Enhanced Sparsification via Stimulative Training

Shengji Tang*[1], Weihao Lin*[1], Hancheng Ye[3],
Peng Ye[1], Chong Yu[2], Baopu Li[4], and Tao Chen†[1]

[1] School of Information Science and Technology, Fudan University, Shanghai, China
[2] Academy for Engineering and Technology, Fudan University, Shanghai, China
[3] Shanghai AI Laboratory, Shanghai, China
[4] Independent Researcher
eetchen@fudan.edu.cn

Abstract. Sparsification-based pruning has been an important category in model compression. Existing methods commonly set sparsity-inducing penalty terms to suppress the importance of dropped weights, which is regarded as the suppressed sparsification paradigm. However, this paradigm inactivates the dropped parts of networks causing capacity damage before pruning, thereby leading to performance degradation. To alleviate this issue, we first study and reveal the relative sparsity effect in emerging stimulative training. Based on the sparsity effect, we propose a structured pruning framework, named STP. It is based on an enhanced sparsification paradigm which maintains the magnitude of dropped weights and enhances the expressivity of kept weights by self-distillation. Besides, to find an optimal architecture for the pruned network, we propose a multi-dimension architecture space and a knowledge distillation-guided exploration strategy. To reduce the huge capacity gap of distillation, we propose a subnet mutating expansion technique. Extensive experiments on various benchmarks indicate the effectiveness of STP. Specifically, without fine-tuning, our method consistently achieves superior performance at different budgets, especially under extremely aggressive pruning scenarios, e.g., remaining 95.11% Top-1 accuracy (72.43% in 76.15%) while reducing 85% FLOPs for ResNet-50 on ImageNet. Codes are at https://github.com/tsj-001/STP.

Keywords: Sparisification · Structured pruning · Self distillation

1 Introduction

Recently, the model pruning [5, 18, 41] technique, which removes redundant parameters to obtain a compact network from an over-parameterized one, has become a mainstream and general model compression method. Early pruning methods [19, 20, 33] focus on designing delicate metrics, e.g., L_1 score [19] or feature activation [29], to measure the importance of parameters and discard the unimportant ones. However, directly removing parameters from the original

*These authors contributed equally.
†Corresponding author

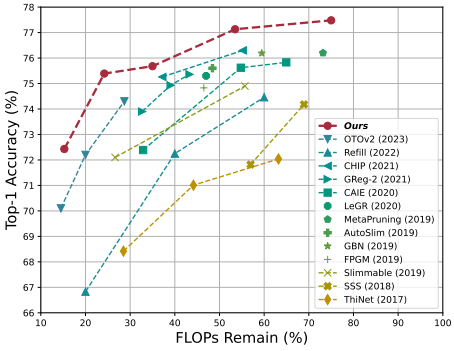


Fig. 1: ResNet-50 on ImageNet dataset. Top-1 accuracy (%) and remaining FLOPs (in percentage) are reported. Without fine-tuning, our method can still obtain an optimal Pareto frontier of efficiency and performance compared with other methods.

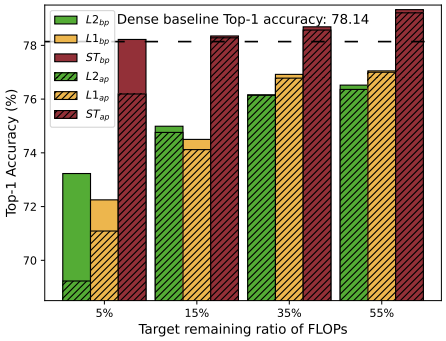


Fig. 2: Different sparsification methods, namely L_1, L_2, and ST, on CIFAR-100 dataset under different FLOPs. The subscripts **ap** and **bp** represent the performance of networks **after** and **before** pruning, respectively.

network often causes dramatic performance degradation, especially in structured pruning where groups of filters are removed. To alleviate the performance degradation, emerging works [13,25,52,53,58] are dedicated to sparsifying parameters before pruning or sparsifying and pruning alternately. The sparsification aims to alter the distribution of parameters and transfer the network expressivity to the kept parameters, which helps to maintain performance after pruning. In current literature, the penalty terms [4,19,54] are commonly employed as regularization to introduce sparsification. They suppress the importance of dropped parameters, e.g., encouraging the pruned parameters to converge to zero [43], which is regarded as the suppressed sparsification paradigm in this manuscript. However, it is pointed out that the suppressed sparsification paradigm commonly causes expressivity damage of the unpruned network during the sparsification procedure [13,52], which limits the pruning upper bound. Numerous works have attempted to mitigate it by using a small regime of penalty stength [41, 54] or well-designed penalty coefficients [13, 52, 53], but only alleviates rather than completely solves this issue. The reason is that the suppressed sparsification paradigm introduces explicit regularization to forcibly drive the pruned parts toward zero and obtain a sparse distribution that is easier to prune. **This process is nearly equivalent to removing the corresponding parameters, causing structural damage and loss of network capacity.** It is detrimental to the preservation of expressive capability before pruning.

From the dual respect of suppression, we consider achieving relative sparsity by enhancing the kept parts while maintaining the magnitude of the to-be-pruned parts, termed enhanced sparsification paradigm. Fig. 3 provides an intuitive comparison of the suppressed and enhanced sparsification. The core consideration behind the enhanced sparsification paradigm is that the enhanced kept parts can occupy a greater significance in the contribution to the overall network's expressive capability while avoiding damage to the entire network. It

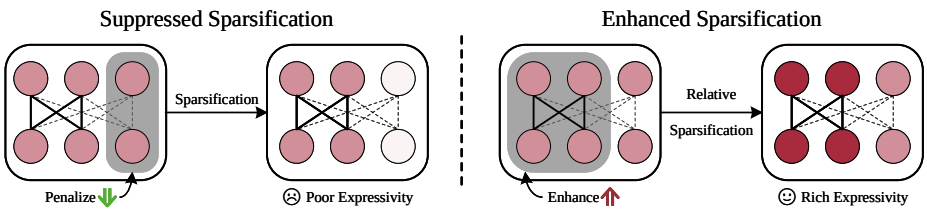


Fig. 3: Comparison of suppressed and enhanced sparsification. The exemplified fully connected layer has three input neurons and three output ones. The solid black lines denote the parameters to remain, while the dashed ones are pruned. The parameters connected to a redder neuron have larger magnitudes.

is noteworthy that relative sparsity differs from the commonly used definition of sparsity. Relative sparsity refers to a situation where some parameters in the network have a notably greater ratio (not the absolute value) than others on certain metrics, such as the L_1 norm. To realize the enhanced sparsification paradigm, we focus on the original purpose of sparsification, which transfers the expressivity from the whole to the kept part for lossless pruning. A popular technique for transferring is knowledge distillation (KD) [26, 31], which aligns the output logits or features to transfer the dark knowledge. Recently, Stimulative Training (ST) [60, 62] utilizes self-distillation to transfer the knowledge from the main network to its weight-sharing subnets and benefits both the main network and subnets. We consider the self-distillation process in ST can be seen as a natural enhancement. As shown in Fig. 4, we study the parameter distribution in ST and observe that without an explicit sparsity constraint, ST results in prominent relative sparsity, which will be further discussed in Sec. 3.2.

Based on the relative sparsity effect in ST, we propose a new enhanced sparsification paradigm and build a one-stage multi-dimension framework for structured pruning, namely ST guided pruning (STP). The core of the method lies in, given a subnet, utilizing the self-distillation in ST to distill the main network to the weight-sharing subnet. This enhances the importance of the subnet, achieves relative sparsification, and thereby facilitates lossless structured pruning.[¶] Besides, STP introduces three key designs to guarantee a high-performance compact network: 1) KD guided exploration. It is a heuristic searching strategy driven by the KD loss to obtain an outstanding subnet for sparsification. We introduce KD guided exploration to gradually explore surpassing subnet architectures during ST; 2) Multi-dimension sampling. The original ST [62] only considers sampling subnets with different depths (total layer numbers). To formulate a more general pruning framework and chase for a better trade-off between sparsity and performance, we extend the sampling dimension of ST to both depth and width (the output channel number of each layer) and design a multi-dimension FLOPs-aware sampling space to fit the desired FLOPs target correctly; 3) Subnet mutating expansion. The FLOPs-aware sampling space leads to monotonic subnet capacity, which harms the performance of the main

[¶]It is noted that since unstructured pruning hardly promises practical speedup, STP only focuses on structured pruning.

network [49]. The subnet mutating expansion generates larger subnets to enrich the subnet capacity, improving the main network.

We summarize our contributions as follows: 1) We first reveal the relative sparsity effect of ST, which renders ST to be an enhanced sparsification paradigm maximumly retaining the capacity of the unpruned network, thus alleviating performance drop after pruning; 2) Based on the enhanced sparsification paradigm, we propose a pruning framework, named stimulative training guided pruning (STP). The STP includes a KD guided architecture exploration method to find optimal subnet architectures, a multi-dimension sampling to better trade-off between sparsity and performance, and a subnet mutating expansion scheme to introduce various support subnets for further performance improvement; 3) Extensive empirical results and comparisons on various mainstream models (ResNet-50, WRN28-10, MobileNetV3, ViT, Swin Transformer) and datasets (CIFAR-100, TinyImageNet, ImageNet, COCO) show that STP obtains compact networks with high performance and extremely low FLOPs without fine-tuning, e.g., preserving 95.11% performance (72.43% in 76.15% Top-1 accuracy) while reducing 85% FLOPs on ImageNet (as shown in Fig. 1).

2 Related Work

2.1 Sparsification-based Pruning

Existing sparsification-based pruning methods can be divided into two categories: static sparsification and dynamic sparsification. Static sparsification aims at imposing a global and unchanged penalty strength to all parameters of the model [3, 4, 14, 19, 36, 38, 41, 43, 54, 56]. The penalty terms in these works are designed to be a norm-based regularization item w.r.t. model parameters. Different from static sparsification methods, dynamic sparsification considers the individual and time-varying sparsification strength of different parameters [4, 9, 13, 52, 53, 53, 59, 61, 66]. AFP [13] utilizes auto-balanced regularization for different weights to transfer the representational capacity from the whole network to the kept part. Greg [52] merely applies sparsification on the dropped parameters and grows the penalty term large gradually to achieve higher sparsity. Previous works mostly follow the suppressed sparsification paradigm, which imposes constraints on dropped parameters, causing expressivity damage before pruning. Differently, our method achieves relative sparsity by enhancing kept parameters and maintaining the magnitude of dropped ones via stimulative training.

2.2 Knowledge Distillation in Pruning

In general, pruning a network inevitably deteriorates its performance. To compensate for the performance degradation, conducting knowledge distillation (KD) after pruing has been proved feasible [1,2,7,35,45,46,57]. In [2], the authors firstly obtain a pruned network via APoZ [29] and then transfer knowledge from the unpruned network to the pruned one via a cosine-similarity-based loss. The authors in [45] apply a Kullback-Leibler-divergence-based loss for KD and verify

its effectiveness in both one-shot and iterative magnitude pruning. Built upon iterative magnitude pruning, the authors in [46] propose a cross-correlation-based KD loss to better align the pruned network with the unpruned one. The aforementioned methods treat KD as a performance booster isolated from pruning. Differently, we treat KD itself as a pruning method, not merely a support technique. Our proposed method incorporates KD into pruning, utilizing self-distillation for sparsification and KD loss to obtain the pruning mask.

2.3 Stimulative Training

Stimulative training (ST) [62] aims to boost the main network by transferring the knowledge from the main network to each depth-wise subnet. In ST, the main network can be seen as an ensemble of shallow subnets, and the final performance is determined by each subnet under the unraveled view [48, 51]. Thus, while training the main network, ST randomly skips some layers to generate subnets and utilizes knowledge distillation to supervise each subnet. The experiments in ST show that ST can obtain performance improvement on the main network while bringing various subnets with marginal performance reduction compared with the main network. This additional benefit of subnets suggests that ST can achieve simultaneous enhancement at both local and global of the main network, which inspires us to study the distribution of parameters in ST and exploit ST as an enhanced sparsification paradigm for pruning.

3 Method

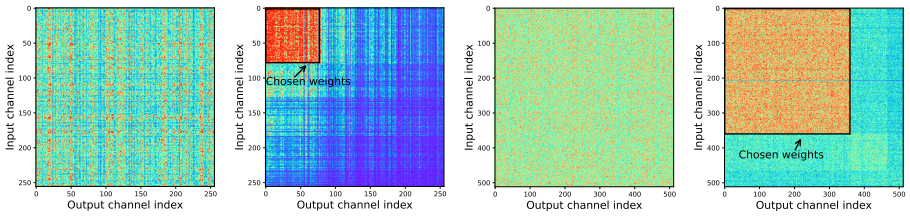
3.1 Problem Formutation

Pruning aims to eliminate parameters from a given network $\mathcal{N}$ with parameters θ to obtain a high-performance compact subnet $\mathcal{N}_s$ with retaining parameters θ_s with specific resource constraint. Although $\mathcal{N}$ can be given with pretrained weights, our method concentrates on pruning from randomly initialized parameters, same as [4, 27]. The pruning process can be abstracted as a function $F(\cdot)$, which takes θ as input and output θ_s, $\theta_s = F(\theta)$[‖]. General pruning mainly consists of two categories of transformations 1) $\mathcal{M} = F_1(\theta)$, which generates a binary mask to determine remaining and dropping parts 2) $\hat{\theta} = F_2(\theta)$, which adjusts the parameter distribution for eliminating with low damage. A typical pruning process can be denoted as:

$$\theta_s = \mathcal{M} \odot \hat{\theta} = F_1(\theta) \odot F_2(\theta). \tag{1}$$

The exact format depends on the specific pruning algorithm. For one-shot pruning, F_1 is applied once [34] while alternately for iterative pruning [17]. We denote the dropped parameters as $\theta_d = \theta \setminus \theta_s$. The specification process belongs to $F_2(\cdot)$

[‖]Because the dependence of dataset is upon the pruning algorithm, for simplicity, we omit the given dataset D as input.



(a) Baseline, layer 3-1 **(b)** ST, layer 3-1, 0.3 **(c)** Baseline, layer 4-2 **(d)** ST, layer 4-2, 0.7

Fig. 4: The convolutional weight magnitude distribution of baseline (standard training without sparsification) and stimulative training (ST) [62] with different sparsity ratios (0.3 and 0.7) in different layers (3-1 and 4-2) on ResNet-50, where "3-1" and "4-2" are the layer index, e.g., given ResNet-50 containing four stages with layers [3,4,6,3], "3-1" represents the first layer of the third stage. The color represents the **relative** magnitude comparison. The redder regions indicate the relatively larger magnitudes, and vice versa. The regions encircled by the black rectangle represent the chosen (kept) weights. Given a structured architecture, ST exhibits a significant relative sparsity effect, enhancing chosen weights accurately.

and a typical method [4, 52, 53] introduces L_2 norm penalty term for θ_d, which optimizes the following objective:

$$\mathcal{L}_{total}(\theta; D) = \mathcal{L}_{task}(\theta; D) + \frac{\lambda}{2}||\theta_d||_2^2, \tag{2}$$

where $\mathcal{L}_{task}$ is the task loss; D is the given dataset;λ is the coefficient of penalty term. The penalty term drives $||\theta_d||$ to converge to zero and eliminates the importance of θ_d for painless removal, which belongs to suppressed paradigm. Whereas θ_d is a part of θ, suppression of θ_d reduces the capacity of θ, which limits the upper bound of θ_s and compromises the final performance.

3.2 Relative Sparsity Effect in Stimulative Training

To boost the performance of the main network, stimulative training (ST) views a network as an ensemble of multiple shallow subnets and transfers the knowledge from the main network to each weight-shared subnet. Formally, ST optimizes the following loss:

$$\mathcal{L}_{ST} = CE(p(x; \theta_m), y) + \lambda KL(p(x; \theta_m)||p(x; \theta_s)), \tag{3}$$

where x and y are the input images and ground truth labels, respectively; $CE(\cdot)$ is the cross entropy; $KL(\cdot)$ is the Kullback-Leibler divergence; λ is the balance coefficient; $p(x; \theta)$ is the output probability with x and θ; θ_m and θ_s are the parameters of the main network and subnet, respectively, and θ_s is the subset of θ_m. According to the optimization of the second term of $\mathcal{L}_{ST}$, *i.e.*, $KL(p(x; \theta_m)||p(x; \theta_s))$, ST can gradually transfer the representative ability from the whole parameters (θ_m) to the partial ones (θ_s), causing the enhancement of subnet weights and implying the potential of relative sparsity.

To investigate the relative sparsity effect in ST, we conduct ST by fixing a specific subnet on ResNet-50 and visually show the magnitude of a convolutional layer's weights. Notely, while the original ST only samples subnet in the depth dimension, we extend it into the width dimension. As shown in Fig. 4, compared with the random weights' distribution in the baseline, *i.e.*, standard training with stochastic gradient descent, ST can significantly enhance the chosen weights, causing the weights' magnitude relatively concentrating on the chosen weights (the black rectangle). It is worth noting that Fig. 4 reports the relative comparison, see Fig. 7 for detailed absolute values. To further display the effect of different sparsification methods before and after pruning, we conduct experiments with different target sparsity. As shown in Fig. 2, the L norm sparsifications harm the performance before pruning (before removing the dropped parameters), while ST maintains and even improves the performance. It suggests ST introduces a moderate and relative sparsity effect via KD to keep the capacity of the dropped parts, which encourages learning better sparsity distribution.

3.3 Framework

The proposed ST guided pruning (STP) framework is illustrated in Fig.5. In detail, given a dense network, *i.e.*, the main network and a target FLOPs constraint, we initialize an architecture pool containing structured subnet architectures sampled from multi-dimension FLOPs-aware sampling space. The training starts with randomly initialized parameters. In each training iteration, the main network firstly forwards to generate the main output, compute and backward the cross entropy between ground truth and main output. Then, a subnet architecture is sampled based on the recorded score (KD loss) in the architecture pool and applied to the main network to construct a weight-shared subnet. The subnet is supervised using the main output by knowledge distillation. Meanwhile, the sampled architecture mask is mutated multi-dimensionally into a larger architecture mask, which is utilized to generate a weight-shared mutating network and also supervised using the main output. To search the optimal architecture, the architecture with the highest score (KD loss) is removed in the architecture pool every k iterations. Due to a designed shrinking schedule, only one subnet architecture mask exists in the architecture pool after T training iterations. We apply it to the main network to obtain the final pruned network while the other parameters are deserted. A typical value of k is the iteration number of an epoch. The pseudo-code is shown in the Appendix. Three critical designs of STP are introduced as follows.

Stimulative Training Sparsification and Multi-dimension Sampling. As stated in Sec. 3.2, inspired by the relative sparsity effect in ST, we exploited it for enhanced sparsification. However, the original ST sampling space only considers the depth dimension (all possible layer combinations), ignoring the more fine-grained width dimension (output channels). The relatively coarse-grained depth sampling space cannot fully explore the trade-off between sparsity and performance. Thus, different from the original ST, we build a multi-dimension sampling space, including both depth and width, to handle general pruning cases in a fine-

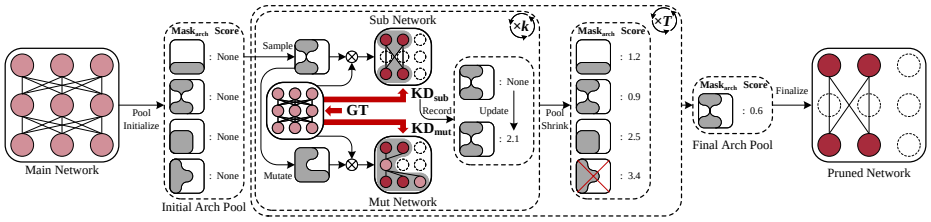


Fig. 5: The framework of the proposed stimulative training guided pruning (STP). The recursive arrows mean repeating the corresponding process multiple times; bold red arrows represent computing loss and supervising. STP gradually explores the optimal subnet architecture guided by the KD loss and utilizes stimulative training to enhance the subnet with the support of mutating networks. Finally, the subnet can be separated from the main network as a structured pruned network with low FLOPs. k is a hyper-parameter, and a typical value is the iteration number of an epoch.

grained manner, which provides more possibilities for finding a high-performance architecture. Exemplary sampled subnets are referred to in Appendix.

Besides the multi-dimension design, we impose a FLOPs-aware constraint on the sampling space, only sampling the subnets with target FLOPs. It can guarantee a more accurate practical speedup ratio compared with sparsity constraint. Because the sampled subnets generally possess much lower capacity than the main network, we adopt the normalized Kullback-Leibler divergence (KL-) [60] to eliminate the capacity gap for higher performance.
Formally, given a structured mask $\mathcal{M}_s$, the parameters of subnets are $\theta_s = \mathcal{M}_s \odot \theta_m$. ST sparsification is optimized by:

$$\mathcal{L}_{STS} = KL\_(\hat{p}(x;\theta_m)||\hat{p}(x;\theta_s)) = \sum_{i=1}^{N} \hat{p}_i(x;\theta_m) \log \frac{\hat{p}_i(x;\theta_m)}{\hat{p}_i(x;\theta_s)}. \tag{4}$$

$\hat{p}(x;\theta)$ is the normalized probability which is donated as:

$$\hat{p}_i(x;\theta) = \frac{e^{\frac{Z_i(x;\theta)}{\|Z(x;\theta)\|}}}{\sum_{j=1}^{N} e^{\frac{Z_j(x;\theta)}{\|Z(x;\theta)\|}}}, \tag{5}$$

where $Z(x;\theta)$ is the output logits. $\mathcal{L}_{STS}$ is essentially a knowledge distillation loss, and illustrated as KD_{sub} in Fig.5. The subnets will also be called target subnets in the following section.

Subnet Mutating Expansion. To mitigate the performance degradation resulting from focusing on training tiny subnets [49], we introduce additional larger subnets to provide support during training. As a distinction, we refer to them as support subnets. Considering that randomly sampling larger support subnets will disrupt the parameter magnitude concentration of target subnets, we propose to mutate and expand the width and depth of target subnets to generate support subnets.

Formally, given a target subnet with layer-wise sparsity ratio $S_{tar} = \{S^1_{tar}, S^2_{tar}, ..., S^L_{tar}\}$, for each S^i_{tar} we uniformly sample a larger ratio S^i_{sup} which satisfies:

Table 1: Comparison of different pruning methods on CIFAR-100. We report the Top-1 accuracy(%) of dense and pruned networks with different remaining FLOPs. "Acc" means Top-1 accuracy(%).

Method	ResNet-50 (Acc: 78.14)			MBV3 (Acc: 78.09)			WRN28-10 (Acc: 82.17)		
	15%	35%	55%	15%	35%	55%	15%	35%	55%
RST-S [4]	75.02	76.38	76.48	72.90	76.78	77.30	78.56	81.18	82.19
Group-SL [16]	49.04	77.90	78.37	1.43	4.90	26.24	42.41	67.71	79.59
OTOv2 [10]	77.04	77.65	78.35	76.29	77.35	78.39	77.26	80.61	80.84
Refill [8]	75.12	77.43	78.19	69.57	75.91	76.96	75.98	79.25	79.56
Ours	**79.64**	**80.0**	**80.41**	**76.81**	**78.57**	**79.09**	**80.12**	**81.63**	**82.72**

$$S_{sup} = \{S^1_{sup}, S^2_{sup}, ..., S^L_{sup}\}, S^i_{sup} \sim U(S^i_{tar}, 1), \tag{6}$$

where $U(\cdot)$ denotes the uniform distribution. The support subnet will be supervised by the main network similar to Formulation 4, denoted as:

$$\mathcal{L}_{SME} = KL_{-}(\hat{p}(x;\theta_m)||\hat{p}(x;\theta_{sup})), \tag{7}$$

where θ_{sup} is the parameters of the support subnet and $\mathcal{L}_{SME}$ is illustrated as KD_{mut} in Fig. 5. Formulation 6 can guarantee that the support subnet contains the whole parameters of the sampled target subnet. Thus, training the support subnet also enhances the target subnet.

Knowledge Distillation guided Exploration. To explore the superior architecture of target subnets, we construct a candidate pool containing numerous architectures satisfying the FLOPs target. KD loss (L_{STS}) is a reliable indicator to evaluate the generation and performance gap between subnets and the main network. Under the constraint of FLOPs, a lower KD loss implies a more superior architecture. Thus, we heuristically utilize KD loss to guide the gradual shrinking of the pool and obtain a high-performance architecture.

Specifically, suppose there is architecture pool with N candidate masks, denoted as $\mathcal{P} = \{\mathcal{M}_1, \mathcal{M}_2, ..., \mathcal{M}_N\}$. Each architecture has a corresponding score, denoted as $Sco = \{Sco_1, Sco_2, ..., Sco_N\}$. While training a target subnet with $\mathcal{M}_i$, we record the $\mathcal{L}_{STS}$ and update the corresponding score via exponential moving average (EMA) [21], which can be denoted as:

$$Sco_i := (1 - \alpha Sco_i) + \alpha \mathcal{L}_{STS}, \tag{8}$$

where $\alpha \in (0, 1)$ is the updating ratio. The pool is shrunk by removing the architectures with the highest scores every k iterations. After T rounds of shrinking, the pool contains a single architecture, which is chosen as the pruned network.

4 Experiment

In this section, we first validate the effectiveness of the proposed STP on three benchmarks: CIFAR-100 [32], Tiny ImagneNet [12] and ImageNet [12]. For CIFAR100 and Tiny ImageNet, three typical CNNs, including ResNet-50 [23], MobileNetV3 (MBV3) [28] and WRN28-10 [67], are evaluated under different pruning ratios, *i.e.*, 15%, 35% and 55%. For ImageNet, we choose ResNet-50 as the

Table 2: Comparison of different pruning methods on Tiny ImageNet. We report the Top-1 accuracy(%) of dense and pruned networks with different remaining FLOPs. "Acc" means Top-1 accuracy(%).

Method	ResNet-50 (Acc: 64.28)			MBV3 (Acc: 63.91)			WRN28-10 (Acc: 61.72)		
	15%	35%	55%	15%	35%	55%	15%	35%	55%
RST-S [4]	63.03	63.24	64.78	55.13	61.26	62.76	58.03	61.41	62.12
Group-SL [16]	0.95	19.94	55.49	0.56	2.35	53.43	0.85	25.74	57.64
OTOv2 [10]	60.38	63.45	65.16	57.61	59.25	60.16	57.19	61.23	61.70
Refill [8]	61.05	64.14	65.02	53.87	61.84	62.49	56.64	61.83	62.22
Ours	**65.93**	**66.65**	**68.27**	**59.74**	**62.11**	**63.64**	**60.53**	**62.44**	**62.81**

Table 3: Verifications of transformers on CIFAR-100. We report the Top-1 accuracy(%) of dense and pruned networks with different remaining FLOPs.

Model	Dense	RST-S [4]			Ours		
		15%	35%	55%	15%	35%	55%
ViT	76.49	70.74	72.05	74.65	**71.28**	**74.49**	**76.24**
Swin	77.16	70.53	72.98	75.25	**75.63**	**76.89**	**77.14**

backbone and compare STP with various structured pruning methods on Top-1 accuracy and FLOPs. Following the benchmarking, ablation studies are conducted to verify the indispensability of each component. Finally, we analyze the KD guided exploration process and the parameter magnitude empirically to reveal the internal mechanism of STP. All ablation and investigation experiments are conducted on CIFAR-100 with ResNet-50. We adopt the standard training settings without bells and whistles. Details can be referred to in Appendix.

4.1 Image Classifcation

Results on CIFAR-100 and Tiny ImageNet. To verify the effectiveness of STP and show its robustness to different networks. We conduct experiments on CIFAR-100 and Tiny ImageNet using ResNet-50, MBV3, and WRN28-10 as backbones. For each benchmark-network pair, three different FLOPs ratios are considered, *i.e.*, 15%, 35%, and 55%. The leading methods for comparison include a slightly modified structured RST [4] (named RST-S**), Group-SL [16], OTOv2 [10], and Refill [8]. For fair comparisons, we utilize consistent basic training settings for STP and the others. The results are shown in Table 1 and Table 2. Compared with the others, STP consistently achieves state-of-the-art performance, especially in extremely low FLOPs. For example, when reducing the FLOPs of ResNet-50 to 15%, Group-SL suffers from a severe accuracy drop from 77.49% (35%) to 49.04% (15%). In contrast, STP still maintains high accuracy, and the accuracy gains compared with the second-best method are up to 2.6% on CIFAR-100 and 2.9% on Tiny ImageNet, respectively.

To further demonstrate the generality of STP, we apply STP on two typical Transformers, i.e., ViT [50] and Swin Transformer [39]. Similar to CNNs, we conduct experiments on CIFAR-100 w.r.t. three FLOPs targets, including 15%, 35%, and 55%. The results are shown in Table 3. It can be observed that STP is

**The original RST does not support a structured ticket, we keep its regular training loss unaltered while constraining the randomly picked subnetwork to be structured.

superior for both Transformers under all target FLOPs compared with RST-S. Moreover, under 55% FLOPs, the STP pruned Transformers can almost realize lossless compression, suffering merely 0.25 and 0.02 performance drop for ViT and Swin Transformer, respectively. Note that the proposed STP is not tailored for Transformers, however, it can still achieve competitive results under 55% FLOPs, implying its immense potential in pruning Transformers.

Table 4: Results of ResNet-50 on Imagenet. We report the Top-1 accuracy(%) of dense and pruned networks with different remaining FLOPs.

Method	Unpruned top-1 (%)	Pruned top-1 (%)	Top-1 drop (%)	FLOPs (%)	Params (%)
OTOv2 [10]	76.10	70.10	6.00	**14.50**	21.30
Refill [8]	75.84	66.83	9.01	20.00	-
Ours	76.15	**72.43**	**3.72**	15.26	18.75
MetaPruning [40]	76.60	73.40	3.20	24.39	-
Slimmable [65]	76.10	72.10	4.00	26.63	-
ThiNet [44]	72.88	68.42	4.46	28.50	33.88
OTOv2 [10]	76.10	74.30	1.80	28.70	37.40
GReg-1 [52]	76.13	73.75	2.38	32.68	-
GReg-2 [52]	76.13	73.90	2.23	32.68	-
CAIE [55]	76.13	72.39	3.74	32.90	30.76
Ours	76.15	**75.39**	**0.76**	**24.21**	35.60
CHIP [47]	76.15	75.26	0.89	37.20	43.30
OTOv2 [10]	76.10	75.20	0.90	37.30	49.60
GReg-1 [52]	76.13	74.85	1.28	39.06	-
GReg-2 [52]	76.13	74.93	1.20	39.06	-
Refill [8]	75.84	72.25	3.59	40.00	-
ThiNet [44]	72.88	71.01	1.87	44.17	48.44
GBN [63]	75.85	75.18	0.67	44.94	46.60
FPGM [24]	76.15	74.83	1.32	46.50	-
LeGR [11]	76.10	75.30	0.80	47.00	-
AutoSlim [64]	76.10	75.60	0.50	48.43	80.78
MetaPruning [40]	76.60	75.40	1.20	48.78	-
Ours	76.15	**75.68**	**0.47**	**35.00**	65.58
CAIE [55]	76.13	75.62	0.51	54.77	77.35
CHIP [47]	76.15	76.30	-0.15	55.20	59.20
Slimmable [65]	76.10	74.90	1.20	55.69	-
TAS [15]	77.46	76.20	1.26	56.50	-
SSS [30]	76.12	71.82	4.30	56.96	61.18
FPGM [24]	76.15	75.59	0.56	57.80	-
LeGR [11]	76.10	75.70	0.40	58.00	-
GBN [63]	75.88	76.19	-0.31	59.46	68.17
Refill [8]	75.84	74.46	1.38	60.00	-
ThiNet [44]	72.88	72.04	0.84	63.21	66.28
GReg-1 [52]	76.13	76.27	-0.14	67.11	-
MetaPruning [40]	76.60	76.20	0.40	73.17	-
Ours	76.15	**77.13**	**-0.98**	**53.49**	84.08
SSS [30]	76.12	75.44	0.68	84.94	99.22
Ours	76.15	**77.48**	**-1.33**	**75.00**	95.99

Results on ImageNet. We also validate STP on the widely used ImageNet-1K benchmark, reporting the performance of STP-pruned ResNet-50 under various FLOPs ranging from 15% to 75% and comparing them with the ones pruned by other methods. As shown in Table. 4, under nearly the same FLOPs, the network pruned by STP consistently achieves higher accuracy than the ones pruned by other methods, demonstrating the superiority of STP. Considering extremely

low FLOPs interval (from 10% to 20%), few methods (STP, OTOv2, and Refill) attempt to prune ResNet-50 to this interval. Among these methods, our STP achieves the highest 72.43% top-1 accuracy, overpassing OTOv2 by 2.33% while remaining almost the same FLOPs (around 15%).

4.2 Ablation Studies

We eliminate the proposed components of STP, *i.e.*, ST relative sparsification, subnet mutating expansion, KD guided architecture exploration and multi-dimension sampling, to validate the single and joint effect of them. The results are shown in Table 5. Firstly, we conduct experiments with the separate removal of each component except the ST relative sparsification and observe different degrees of performance degradation, which demonstrates their indispensability. Then, we remove these components step-by-step and observe consistent performance degradation, which demonstrates their joint effectiveness. Note that, to remove the ST relative sparsification, we replace it with the L_2 regularization, which causes significant performance damage (-2.7% Top-1 accuracy), demonstrating the superiority of the ST relative sparsification.

Table 5: Influence of different components in stimulative training guided pruning.

ST relative sparsification	Subnet mutating expansion	KD guided architecture exploration	Multi-dimension sampling	Top-1 Acc(%)
✓	✓	✓	✓	**79.64**
✓	×	✓	✓	78.12
✓	✓	×	✓	78.47
✓	✓	✓	×	78.23
✓	✓	×	×	77.97
✓	×	×	×	77.46
×	×	×	×	74.76

4.3 Downstream Tasks

To validate the practical capacity of STP, we conduct experiments on downstream tasks, including object detection and instance segmentation. Specifically, we apply STP on ResNet-50 in ImageNet to generate pretrained pruned networks with three remaining FLOPs, i.e., 15%, 35%, 55%. These pretrained pruned networks are utilized as the backbone in Mask R-CNN [22], which is a multi-task framework. Then, the whole Mask R-CNN is fine-tuned and trained on COCO2017 [37] under the supervision of object detection and instance segmentation. For the fine-tuning settings, we follow the standard 1x training schedule [6] and adopt AdamW [42] to train the Mask R-CNN for 12 epochs with weight decay 0.1. The batch size is 16 and the initial learning rate is 0.0001. The learning rate decays 10% at the 8-th and the 12-th epoch. We report the mean average precision (mAP) under different intersection over union (IoU) of the bounding boxe and mask as the evaluation metrics of object detection and instance segmentation, respectively.

The results are shown in Table 6. It can be observed that STP consistently surpasses GReg [52] on both detection and segmentation. Under relatively

Table 6: The results of downstream tasks, i.e., object detection and instance segmentation, on COCO2017. The pretrained pruned ResNet-50 with different remaining FLOPs are utilized as the backbone in Mask R-CNN and fine-tuned on different tasks.

Method	ImageNet Acc (%)	Det mAP	Det mAP@0.75	Det mAP@0.5	Seg mAP	Seg mAP@0.75	Seg mAP@0.5
Baseline (100% FLOPs)	76.15	37.8	40.8	58.6	34.5	36.7	55.4
GReg-2 [52] (40% FLOPs)	74.93	36.6	39.6	57.4	33.9	36.2	54.4
GReg-1 [52] (67% FLOPs)	76.27	37.6	40.5	58.7	34.6	36.4	55.3
Ours (15% FLOPs)	72.43	35.0	37.7	55.6	32.5	34.6	52.8
Ours (35% FLOPs)	75.68	37.4	40.7	58.1	34.5	36.6	55.2
Ours (55% FLOPs)	**77.13**	**38.1**	**41.1**	**59.0**	**35.1**	**37.6**	**55.8**

low FLOPs (35%), STP causes marginal performance reduction (-0.1% on Det mAP@0.75 and Seg mAP@0.75) on downstream tasks. In the case of extremely low FLOPs(15%), STP can maintain the basic performance and does not cause unbearable performance degradation. It is worth noting that STP can improve performance (+0.3% on Det mAP@0.75 and +0.9% on Seg mAP@0.75) with 1.8x speed up (55% remaining FLOPs). We consider the reason is that the ST guided training framework can drive the pruned network to extract more representative features for downstream tasks. The above observations demonstrate that STP can be simply and practically deployed in different tasks and obtain competitive performance with various target FLOPs, which is friendly for resource-limited edge devices. Besides, the performance improvement suggests the potential of STP to improve the Pareto frontier between FLOPs and performance.

4.4 Evaluation of Architecture Exploration

To verify that our architecture exploration strategy can gradually explore the subnets with outstanding performance, we record the pool in different training epochs. We evaluate the effectiveness from two aspects: 1) performance: the average performance of architectures in the pool 2) clustering: the average sum of squared error (SSE) distance between the clustering center and architectures in the pool, where the higher SSE means lower clustering. Notely, for estimating the clustering of architectures, we convert the normalized sparsity ratio of each layer and depth into a vector and regard it as a proxy of architectures. Due to the unbearable computational overhead of fully training and measuring each subnet in the initialized pool(containing more than 1000 subnets), we conduct another experiment that utilizes the original ST while uniformly sampling the whole pool. After ST, we evaluate all subnets in the initialized pool on the test dataset as the proxy (or a look table) of the architectures' performance.

The results are shown in Fig. 6. At the early stage of STP, the randomly initialized pool contains various architectures with low average performance and markedly different structures. As training progresses, the average performance is gradually raised, and at the same time, the similarity of structures also increases. It demonstrates that under the constraint of FLOPs, our strategy can progressively drive the architecture exploration to converge to an optimal region and finally obtain a high-performance architecture for sparsification.

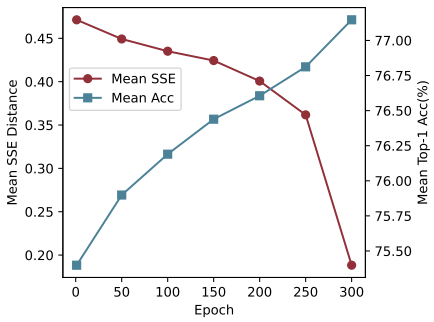


Fig. 6: Evaluation of KD guided architecture exploration. The mean sum of suqred error (SSE) distance measures the divergence of architectures pool, and lower SSE means higher clustering. During the training, such exploration progressively concentrates on high-performance architectures.

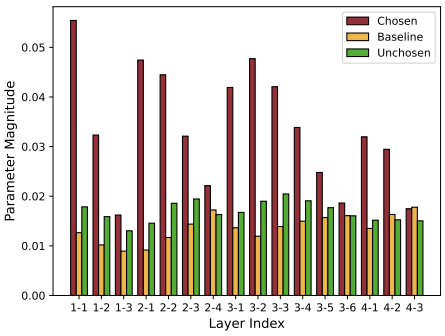


Fig. 7: The parameter magnitude of the main network trained by STP. The horizontal axis is the layer index like Fig. 4. It empirically verifies that STP enhances the chosen (kept) parameters with a relative sparsity effect while maintaining the magnitude of unchosen (dropped) ones.

4.5 Parameter Magnitude Analysis

To verify the relative sparsification effect of STP, we record the average magnitude of the chosen (kept), unchosen (dropped), and baseline (standard training without sparsification) parameters in different convolutional layers of ResNet-50. As illustrated in Fig. 7, the magnitude of chosen parameters is remarkably and consistently higher than the unchosen part in the front layers, while the unchosen part has a similar magnitude as the baseline. It demonstrates that STP achieves the evident relative sparsity effect like the original ST without suppressing the dropped part. Due to the subnet mutating expansion, the unchosen part has small parability of being enhanced, thus, the magnitude of unchosen part is marginally higher than the baseline. There is another phenomenon that in the later layer of stages, the relative sparsification effect becomes less pronounced. That is because during training, for low FLOPs targets, e.g., 15%, the later layers are dropped with a greater probability and sparsified less. Combining with the empirical results in Table 1, we conclude that STP can obtain a sparsity parameter distribution for pruning with less performance degradation.

5 Conclusion

Suppressed sparsification paradigm has been widely utilized in sparsification-based pruning, but suffers from capacity damage before pruning. In this manuscript, we reveal the relative sparsity effect in ST. Based on it, we propose an enhanced sparsification paradigm for structured pruning. Besides, we propose the multi-dimension sampling space, subnet mutating extension, and architecture pool exploration strategy, which formulate a one-stage pruning framework, named stimulative training guided pruning (STP), which can obtain a compact network with extremely low FLOPs and high performance. Comprehensive experiments on mainstream datasets and networks verify the effectiveness of STP.

Acknowledgments

This work is supported by National Key Research and Development Program of China (No. 2022ZD0160101), National Natural Science Foundation of China (No. 62071127, and 62101137), Shanghai Natural Science Foundation (No. 23ZR1402900), Shanghai Municipal Science and Technology Major Project (No.2021SHZDZX0103). The computations in this research were performed using the CFFF platform of Fudan University.

References

1. Aflalo, Y., Noy, A., Lin, M., Friedman, I., Zelnik, L.: Knapsack pruning with inner distillation. arXiv preprint arXiv:2002.08258 (2020)
2. Aghli, N., Ribeiro, E.: Combining weight pruning and knowledge distillation for cnn compression. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. pp. 3191–3198 (2021)
3. Alvarez, J.M., Salzmann, M.: Learning the number of neurons in deep networks. In: Lee, D., Sugiyama, M., Luxburg, U., Guyon, I., Garnett, R. (eds.) Advances in Neural Information Processing Systems. vol. 29. Curran Associates, Inc. (2016)
4. Bai, Y., Wang, H., TAO, Z., Li, K., Fu, Y.: Dual lottery ticket hypothesis. In: International Conference on Learning Representations (2021)
5. Blalock, D., Gonzalez Ortiz, J.J., Frankle, J., Guttag, J.: What is the state of neural network pruning? Proceedings of machine learning and systems **2**, 129–146 (2020)
6. Chen, K., Wang, J., Pang, J., Cao, Y., Xiong, Y., Li, X., Sun, S., Feng, W., Liu, Z., Xu, J., et al.: Mmdetection: Open mmlab detection toolbox and benchmark. arXiv preprint arXiv:1906.07155 (2019)
7. Chen, L., Chen, Y., Xi, J., Le, X.: Knowledge from the original network: restore a better pruned network with knowledge distillation. Complex & Intelligent Systems pp. 1–10 (2021)
8. Chen, T., Chen, X., Ma, X., Wang, Y., Wang, Z.: Coarsening the granularity: Towards structurally sparse lottery tickets. In: International Conference on Machine Learning. pp. 3025–3039. PMLR (2022)
9. Chen, T., Cheng, Y., Gan, Z., Yuan, L., Zhang, L., Wang, Z.: Chasing sparsity in vision transformers: An end-to-end exploration. Advances in Neural Information Processing Systems **34**, 19974–19988 (2021)
10. Chen, T., Liang, L., Tianyu, D., Zhu, Z., Zharkov, I.: Otov2: Automatic, generic, user-friendly. In: International Conference on Learning Representations (2023)
11. Chin, T.W., Ding, R., Zhang, C., Marculescu, D.: Towards efficient model compression via learned global ranking. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. pp. 1518–1528 (2020)
12. Deng, J., Dong, W., Socher, R., Li, L.J., Li, K., Fei-Fei, L.: Imagenet: A large-scale hierarchical image database. In: 2009 IEEE conference on computer vision and pattern recognition. pp. 248–255. Ieee (2009)
13. Ding, X., Ding, G., Han, J., Tang, S.: Auto-balanced filter pruning for efficient convolutional neural networks. In: Proceedings of the AAAI Conference on Artificial Intelligence. vol. 32 (2018)

14. Ding, X., Hao, T., Tan, J., Liu, J., Han, J., Guo, Y., Ding, G.: Resrep: Lossless cnn pruning via decoupling remembering and forgetting. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 4510–4520 (2021)
15. Dong, X., Yang, Y.: Network pruning via transformable architecture search. Advances in Neural Information Processing Systems **32** (2019)
16. Fang, G., Ma, X., Song, M., Mi, M.B., Wang, X.: Depgraph: Towards any structural pruning. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 16091–16101 (2023)
17. Frankle, J., Carbin, M.: The lottery ticket hypothesis: Finding sparse, trainable neural networks. In: International Conference on Learning Representations (2018)
18. Han, S., Mao, H., Dally, W.J.: Deep compression: Compressing deep neural networks with pruning, trained quantization and huffman coding. arXiv preprint arXiv:1510.00149 (2015)
19. Han, S., Pool, J., Tran, J., Dally, W.: Learning both weights and connections for efficient neural network. Advances in neural information processing systems **28** (2015)
20. Hassibi, B., Stork, D.: Second order derivatives for network pruning: Optimal brain surgeon. Advances in neural information processing systems **5** (1992)
21. He, K., Fan, H., Wu, Y., Xie, S., Girshick, R.: Momentum contrast for unsupervised visual representation learning. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. pp. 9729–9738 (2020)
22. He, K., Gkioxari, G., Dollár, P., Girshick, R.: Mask r-cnn. In: Proceedings of the IEEE international conference on computer vision. pp. 2961–2969 (2017)
23. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 770–778 (2016)
24. He, Y., Liu, P., Wang, Z., Hu, Z., Yang, Y.: Filter pruning via geometric median for deep convolutional neural networks acceleration. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. pp. 4340–4349 (2019)
25. He, Y., Zhang, X., Sun, J.: Channel pruning for accelerating very deep neural networks. In: Proceedings of the IEEE international conference on computer vision. pp. 1389–1397 (2017)
26. Hinton, G., Vinyals, O., Dean, J.: Distilling the knowledge in a neural network. stat **1050**, 9 (2015)
27. Hou, Z., Qin, M., Sun, F., Ma, X., Yuan, K., Xu, Y., Chen, Y.K., Jin, R., Xie, Y., Kung, S.Y.: Chex: Channel exploration for cnn model compression. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 12287–12298 (2022)
28. Howard, A., Sandler, M., Chu, G., Chen, L.C., Chen, B., Tan, M., Wang, W., Zhu, Y., Pang, R., Vasudevan, V., et al.: Searching for mobilenetv3. In: Proceedings of the IEEE/CVF international conference on computer vision. pp. 1314–1324 (2019)
29. Hu, H., Peng, R., Tai, Y.W., Tang, C.K.: Network trimming: A data-driven neuron pruning approach towards efficient deep architectures. arXiv preprint arXiv:1607.03250 (2016)
30. Huang, Z., Wang, N.: Data-driven sparse structure selection for deep neural networks. In: Proceedings of the European conference on computer vision (ECCV). pp. 304–320 (2018)
31. Kim, K., Ji, B., Yoon, D., Hwang, S.: Self-knowledge distillation with progressive refinement of targets. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 6567–6576 (2021)

32. Krizhevsky, A., Hinton, G., et al.: Learning multiple layers of features from tiny images (2009)
33. LeCun, Y., Denker, J., Solla, S.: Optimal brain damage. Advances in neural information processing systems **2** (1989)
34. Li, H., Asim, K., Igor, D., Hanan, S., Hans, P.G.: Pruning filters for efficient convnets. In: 5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings (2017)
35. Li, T., Li, J., Liu, Z., Zhang, C.: Few sample knowledge distillation for efficient network compression. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 14639–14647 (2020)
36. Lin, S., Ji, R., Li, Y., Deng, C., Li, X.: Toward compact convnets via structure-sparsity regularized filter pruning. IEEE transactions on neural networks and learning systems **31**(2), 574–588 (2019)
37. Lin, T.Y., Maire, M., Belongie, S., Hays, J., Perona, P., Ramanan, D., Dollár, P., Zitnick, C.L.: Microsoft coco: Common objects in context. In: Computer Vision–ECCV 2014: 13th European Conference, Zurich, Switzerland, September 6-12, 2014, Proceedings, Part V 13. pp. 740–755. Springer (2014)
38. Liu, B., Wang, M., Foroosh, H., Tappen, M., Pensky, M.: Sparse convolutional neural networks. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 806–814 (2015)
39. Liu, Z., Lin, Y., Cao, Y., Hu, H., Wei, Y., Zhang, Z., Lin, S., Guo, B.: Swin transformer: Hierarchical vision transformer using shifted windows. In: Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV) (2021)
40. Liu, Z., Mu, H., Zhang, X., Guo, Z., Yang, X., Cheng, K.T., Sun, J.: Metapruning: Meta learning for automatic neural network channel pruning. In: Proceedings of the IEEE/CVF international conference on computer vision. pp. 3296–3305 (2019)
41. Liu, Z., Li, J., Shen, Z., Huang, G., Yan, S., Zhang, C.: Learning efficient convolutional networks through network slimming. In: Proceedings of the IEEE international conference on computer vision. pp. 2736–2744 (2017)
42. Loshchilov, I., Hutter, F.: Decoupled weight decay regularization (2019)
43. Louizos, C., Welling, M., Kingma, D.P.: Learning sparse neural networks through l_0 regularization. In: International Conference on Learning Representations (2018)
44. Luo, J.H., Wu, J., Lin, W.: Thinet: A filter level pruning method for deep neural network compression. In: Proceedings of the IEEE international conference on computer vision. pp. 5058–5066 (2017)
45. Ma, H., Chen, T., Hu, T.K., You, C., Xie, X., Wang, Z.: Good students play big lottery better. arXiv preprint arXiv:2101.03255 **3** (2021)
46. Neill, J.O., Dutta, S., Assem, H.: Deep neural compression via concurrent pruning and self-distillation. arXiv preprint arXiv:2109.15014 (2021)
47. Sui, Y., Yin, M., Xie, Y., Phan, H., Aliari Zonouz, S., Yuan, B.: Chip: Channel independence-based pruning for compact neural networks. Advances in Neural Information Processing Systems **34**, 24604–24616 (2021)
48. Sun, T., Ding, S., Guo, L.: Low-degree term first in resnet, its variants and the whole neural network family. Neural Networks **148**, 155–165 (2022)
49. Tang, S., Ye, P., Li, B., Lin, W., Chen, T., He, T., Yu, C., Ouyang, W.: Boosting residual networks with group knowledge. arXiv preprint arXiv:2308.13772 (2023)
50. Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A.N., Kaiser, Ł., Polosukhin, I.: Attention is all you need. Advances in neural information processing systems **30** (2017)

51. Veit, A., Wilber, M.J., Belongie, S.: Residual networks behave like ensembles of relatively shallow networks. Advances in neural information processing systems **29** (2016)
52. Wang, H., Qin, C., Zhang, Y., Fu, Y.: Neural pruning via growing regularization. In: International Conference on Learning Representations (ICLR) (2021)
53. Wang, H., Zhang, Q., Wang, Y., Yu, L., Hu, H.: Structured pruning for efficient convnets via incremental regularization. In: 2019 International Joint Conference on Neural Networks (IJCNN). pp. 1–8. IEEE (2019)
54. Wen, W., Wu, C., Wang, Y., Chen, Y., Li, H.: Learning structured sparsity in deep neural networks. Advances in neural information processing systems **29** (2016)
55. Wu, Y.C., Liu, C.T., Chen, B.Y., Chien, S.Y.: Constraint-aware importance estimation for global filter pruning under multiple resource constraints. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops. pp. 686–687 (2020)
56. Xia, B., He, J., Zhang, Y., Wang, Y., Tian, Y., Yang, W., Van Gool, L.: Structured sparsity learning for efficient video super-resolution. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 22638–22647 (2023)
57. Xia, M., Zhong, Z., Chen, D.: Structured pruning learns compact and accurate models. arXiv preprint arXiv:2204.00408 (2022)
58. Yang, H., Wen, W., Li, H.: Deephoyer: Learning sparser neural network with differentiable scale-invariant sparsity measures. In: International Conference on Learning Representations (2019)
59. Ye, H., Zhang, B., Chen, T., Fan, J., Wang, B.: Performance-aware approximation of global channel pruning for multitask cnns. IEEE Transactions on Pattern Analysis and Machine Intelligence **45**(8), 10267–10284 (2023)
60. Ye, P., He, T., Tang, S., Li, B., Chen, T., Bai, L., Ouyang, W.: Stimulative training++: Go beyond the performance limits of residual networks. arXiv preprint arXiv:2305.02507 (2023)
61. Ye, P., Li, B., Li, Y., Chen, T., Fan, J., Ouyang, W.: β-darts: Beta-decay regularization for differentiable architecture search. In: 2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). pp. 10864–10873. IEEE (2022)
62. Ye, P., Tang, S., Li, B., Chen, T., Ouyang, W.: Stimulative training of residual networks: A social psychology perspective of loafing. Advances in Neural Information Processing Systems **35**, 3596–3608 (2022)
63. You, Z., Yan, K., Ye, J., Ma, M., Wang, P.: Gate decorator: Global filter pruning method for accelerating deep convolutional neural networks. Advances in neural information processing systems **32** (2019)
64. Yu, J., Huang, T.: Autoslim: Towards one-shot architecture search for channel numbers. arXiv preprint arXiv:1903.11728 (2019)
65. Yu, J., Yang, L., Xu, N., Yang, J., Huang, T.: Slimmable neural networks. arXiv preprint arXiv:1812.08928 (2018)
66. Yu, S., Chen, T., Shen, J., Yuan, H., Tan, J., Yang, S., Liu, J., Wang, Z.: Unified visual transformer compression. arXiv preprint arXiv:2203.08243 (2022)
67. Zagoruyko, S., Komodakis, N.: Wide residual networks. In: British Machine Vision Conference 2016. British Machine Vision Association (2016)