^l/w^l denote the height/weight of the input feature maps. Given the input feature maps $\mathbf{X}^l \in \mathbb{R}^{N \times n^l \times h^l \times w^l}$, the output feature maps $\mathbf{Y}^l$(*i.e.*, $\mathbf{X}^{l+1} \in \mathbb{R}^{N \times n^{l+1} \times h^{l+1} \times w^{l+1}}$) can be computed by applying weights $\boldsymbol{\theta}^l \in \mathbb{R}^{n^{l+1} \times n^l \times k \times k}$.

For the i-th sample and the j-th channel of the l-th layer, the output feature map can be formulated as

$$\mathbf{Y}_{i,j}^{l} = \sigma\left(\boldsymbol{\theta}_{j}^{l} \otimes \mathbf{X}_{i}^{l} + \mathbf{b}^{l}\right), \tag{6}$$

where $\sigma(\cdot)$ denotes nonlinear activation functions and $\otimes$ denotes the convolutional operation. In the following, the bias term $\mathbf{b}^{l}$ and the layer index l are omitted for simplicity.

4.2 The Basic Response of Channel

In the previous works [5,10], all feature maps are reconstructed to preserve the ability of the channel but ignore that the output features corresponding to the redundant channel and the important channel have different contributions to the layer output. Thus the less important feature maps are also recovered by mistake.

In this paper, without reconstructing the feature maps directly, we aim to summarize the output features by the Wasserstein barycenter, which is defined as the basic response of the channel. By aggregating the output feature maps, the basic response could represent the unique output characteristics of the channel, which is independent of the specific input category. Therefore, by analyzing the discrepancy between the basic responses, it is possible to identify channels that need to discard.

The j-th channel's output $\mathbf{Y}_{:,j} = \{\mathbf{Y}_{1,j}, \mathbf{Y}_{2,j}, \ldots, \mathbf{Y}_{N,j}\}$ is a collection of output features learned from N training samples. According to (5), the basic response, or the barycenter of the j-th channel is

$$\boldsymbol{\Lambda}_{j} = \arg\min_{\boldsymbol{\Lambda}} \sum_{i=1}^{N} \frac{1}{N} \mathbf{W}_{2}^{2}\left(\boldsymbol{\Lambda}, \mathbf{Y}_{i,j}\right), \tag{7}$$

where the basic response $\boldsymbol{\Lambda}_{j}$ summarizes the set of output distributions, and remain the same shape of output feature map. Then the basic responses of the l-th layer are $\{\boldsymbol{\Lambda}_{j}\}_{j=1}^{n^{l+1}}$. According to (7), the calculation of the basic responses does not require any training or reconstruction process.

4.3 Channel Discrepancy

We introduce the Wasserstein distance as a new indicator of channel importance, due to its superior ability to measure the discrepancy of distributions. Once obtained the basic responses of each channel, the discrepancy $d_{j,k}$ of the j-th and the k-th channel is defined as

$$d_{j,k} = \mathbf{W}_{2}^{2}\left(\boldsymbol{\Lambda}_{j}, \boldsymbol{\Lambda}_{k}\right). \tag{8}$$

$d_{j,k}$ measures the difference between the basic responses of the two channels. The larger the value of discrepancy, the greater the difference between the output of

Algorithm 1 Algorithm of channel pruning

Input: Training data $\mathcal{D}$, pre-trained model $\mathcal{M}$ with weights $\{\boldsymbol{\theta}^l\}_{l=1}^L$, pruning rate P_l, number of training samples N to compute the basic response.
Output: The pruned model with selected channels.

```
Forward model and get the output feature maps {Y^l}_{l=1}^L.
for layer l in model M do
    for channel in layer l do
        Compute basic response of channel via (7).
    end for
    Compute channel discrepancy via (11).
    Remove P_l-rate channels with smallest channel discrepancy.
end for
Fine-Tune the pruned model.
```

the two channels. The *layer-discrepancy* LD_j of the j-th channel is the average of the discrepancy between one channel and other channels, *i.e.*,

$$LD_j = \frac{1}{n^{l+1}-1} \sum_{k=1,j\neq k}^{n^{l+1}} d_{j,k} \tag{9}$$

where n^{l+1} is the number of channels in the l-th layer. The channel with high *layer-discrepancy* means that its output feature is unique and irreplaceable, therefore is important to the layer.

The redundancy of a channel depend on not only the *layer-discrepancy* but also its feature representation ability. An important channel is very active for different inputs. The more active the channel is, the richer the output feature maps. Such ability is called the *output-discrepancy*. According to (7) the *output-discrepancy* OD_j can be computed by

$$OD_j = \sum_{i=1}^{N} \frac{1}{N} \mathbf{W}_2^2 \left(\boldsymbol{\Lambda_j}, \mathbf{Y}_{i,j}\right), \tag{10}$$

where $\boldsymbol{\Lambda_j}$ represents the basic response and $\mathbf{Y}_{i,j}$ denotes the output feature maps of the j-th channel. The *output-discrepancy* measures the feature representation ability of the channel. The less redundant channel has large *output-discrepancy*.

For the j-th channel in the l-th layer, by combining both *layer-discrepancy* and *output-discrepancy*, we have a joint discrepancy formula as follows

$$D_j = LD_j + \beta OD_j = \frac{1}{n^{l+1}-1} \sum_{k=1,j\neq k}^{n^{l+1}} d_{j,k} + \beta \sum_{i=1}^{N} \frac{1}{N} \mathbf{W}_2^2 \left(\boldsymbol{\Lambda_j}, \mathbf{Y}_{i,j}\right), \tag{11}$$

where the two terms are balanced by β. D_j measures the channel's contribution to the entire layer and is called the *channel discrepancy*.

The *channel discrepancy* not only considers the mutual relationship between channels but also considers the channel's feature representation ability, therefore

Table 1. Comparison of pruning results for VGG-16 on CIFAR-10. "B. Top-k" and "P. Top-k" denote the top-k accuracy of the baseline and pruned model respectively. "Top-k↓(%)" means the top-k accuracy loss of pruned model compared to its baseline (smaller is better). "FLOPs↓(%)" denotes the reduction of FLOPs.

Model	Method	B. Top-1(%)	P. Top-1(%)	Top-1↓(%)	FLOPs↓(%)
VGG-16	L1	93.25	93.40	-0.15	34.2
	CP	**93.99**	93.67	0.32	50.0
	ThinNet	**93.99**	93.85	0.14	50.0
	Ours	93.73 ± 0.16	**93.99 ± 0.28**	**-0.26**	**50.4**

channels with the smallest *channel discrepancy* can be directly identified and safely pruned.

4.4 Pruning Algorithm

The proposed pruning algorithm can be described in the following steps to prune the redundant channels at a pruning rate P_l:

1. For each channel's output feature maps $\mathbf{Y}_{:,j}$ in $\mathbf{Y}$, calculate its corresponding basic response $\boldsymbol{\Lambda}_j$ through (7).
2. Obtain the *channel discrepancy* by using (11).
3. Remove P_l-rate channels in l-th layer with the minimal *channel discrepancy*, and the related channels in the next layer are removed in the meanwhile.

All the channels with less discrepancies will be directly identified and then pruned together. The performance of the pruned model can be quickly restored after fine-tuning. This non-iterative pruning manner makes our method more effective than the previous methods. Algorithm 1 details the procedures of our pruning technique.

5 Experiments

5.1 Experimental Settings

The proposed pruning algorithm is evaluated on CIFAR-10 [1] and ImageNet [16] with popular CNN networks, including VGGNet [2] and ResNet [3]. The CIFAR-10 dataset contains 50K training images and 10K validation images of 10 classes, and the ImageNet dataset consists of 1.28M training images and 50k validation images in 1000 classes. The CIFAR-10 experiments follow the same hyper-parameters setting in [3], training schedule and data argumentation in [33], and the "mean ± std" of accuracy is reported by running the experiment three times. For the ImageNet experiments, we follow the default settings in [3]. The training schedule is same as FPGM [31]. To balance the performance and efficiency, the number of the training samples, N is set to 256. The computations (elementwise arithmetic) of the Wasserstein distance and barycenter are accelerated by four NVIDIA GTX 1080 Ti GPUs.

Table 2. Pruning performance of Resnet-56 and Resnet-110 on CIFAR-10. For ResNet-56, our method achieves the highest performance with similar FLOPs reduction compared to previous methods. For ResNet-110, with 70% of the FLOPs reduced, the pruned model could still maintain almost the same performance as the baseline.

Model	Method	B. Top-1(%)	P. Top-1(%)	Top-1↓(%)	FLOPs↓(%)
ResNet-56	L1	93.04	93.06	-0.02	27.6
	CP	92.80	91.80	1.00	50.0
	ThinNet	**93.80**	92.98	0.82	49.8
	SFP	93.59	93.35	0.24	52.6
	AMC	92.80	91.90	0.90	50.0
	LFC	93.57	93.32	0.25	61.5
	FPGM	93.59	93.26	0.33	52.6
	Ours(40%)	93.28 ± 0.26	93.69 ± 0.24	-0.41	39.4
	Ours(50%)	93.28 ± 0.26	**93.71 ± 0.14**	**-0.43**	49.4
	Ours(60%)	93.28 ± 0.26	93.30 ± 0.34	-0.02	**59.4**
ResNet-110	L1	93.53	93.55	-0.02	15.9
	SFP	**93.68**	93.86	-0.18	40.8
	FPGM	**93.68**	93.74	-0.16	52.3
	Ours(40%)	93.59 ± 0.31	**94.22 ± 0.28**	**-0.63**	39.4
	Ours(50%)	93.59 ± 0.31	94.11 ± 0.43	-0.52	49.5
	Ours(60%)	93.59 ± 0.31	93.86 ± 0.37	-0.27	59.5
	Ours(70%)	93.59 ± 0.31	93.83 ± 0.42	-0.24	**69.9**

5.2 VGG-16 on CIFAR-10

VGGNet is widely used for vision tasks, which has a plain and straight forward network architecture. We select VGG-16 as a representative model. Following PFEC [29], the VGG-16 with Batch Normalization layer is used in the experiment, which is trained from scratch with 93.73% accuracy. When pruning, 30% convolution channels of each convolution layer in VGG-16 has been removed.

Table 1 shows the results compared with PFEC, CP [10] and ThinNet [5]. With 50.4% of FLOPs reduction, our pruning method could even gain a 0.26% improvement on accuracy.

5.3 ResNet on CIFAR-10

We also make experiments for Resnet-56 and ResNet-110 on CIFAR-10 dataset with different pruning rates. Table 2 shows the results.

For ResNet-56, we achieves similar FLOPs reduction (59.4% *vs.* 61.5%) compared to LFC [12], and maintains almost the same accuracy with respect to baseline model (−0.02%), while LFC drops 0.25%. Moreover, By setting the pruning rate to 50%, the proposed method achieves the best accuracy. With similar FLOPs reduction, our method preserves the highest performance by comparison with CP, ThinNet, SFP [30], AMC [34], and FPGM.

Table 3. Pruning results for Resnet-50 and Resnet-101 on ImageNet. Our proposed method achieves the best top-1 accuracy and top-5 accuracy compared with the existing methods.

Model	Method	B. Top-1(%)	P. Top-1(%)	Top-1↓(%)	Top-5↓(%)	FLOPs↓(%)
ResNet-50	ThinNet	72.88	71.01	1.87	1.12	**55.8**
	GAL	76.15	71.95	4.20	1.93	43.0
	SN	76.10	74.90	1.20	-	43.0
	SFP	76.15	62.14	14.01	8.27	41.8
	GDP	75.13	71.89	3.24	1.59	51.3
	LFC	75.30	73.40	1.90	0.80	50.4
	Taylor	**76.18**	74.50	1.68	-	44.9
	FPGM	76.15	74.83	1.32	0.55	53.5
	Ours(45%)	76.15	**75.27**	**0.88**	**0.37**	50.4
ResNet-101	BN-ISTA	76.40	74.56	1.84	-	52.7
	Taylor	**77.37**	75.95	1.42	-	**63.5**
	Ours(50%)	**77.37**	**76.73**	**0.64**	**0.39**	58.8

For ResNet-110, by setting the pruning rate to 40%, our method achieves 94.22% accuracy, which is 0.63% higher than the baseline. As more FLOPs are reduced, the improved performance starts to decrease, but still performs better than previous methods. Under a FLOPs reduction of 69.9%, the pruned model can still exceed the baseline by 0.24%.

It is worth noting that for pruning the pre-trained model of ResNet-56 or ResNet-110, our method outperforms the baseline under various pruning ratios. However, previous methods suffer performance loss under high FLOPs reduction. These results verify the ability of our algorithm to compress the model while maintaining or even achieving better performance.

5.4 ResNet on ImageNet

To further verify the effectiveness of the proposed pruning method, we test ResNet-50 and ResNet-101 as two variants of ResNet on ImageNet, specifically the ILSVRC2012 version. Following [5,30], the shortcut connections are maintained for simplicity. We report our results in Table 3.

Methods like GAL [35], SFP, and GDP [9] suffer a lot from FLOPs reduction of ResNet-50, while our method achieves the best results with the only 0.88% top-1 and 0.37% top-5 accuracy loss, which outperforms the previous channel pruning methods(ThinNet, SN [36], SFP, LFC, FPGM, Taylor [37]). The reason is that our method removes channels with the least contribution to the networks according to the *channel discrepancy*, thus the model could keep high performance after pruning. The pruned model of ResNet-101 maintains high top-1 accuracy (76.73%), and the top-5 accuracy only decreases by 0.39%, while the accuracy drops of BN-ISTA [8] and Taylor are much larger than our method.

Table 4. Pruning results by different measures. "A" with "B": compute barycenter by "A" and discrepancy by "B".

ResNet-56(50%)	Our method	WB with JS	EB with WD	EB with JS
	93.71	93.51	93.34	93.30

Table 5. Resnet-56 pruning results on CIFAR-10 using different orders ($p = 1, 2$) of Wasserstein distance.

Pruning ratio	FLOPs↓(%)	Top-1 accuracy(%)	
		$p = 1$	$p = 2$
40%	39.4	93.57	**93.64**
50%	49.4	93.47	**93.71**
60%	59.4	93.09	**93.30**

6 Discussions

6.1 Comparing Wasserstein metric with other measures

The Kullback-Leibler (KL) divergence and Jensen-Shannon (JS) divergence are commonly used to quantify the divergence between two distributions. The JS divergence is symmetric while the KL divergence is not. As bin-by-bin dissimilarity measures, these divergences only consider the relationships of bins belonging to the same index but ignore the information across bins [38]. Moreover, they require distributions to have joint supports [39]. However, Wasserstein distance (WD) can handle these problems well and be aware of the spatial information, thus captures perfectly the underlying probability space of the distributions [13].

Meanwhile, as a probability measure, Wasserstein barycenter (WB) captures important characteristics and preserves the basic structure of distributions better than the Euclidean barycenter (EB) [14].

To demonstrate the discussions above, we experiment with the EB/WB to compute barycenters, and JS/WD to compute the discrepancies. Results for ResNet-56 on CIFAR-10 are reported in Table 4, which show the effectiveness of our method.

6.2 Influence of the Order of the Wasserstein Distance

The discrepancies of channels can be computed using different orders of the Wasserstein distance. According to (2), the transport cost is ℓ_1-norm corresponding to the 1-Wasserstein distance for $p = 1$, and is squared Euclidean distance corresponding to the squared 2-Wasserstein distance for $p = 2$.

We explore the influence of the above two Wasserstein distances by pruning ResNet-56 on CIFAR-10. Top-1 accuracy drops of the pruned model are reported in Table 5. Obviously, the squared 2-Wasserstein distance achieves better results

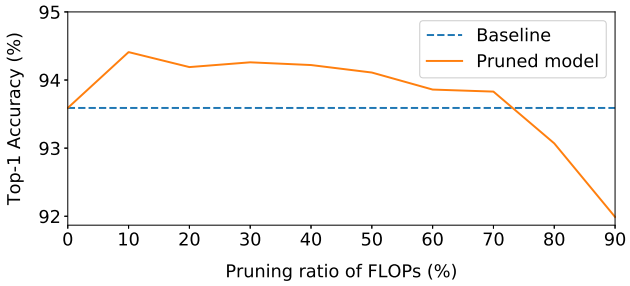

Fig. 1. Accuracy of ResNet-110 on CIFAR-10 regarding different pruning ratio of FLOPs. The dashed line and solid curve denote the baseline and pruned model respectively.

compared to the 1-Wasserstein distance, which means that 2-Wasserstein distance can find redundant channels more effectively. In addition, the proposed method is completely using the squared 2-Wasserstein distance to maintain consistency.

6.3 Varying Pruning Ratio of FLOPs

We conducted further experiments for ResNet-110 on CIFAR-10 with varying pruning ratios of FLOPs. Results are shown in Figure 1. According to Figure 1, when the pruning ratio increases to 10%, the performance rises sharply. It means that after pruning, the redundancy of ResNet is reduced, thereby alleviating the overfitting of the model. When the pruning ratio of FLOPs changes from 20% to 70%, the accuracy is slowly decreasing. As the pruning ratio larger than 70%, the accuracy of the pruned model drops dramatically, because the performance is limited by the high pruning ratio. It is worth noting that the pruned model always maintains a higher performance than the baseline when the pruning ratio is less than 70%. Overall, the high performance of the pruned model illustrates the powerful ability of the proposed method in compressing redundant models.

7 Conclusions

In this paper, a novel channel pruning method is proposed via the Wasserstein metric to accelerate CNN models. The Wasserstein barycenter is utilized to generate the basic response, which integrates the output features and summarizes the basic characteristics of each channel. Then the *channel discrepancy* of each channel measures the channel contribution, by considering both the channel's feature representation ability and channel relationships based on the Wasserstein distance. Finally, channels with the smallest discrepancies are selected and pruned. The comprehensive experiments on various popular network architectures verify the superior performance of the proposed method compared with the existing methods.

References

1. Krizhevsky, A., Hinton, G., et al.: Learning multiple layers of features from tiny images. Technical report, Citeseer (2009)
2. Simonyan, K., Zisserman, A.: Very deep convolutional networks for large-scale image recognition. In: ICLR. (2015)
3. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: CVPR. (2016) 770–778
4. Wang, T., Piao, Y., Li, X., Zhang, L., Lu, H.: Deep learning for light field saliency detection. In: ICCV. (2019)
5. Luo, J.H., Wu, J., Lin, W.: Thinet: A filter level pruning method for deep neural network compression. In: ICCV. (2017) 5058–5066
6. Wen, W., Wu, C., Wang, Y., Chen, Y., Li, H.: Learning structured sparsity in deep neural networks. In: NIPS. (2016) 2074–2082
7. Liu, Z., Li, J., Shen, Z., Huang, G., Yan, S., Zhang, C.: Learning efficient convolutional networks through network slimming. In: ICCV. (2017) 2736–2744
8. Ye, J., Lu, X., Lin, Z., Wang, J.Z.: Rethinking the smaller-norm-less-informative assumption in channel pruning of convolution layers. In: ICLR. (2018)
9. Lin, S., Ji, R., Li, Y., Wu, Y., Huang, F., Zhang, B.: Accelerating convolutional networks via global & dynamic filter pruning. In: IJCAI. (2018) 2425–2432
10. He, Y., Zhang, X., Sun, J.: Channel pruning for accelerating very deep neural networks. In: ICCV. (2017) 1389–1397
11. Wang, D., Zhou, L., Zhang, X., Bai, X., Zhou, J.: Exploring linear relationship in feature map subspace for convnets compression. arXiv preprint arXiv:1803.05729 (2018)
12. Singh, P., Verma, V.K., Rai, P., Namboodiri, V.P.: Leveraging filter correlations for deep model compression. arXiv preprint arXiv:1811.10559 (2018)
13. Villani, C.: Optimal transport: old and new. Volume 338. Springer Science & Business Media (2008)
14. Cuturi, M., Doucet, A.: Fast computation of wasserstein barycenters. In: ICML. (2014)
15. Solomon, J., De Goes, F., Peyré, G., Cuturi, M., Butscher, A., Nguyen, A., Du, T., Guibas, L.: Convolutional wasserstein distances: Efficient optimal transportation on geometric domains. ACM Transactions on Graphics **34** (2015) 66
16. Russakovsky, O., Deng, J., Su, H., Krause, J., Satheesh, S., Ma, S., Huang, Z., Karpathy, A., Khosla, A., Bernstein, M., et al.: Imagenet large scale visual recognition challenge. IJCV (2015) 211–252
17. Denton, E.L., Zaremba, W., Bruna, J., LeCun, Y., Fergus, R.: Exploiting linear structure within convolutional networks for efficient evaluation. In: NIPS. (2014) 1269–1277
18. Jaderberg, M., Vedaldi, A., Zisserman, A.: Speeding up convolutional neural networks with low rank expansions. In: BMVC. (2014)
19. Lebedev, V., Ganin, Y., Rakhuba, M., Oseledets, I.V., Lempitsky, V.S.: Speeding-up convolutional neural networks using fine-tuned cp-decomposition. In: ICLR. (2015)
20. Chen, W., Wilson, J., Tyree, S., Weinberger, K., Chen, Y.: Compressing neural networks with the hashing trick. In: ICML. (2015) 2285–2294
21. Courbariaux, M., Hubara, I., Soudry, D., El-Yaniv, R., Bengio, Y.: Binarized neural networks: Training deep neural networks with weights and activations constrained to+ 1 or-1. arXiv preprint arXiv:1602.02830 (2016)

22. Rastegari, M., Ordonez, V., Redmon, J., Farhadi, A.: Xnor-net: Imagenet classification using binary convolutional neural networks. In: ECCV, Springer (2016) 525–542
23. Zhao, R., Hu, Y., Dotzel, J., De Sa, C., Zhang, Z.: Improving neural network quantization without retraining using outlier channel splitting. In: ICML. (2019) 7543–7552
24. Liu, Z., Luo, W., Wu, B., Yang, X., Liu, W., Cheng, K.T.: Bi-real net: Binarizing deep network towards real-network performance. IJCV **128** (2020) 202–219
25. LeCun, Y., Denker, J.S., Solla, S.A.: Optimal brain damage. In: NIPS. (1990) 598–605
26. Hassibi, B., Stork, D.G.: Second order derivatives for network pruning: Optimal brain surgeon. In: NIPS. (1993) 164–171
27. Han, S., Mao, H., Dally, W.J.: Deep compression: Compressing deep neural networks with pruning, trained quantization and huffman coding. ICLR (2016)
28. Guo, Y., Yao, A., Chen, Y.: Dynamic network surgery for efficient dnns. In: NIPS. (2016) 1379–1387
29. Li, H., Kadav, A., Durdanovic, I., Samet, H., Graf, H.P.: Pruning filters for efficient convnets. In: ICLR. (2017)
30. He, Y., Kang, G., Dong, X., Fu, Y., Yang, Y.: Soft filter pruning for accelerating deep convolutional neural networks. In: IJCAI. (2018)
31. He, Y., Liu, P., Wang, Z., Hu, Z., Yang, Y.: Filter pruning via geometric median for deep convolutional neural networks acceleration. In: CVPR. (2019) 4340–4349
32. Cuturi, M.: Sinkhorn distances: Lightspeed computation of optimal transport. In: NIPS. (2013)
33. Zagoruyko, S., Komodakis, N.: Wide residual networks. In: BMVC. (2016)
34. He, Y., Lin, J., Liu, Z., Wang, H., Li, L.J., Han, S.: Amc: Automl for model compression and acceleration on mobile devices. In: ECCV. (2018) 784–800
35. Lin, S., Ji, R., Yan, C., Zhang, B., Cao, L., Ye, Q., Huang, F., Doermann, D.: Towards optimal structured cnn pruning via generative adversarial learning. In: CVPR. (2019) 2790–2799
36. Yu, J., Yang, L., Xu, N., Yang, J., Huang, T.S.: Slimmable neural networks. In: ICLR. (2019)
37. Molchanov, P., Mallya, A., Tyree, S., Frosio, I., Kautz, J.: Importance estimation for neural network pruning. In: CVPR. (2019)
38. Rubner, Y., Tomasi, C., Guibas, L.J.: The earth mover's distance as a metric for image retrieval. IJCV **40** (2000) 99–121
39. Arjovsky, M., Chintala, S., Bottou, L.: Wasserstein GAN. In: ICML. (2017) 214–223