Toward domain generalized pruning by scoring out-of-distribution importance

Anonymous Author(s)
Affiliation
Address
email

Abstract

Filter pruning has been widely used for compressing convolutional neural networks to reduce computation costs during the deployment stage. Recent studies have shown that filter pruning techniques can achieve lossless compression of deep neural networks, reducing redundant filters (kernels) without sacrificing accuracy performance. However, the evaluation is done when the training and testing data are from similar environmental conditions (independent and identically distributed), and how the filter pruning techniques would affect the cross-domain generalization (out-of-distribution) performance is largely ignored. We conduct extensive empirical experiments and reveal that although the intra-domain performance could be maintained after filter pruning, the cross-domain performance will decay to a large extent. As scoring a filter's importance is one of the central problems for pruning, we design the importance scoring estimation by using the variance of domain-level risks to consider the pruning risk in the unseen distribution. As such, we can remain more domain generalized filters. The experiments show that under the same pruning ratio, our method can achieve significantly better cross-domain generalization performance than the baseline filter pruning method. For the first attempt, our work sheds light on the joint problem of domain generalization and filter pruning research.

1 Introduction

In the past decade, convolutional neural networks (CNNs) have been rapidly developed and widely applied for computer vision [9, 8]. For better precision performance, CNN models are gradually designed deeper and larger. However, this will lead to more model parameters and thus result in a higher burden of computation, making the model inefficient for real-time inference.

Recent research shows that CNNs models are usually overparameterized, and the overparameterized models have redundant parameters that can be pruned without performance decay [5, 16, 10]. Among different model compression techniques, filter pruning is an effective structured pruning method, which prunes redundant convolutional filters or kernels (hereafter, we use filters, kernels, or neurons alternatively). It has been shown that state-of-the-art filter pruning methods [10, 3, 16] can achieve a high compression ratio with little performance drop. However, in the research community of neural network compression, the pruned network's performance is mainly evaluated based on the intra-domain setting, but the domain shift [13, 6] problem is largely ignored. Also, it has not been explored yet how the filter pruning would affect the performance when there exists domain shift between the training and testing data. We discuss the related works about generalization and pruning in Appendix A.1.

Submitted to Workshop on Distribution Shifts, 36th Conference on Neural Information Processing Systems (NeurIPS 2022). Do not distribute.

Motivated by the above discussion, we investigate how the filter pruning would affect the cross-domain generalization performance. We implement state-of-the-art filter pruning method [16]. Through experiments on the Domain Generalization (DG) benchmark, we find that the pruning method can preserve the intra-domain performance after pruning. However, the cross-domain testing shows significant performance decay when the distribution shift (domain shift) exists between the training and testing data. Therefore, we reveal that **pruning can degrade cross-domain generalization performance significantly**. This is a crucial problem urgent to be solved because the so-called "lossless" compressed models would work unexpectedly in the real world, where distribution shift problem usually appears.

To alleviate the problem of unexpected unseen-domain performance drops from pruning, we revisit existing pruning methods. In the common pruning-then-finetuning paradigm, [1, 15, 17, 3, 16, 7], and a filter' importance score is calculated. Then, the filters with the lowest importance are masked or pruned. Thus, scoring a filter's importance is one of the central problems in pruning [21]. Existing importance criteria are mainly based on magnitude [3], gradient information [16] or feature information [7], which may work with the Independent and Identically Distributed (i.i.d) assumption, but cross-domain generalization is ignored which is also named the Out-of-Distribution (o.o.d) problem. Thus, directly pruning low-importance neurons will result in unexpected performance drops in the unseen target data domain.

As the neurons' importance scoring is one of the central problems in pruning [21], we explore improving the importance estimation by taking into account the out-of-distribution risks. In this paper, we introduce the Importance of out-of-disribution Risk (IoR) to measure the filter importance, which is simple to implement by just including the gradient information of the variance of domain level risks to calculate the importance. In the experiments of the DG benchmarks PACS [12], we evaluate both the intra-domain and cross-domain performance before and after pruning. Compared with the baseline pruning method [16], our method with IoR can achieve better cross-domain generalization performance under the same pruning ratio.

2 Preliminary

A CNN model that has a number of M filters can be parameterized as $\Theta = \{\theta_1, \theta_2, ...\theta_M\}$. Each filter θ_m is a convolutional kernel. A heuristic solution is to select unimportant filters to be pruned and then finetune the pruned model [16]. Therefore, how to measure the importance of filters is one of the central problems [21]. When there N multiple data domains $\mathbb{D} = \{D_1, D_2, ...D_N\}$ available, the importance can be expressed as

$$I_m(\Theta) = (\frac{1}{N}\sum_i^N \mathcal{R}_i(X, Y, \Theta) - \frac{1}{N}\sum_i^N \mathcal{R}_i(X, Y, \Theta|\theta_m = 0))^2, \tag{1}$$

where $\mathcal{R}_i$ denotes the empirical risk of domain D_i, and $\theta_m = 0$ means that filter θ_m is pruned. Based on first-order taylor expansion [16], Eq.1 can be approximated as

$$I_m(\Theta) = (\frac{1}{N}\sum_i^N \frac{\partial \mathcal{R}_i}{\partial \theta_m}\theta_m)^2 \tag{2}$$

As shown in Eq. 1, the importance of filter θ_m is measured by evaluating how much risk is increased if θ_m is pruned ($\theta_m = 0$). If the i.i.d assumption holds, the training data and testing data are drawn from the same distribution. As such, Eq. 1 can measure the importance of a filter w.r.t the testing data. However, distribution shift often exists in the real world, and testing data could be in a different distribution of training data. As such, directly using Eq. 1 for pruning could lead to unexpected performance drop in an unseen data distribution. To reduce the sacrifice of testing performance on the unseen data domain, we propose to select more generalized filters and pruned those less generalized filters by improving the importance scoring criterion.

3 Scoring out-of-distribution importance

To improve the importance scoring, we consider the out-of-distribution risk to design the importance score. Inspired by risk extrapolation [11], we approximate the out-of-distribution risk $\mathcal{R}^o$ by

$$\mathcal{R}^o \approx Var\{\mathcal{R}_1, \mathcal{R}_2, ...\mathcal{R}_N\}, \tag{3}$$

Table 1: Pruning ResNet18 and ResNet50 models on the PACS benchmark. "Intra" means the intra-domain top-1 accuracy (%), and "Cross" means the cross-domain top-1 accuracy (%)

Model	Art		Cartoon		Photo		Sketch	
	Intra	Cross	Intra	Cross	Intra	Cross	Intra	Cross
ResNet18 (before pruning)	99.63	76.86	100	78.20	97.39	93.59	99.68	77.86
ResNet18-50% (baseline)	99.28	74.94	99.97	77.50	95.82	92.04	99.27	**77.41**
ResNet18-50% with IoR (ours)	99.34	**75.60**	99.90	**78.79**	96.80	**92.53**	99.42	75.98
ResNet18-30% (baseline)	97.12	**68.85**	97.86	**73.74**	91.34	85.09	96.79	69.51
ResNet18-30% with IoR (ours)	95.94	68.37	97.59	72.31	92.98	**86.37**	96.55	**70.81**
ResNet50 (before pruning)	95.04	83.50	96.78	81.27	96.56	96.11	99.84	83.53
ResNet50-50% (baseline)	99.13	82.84	99.82	80.30	98.49	93.68	100	79.04
ResNet50-50% with IoR (ours)	99.22	**83.36**	98.36	**80.56**	98.10	**94.49**	99.97	**79.74**
ResNet50-30% (baseline)	97.37	74.53	97.99	76.79	93.14	86.86	98.57	74.87
ResNet50-30% with IoR (ours)	97.55	**75.76**	98.12	**77.11**	96.68	**91.47**	98.57	**78.36**

Table 2: Pruning 50% of the filters of the ResNet models pretrained by ERM, CORAL, and Mixup on the PACS benchmark. "Intra" means the intra-domain top-1 accuracy (%), and "Cross" means the cross-domain top-1 accuracy (%).

PretrainMethod	Model	Art		Cartoon		Photo		Sketch	
		Intra	Cross	Intra	Cross	Intra	Cross	Intra	Cross
ERM	ResNet50	95.04	83.50	96.78	81.27	96.56	96.11	99.84	83.53
	ResNet50-50%	99.13	82.84	99.82	80.30	98.49	93.68	100	79.04
	Δ	+4.09	-0.66	+3.14	-0.97	+1.93	-2.43	+0.16	-4.49
CORAL	ResNet50	96.15	87.06	98.45	83.96	99.29	96.47	97.57	81.65
	ResNet50-50%	96.77	80.37	98.20	76.83	99.17	94.49	97.08	73.02
	Δ	+0.62	-6.69	-0.25	-7.12	-0.12	-1.98	-0.49	-8.63
Mixup	ResNet50	96.52	87.55	94.47	82.25	98.70	97.13	99.03	80.63
	ResNet50-50%	96.40	79.15	96.14	79.14	98.58	94.91	98.21	69.41
	Δ	-0.12	-8.40	+1.67	-3.11	-0.12	-2.22	-0.81	-11.22

where Var represents the variance of empirical risks from the source domains.

As such, the importance in Eq. 2 can be improved as I_m^{IoR}:

$$I_m^{IoR}(\Theta) = (\frac{1}{N}\sum_{i}^{N}\frac{\partial \mathcal{R}_i}{\partial \theta_m}\theta_m)^2 + \alpha(\frac{\partial Var\{\mathcal{R}_1, \mathcal{R}_2, ...\mathcal{R}_N\}}{\partial \theta_m}\theta_m)^2, \tag{4}$$

where α is a constant scaling factor and $\alpha = 1$ in our experiments. Since the gradient $\frac{\partial \mathcal{R}_i}{\partial \theta_m}$ can be obtained during the training and pruning, there would not be much extra computation needed for calculating Eq. 2. During pruning, importance scores of all filters are obtained. The filters are ranked according to importance scores in ascending order. In each iteration, top-K remained filters are removed and then finetuned.

4 Experiments

4.1 Implementation

We follow the leave-one-domain-out protocol and the setting used in [25]. The setup of data of training, (intra-domain) validation, and (cross-domain) testing can be found in Appendix A.2. To implement the baseline, we use the official implementation of Taylor Pruning with the suggested hyper-parameters [16]. Detailed parameters setting can be found in Appendix A.3. Our proposed method also uses the same hyperparameters. All experimental results are reported by averaging the results from five independent runs with different random seeds.

4.2 Experimental Results

We present the results by asking three research questions.

Research Question Q1: Will filter pruning affect the domain generalization performance?

Following DomainBed[1] [6], we first use the Empirical Risk Minimization (ERM) to pretrain the ResNet18 and ResNet50 models, which are often used as backbones in domain generalization benchmarks [19, 22]. Then, we use the baseline pruning method [16] to prune the pretrained ResNet18 and ResNet50 models. After pruning, 50% or 30% of filters remain, and the pruned models are denoted as ResNet18/50-50% or ResNet18/50-30%, respectively, as shown in Table 1. We evaluate the top-1 accuracy of intra-domain and cross-domain accuracy before and after pruning.

Table 1 shows the pruning results on the PACS benchmark. When we compare the intra/cross-domain accuracy before and after pruning, we find that 1) *cross-domain generalization performance is not guaranteed even the intra-domain evaluation shows "lossless" compression results*. As shown in Table 1, when the pruning to 50% or 30%, the intra-domain accuracy could have insignificant decay and could even improve to some extent, which indicates the "lossless" pruning. However, we can see the pruned models' cross-domain generalization performance degrades significantly in different experiments (Art, Cartoon, Photo, Sketch) under different compression ratios. The most interesting experiment is the *Photo* experiment of ResNet50. It appears that the original model has good generalization performance since the cross-domain accuracy is close to intra-domain accuracy before pruning (96%). However, after pruning (50% or 30%), the compressed model's cross-domain accuracy results are far lower than intra-domain results.

Research Question Q2: Can we reduce the sacrifice of the cross-domain performance from pruning?
We explore this question by using our proposed importance scoring based on out-of-distribution risk. In Table 1, we compare the pruning results of the baseline and our method (with IoR). When comparing the cross-domain accuracy of the compression ratios 50% and 30% for ResNet18 and ResNet50, there are 16 groups to be compared. Our IoR can achieve better results in 13 out of 16 comparison pairs (see the bold **accuracy** results). As such, more cross-domain generalization performance can be saved from pruning by using our IoR. Thus, our work shows that designing a better importance criterion is a feasible way to save cross-domain generalization from pruning. In future work, we could design better criteria to further save the cross-domain generalization performance.

Research Question Q3: Is a model pretrained by domain-generalization methods more than ERM robust after pruning? We then investigate if a model pretrained by using domain generalization methods can be more robust during the pruning. Similar to Q1, we follow DomainBed [6] to use the domain generalization methods CORAL [18] and Mixup [23] to pretrain models before filter pruning. Table 2 compares the accuracy results of ResNet50 models pretrained by ERM, CORAL, and Mixup, before and after pruning 50% of filters, on the PACS benchmarks. In the experiments of Art, Cartoon and Photo, CORAL and Mixup can help to achieve higher cross-domain accuracy results than ERM (before pruning). Also, for all experimental results of CORAL and Mixup in Table 2, the intra-domain accuracy performance is well maintained, with a drop of 0.81% at most. However, after pruning, we observe that the cross-domain accuracy results of CORAL and Mixup generally drop far more than that of the ERM. We conjecture the reason that in the pruning-then-finetuning paradigm, the finetuning is based on ERM, which has different optimization processes from the DG optimization methods. Therefore, when pruning the CORAL and Mixup pretrained models, finetuning the pruned models with ERM could make the optimization process inconsistent, leading to poorer performance.

5 Conclusion

In this work, we investigate how filter pruning would affect the cross-domain generalization when there exists domain shift between training and testing data. By conducting the experiments to answer the three research questions we raise, we reveal that: 1) *"lossless" compression can lead to significant cross-domain performance drop, even with few decays of intra-domain accuracy*; 2) *using domain generalization methods to pretrain a more generalized model (than ERM) could not help to reserve the drop accuracy from pruning*; 3) *our IoR can alleviate the cross-domain accuracy drop from pruning by a better importance criterion*. Our work shed the light on the joint problem of pruning and domain generalization. We wish our work can attract more research attention to the joint problem of pruning and domain generalization.

[1] https://github.com/facebookresearch/DomainBed

References

[1] Reza Abbasi-Asl and Bin Yu. Structural compression of convolutional neural networks. *arXiv preprint arXiv:1705.07356*, 2017.

[2] Shangyu Chen, Wenya Wang, and Sinno Jialin Pan. Cooperative pruning in cross-domain deep neural network compression. In *International Joint Conferences on Artificial Intelligence*, 2019.

[3] Xiaohan Ding, Tianxiang Hao, Jianchao Tan, Ji Liu, Jungong Han, Yuchen Guo, and Guiguang Ding. Resrep: Lossless cnn pruning via decoupling remembering and forgetting. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 4510–4520, 2021.

[4] Xiaoyu Feng, Zhuqing Yuan, Guijin Wang, and Yongpan Liu. Admp: An adversarial double masks based pruning framework for unsupervised cross-domain compression. *arXiv preprint arXiv:2006.04127*, 2020.

[5] Jonathan Frankle and Michael Carbin. The lottery ticket hypothesis: Finding sparse, trainable neural networks. *arXiv preprint arXiv:1803.03635*, 2018.

[6] Ishaan Gulrajani and David Lopez-Paz. In search of lost domain generalization. *arXiv preprint arXiv:2007.01434*, 2020.

[7] Jinyang Guo, Wanli Ouyang, and Dong Xu. Channel pruning guided by classification loss and feature importance. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 10885–10892, 2020.

[8] Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross Girshick. Mask r-cnn. In *Proceedings of the IEEE international conference on computer vision*, pages 2961–2969, 2017.

[9] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[10] Yihui He, Ji Lin, Zhijian Liu, Hanrui Wang, Li-Jia Li, and Song Han. Amc: Automl for model compression and acceleration on mobile devices. In *Proceedings of the European conference on computer vision (ECCV)*, pages 784–800, 2018.

[11] David Krueger, Ethan Caballero, Joern-Henrik Jacobsen, Amy Zhang, Jonathan Binas, Dinghuai Zhang, Remi Le Priol, and Aaron Courville. Out-of-distribution generalization via risk extrapolation (rex). In *International Conference on Machine Learning*, pages 5815–5826. PMLR, 2021.

[12] Da Li, Yongxin Yang, Yi-Zhe Song, and Timothy M Hospedales. Deeper, broader and artier domain generalization. In *Proceedings of the IEEE international conference on computer vision*, pages 5542–5550, 2017.

[13] Haoliang Li, Sinno Jialin Pan, Shiqi Wang, and Alex C Kot. Domain generalization with adversarial feature learning. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 5400–5409, 2018.

[14] Lucas Liebenwein, Cenk Baykal, Brandon Carter, David Gifford, and Daniela Rus. Lost in pruning: The effects of pruning neural networks beyond test accuracy. *Proceedings of Machine Learning and Systems*, 3:93–138, 2021.

[15] Jian-Hao Luo, Jianxin Wu, and Weiyao Lin. Thinet: A filter level pruning method for deep neural network compression. In *Proceedings of the IEEE international conference on computer vision*, pages 5058–5066, 2017.

[16] Pavlo Molchanov, Arun Mallya, Stephen Tyree, Iuri Frosio, and Jan Kautz. Importance estimation for neural network pruning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 11264–11272, 2019.

[17] Pavlo Molchanov, Stephen Tyree, Tero Karras, Timo Aila, and Jan Kautz. Pruning convolutional neural networks for resource efficient inference. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*. OpenReview.net, 2017.

[18] Baochen Sun and Kate Saenko. Deep coral: Correlation alignment for deep domain adaptation. In *European conference on computer vision*, pages 443–450. Springer, 2016.

[19] Chris Xing Tian, Haoliang Li, Xiaofei Xie, Yang Liu, and Shiqi Wang. Neuron coverage-guided domain generalization. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, pages 1–1, 2022.

[20] Eric Tzeng, Judy Hoffman, Kate Saenko, and Trevor Darrell. Adversarial discriminative domain adaptation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 7167–7176, 2017.

[21] Huan Wang, Can Qin, Yulun Zhang, and Yun Fu. Neural pruning via growing regularization. In *International Conference on Learning Representation (ICLR)*, 2021.

[22] Jingye Wang, Ruoyi Du, Dongliang Chang, and Zhanyu Ma. Domain generalization via frequency-based feature disentanglement and interaction. *arXiv preprint arXiv:2201.08029*, 2022.

[23] Shen Yan, Huan Song, Nanxiang Li, Lincan Zou, and Liu Ren. Improve unsupervised domain adaptation with mixup training. *arXiv preprint arXiv:2001.00677*, 2020.

[24] Kaiyang Zhou, Ziwei Liu, Yu Qiao, Tao Xiang, and Chen Change Loy. Domain generalization: A survey. *arXiv e-prints*, pages arXiv–2103, 2021.

[25] Kaiyang Zhou, Yongxin Yang, Timothy Hospedales, and Tao Xiang. Learning to generate novel domains for domain generalization. In *European conference on computer vision*, pages 561–578. Springer, 2020.

A Appendix

A.1 Related works

With regards to pruning and generalization, the most relevant work is [14]. While [14] mainly focuses on the pruning and robustness against corruption. Our work mainly focuses on the pruning and domain generalization toward “covariate shift".

Also, there are other papers about cross-domain compression [2, 4], aiming to utilize the source domain data to improve the pruning performance in the target domain. In [2, 4], the target domain data is available for training and pruning, which is similar to the **domain adaptation** setting [20]. However, our work is related to the **domain generalization** setting [13, 24], where the target domain data is unseen (unavailable) during the pruning process.

A.2 Evaluation protocol

We follow the leave-one-domain-out protocol and the setting used in [25]. The data in PACS of each domain is split into two parts according to training : validation = 9 : 1. In each experiment, one domain is left as the unseen target domain for testing, while the rest data of the other domains is used as source domains for training. For example, in the experiment of Photo of the PACS benchmark, the training parts of Art, Cartoon, and Sketch are combined for the model pretraining and pruning. Meanwhile, the validation parts of Art, Cartoon, and Sketch are combined to validate the top-1 accuracy. The pruned model with the best top-1 validation accuracy is selected for the testing, and the best validation accuracy is reported as the intra-domain performance. After that, we use all the data of the Photo domain as the unseen target domain to test the cross-domain performance in experiments of before or after pruning.

A.3 Implementation details

As the baseline pruning method is equivalent to the Taylor pruning [16], we use its official PyTorch implementation [2] to conduct baseline pruning experiments. After each convolutional layer, we utilize the gate replacement to use the gate's importance to represent its filter's importance [16] . Following the recommended settings [16], we use batch size 64, learning rate 0.001, and epochs 100. The pruning interval is 30 mini-batches, during which a maximum of 100 filters would be pruned. Also, we use the exponential moving average filter (momentum) with a coefficient 0.9 to calculate the importance scores over different mini-batches. The experimental results are reported by averaging the results five independent runs with different random seeds.

[2] https://github.com/NVlabs/Taylor_pruning