Prune Your Model Before Distill It

Jinhyuk Park[1] and Albert No[1]

Hongik University, Seoul 04066, Korea
c0292601@g.hongik.ac.kr, albertno@hongik.ac.kr

Abstract. Knowledge distillation transfers the knowledge from a cumbersome teacher to a small student. Recent results suggest that the student-friendly teacher is more appropriate to distill since it provides more transferrable knowledge. In this work, we propose the novel framework, "prune, then distill," that prunes the model first to make it more transferrable and then distill it to the student. We provide several exploratory examples where the pruned teacher teaches better than the original unpruned networks. We further show theoretically that the pruned teacher plays the role of regularizer in distillation, which reduces the generalization error. Based on this result, we propose a novel neural network compression scheme where the student network is formed based on the pruned teacher and then apply the "prune, then distill" strategy. The code is available at https://github.com/ososos888/prune-then-distill.

Keywords: Knowledge distillation, label smoothing regularization (LSR), neural network compression, pruning

1 Introduction

Recent progress in neural networks (NN) in various tasks highly depends on its over-parameterization, such as classification [21, 55], language understanding [7, 8], and self-supervised learning [3, 15]. This leads to extensive computational cost and even causes environmental issues [38]. Therefore, neural network compression techniques have received increasing attention, such as knowledge distillation [20, 40, 51] and pruning [9, 13, 26, 31].

Knowledge distillation (KD) [20] is a model compression tool that transfers the features from a cumbersome network to a smaller network. At first glance, a powerful teacher with higher accuracy may show better distillation results; however, Cho and Hariharan [4] showed that the less-trained teacher teaches better when the student network does not have enough capability. Lately, a line of works has proposed distillation schemes that focus on a "student-friendly" teacher, which provides more transferrable knowledge to the student network with limited capacity [36, 37].

On the other hand, network pruning [26] is another network compression technique that effectively removes networks' weights or neurons while maintaining accuracy. Since pruning simplifies the neural network, we naturally conjecture that the pruned teacher provides student-friendly knowledge that is easier

to transfer. This intuition leads us to our main question: *can pruning boost the performance of knowledge distillation?*

To answer this question, we propose a new framework, "prune, then distill," consisting of three steps: 1) train the (teacher) network, 2) prune the (teacher) network, and 3) distill the pruned network to the smaller (student) network. We examine several simple experiments to verify the proposed idea that compares the test accuracy of student networks with and without (unstructured) pruning on the teachers' side. More precisely, We conduct three experiments: 1) distill VGG19 [43] to VGG11, 2) distill VGG19 and ResNet18 [16] to itself (self distillation), and 3) distill ResNet18 to VGG16 and MobileNetV2 [42]. In all three cases, we observe that the student learned from the pruned teacher generally outperforms the student learned from an unpruned teacher.

We then provide theoretical support to answer why the pruned teacher is better in distillation. Knowledge distillation can be viewed as a label smoothing regularization (LSR) [54,58], which regularizes training by providing a smoother label. We find that a teacher trained with regularization provides a smoother label than the original teacher. This implies that the distillation with a regularized teacher is equivalent to LSR with smoother labels. Since pruning can be viewed as a regularized model with a sparsity-inducing regularizer [28], we conclude that the pruned teacher regularizes the distillation process.

Based on the observation that pruned teacher provides a better knowledge in distillation, we then suggest a novel network compression scheme. When a cumbersome network is provided, we want to compress the network by applying the "prune, then distill" strategy. However, since the distillation transfers knowledge to a *given* student network, the student network architecture design is required. The main idea of student network construction is matching the teacher and the student layerwise. We propose a student network with the same depth but fewer neurons so that the number of weights per layer matches the number of nonzero weights of the pruned network in the corresponding layer. We evaluate the proposed compression scheme with extensive experiments.

We summarize our contributions as:

- We propose a novel framework, "prune, then distill," that prunes teacher networks before distillation.
- We examine experiments that verify unstructured pruning on the teacher can boost the performance of knowledge distillation.
- We also provide a theoretical analysis that the distillation from a pruned teacher is effectively a label smoothing regularization with smoother labels.
- We propose a novel network compression that constructs the student network based on the pruned teacher, then apply the "prune, then distill" strategy.

2 Related Works

This section is devoted to prior works on neural network (NN) compression that are related to our work. In particular, we focus on knowledge distillation and

network pruning. Note that there are other NN compression techniques such as quantization [2, 29], coding [14, 50], and matrix factorization [22, 41].

2.1 Knowledge Distillation

Knowledge distillation (KD) [20] transfers the knowledge from the strong teacher network to a smaller student network. The student network is trained with soft targets provided by the teacher network and some intermediate features [40, 52, 56]. There are variations of KD such as KD using GAN [51], Jacobian matching KD [5, 44], distillation of activation boundaries [19], contrastive distillation [48], and distillation from graph neural networks [23, 52].

Recently, many works have reported that the large gap between student and teacher causes degradation in student network performance [36]. Cho and Hariharan showed that the less-trained network transfers better knowledge to a small network [4]. Park et al. [37] proposed a student-aware teacher learning to transfer the teacher's knowledge effectively. In this paper, we provide an extremely simple way to generate a student-friendly teacher using unstructured pruning.

2.2 Pruning

There are two main branches of pruning: 1) unstructured pruning, which prunes individual weights, and 2) structured pruning, which prunes neurons (in most cases, channels of convolutional neural networks). Although both approaches share a similar idea, these two strategies have been developed independently.

Unstructured pruning: Unstructured pruning [26] removes NN components in weight-level while maintaining the number of neurons in the network. A general pruning pipeline consists of three steps: 1) train a large network, 2) prune weights (or neurons) based on its own rule, then 3) fine-tune the pruned model. The iterative magnitude pruning (IMP) technique, which iteratively applies magnitude-based pruning and fine-tuning, shows remarkable performance [13]. Lottery ticket rewinding (LTR), an iterative magnitude pruning method with weight rewinding, is highly successful [9, 10]. Recently, IMP with learning rate (LR) rewinding, which repeats the learning rate schedule, shows better results in bigger networks [39]. However, the network architecture after unstructured pruning remains the same (i.e., number of channels per layer). It is hard to fully enjoy the benefit of a pruned network without dedicated hardware [12].

Structured pruning: Structured pruning removes NN parameters at the level of neurons (mostly channels) [1,31,32,35,46]. It provides a smaller network with efficient network architecture, and we can save computational resources without designing dedicated hardware or libraries. Like magnitude-based unstructured pruning, the most naive method is to prune filters based on weights [17, 31]. Another approach is adding an extra regularizer that induces sparsity while training [18, 49, 57]. Liu et al. [33] and Ye et al. [53] proposed the structured pruning scheme based on batch normalization (BN) scale factor of filters. Zhuang et al. [59] adds polarization regularizer to structured pruning with BN scale

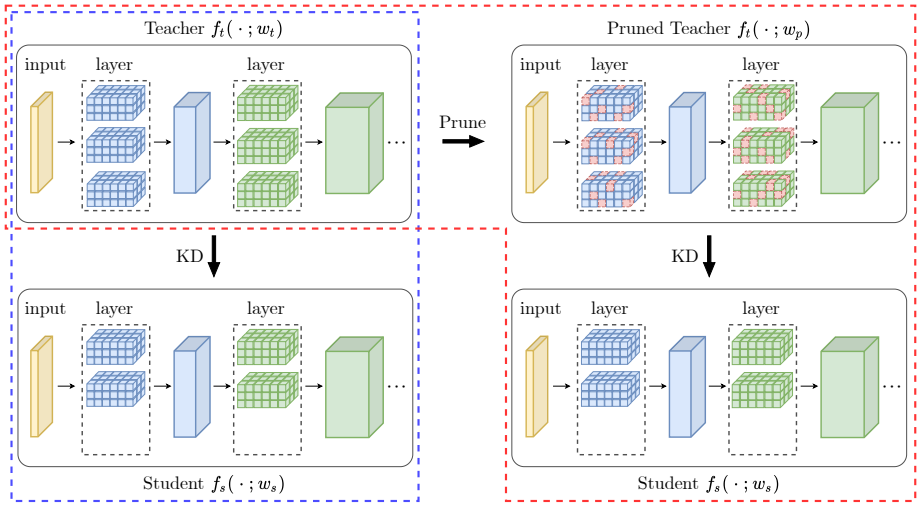

Fig. 1: Overview of the "prune, then distill" strategy. Instead of distilling directly from the teacher to the student (blue dotted box), we prune the teacher first, then distill from the pruned teacher to the student (red dotted box).

factor. However, due to the structural constraint, the pruned network has more weights (parameters) than unstructured pruning [34].

3 Prune, then Distill

3.1 Exploratory Experiments

We conduct experiments to verify the effectiveness of pruned teachers in KD. Instead of distilling the teacher network directly (dotted-blue block in Figure 1), we first (unstructured) prune the teacher network and then distill to the student network (dotted-red block in Figure 1).

Setups: We mainly considers VGG [43] and ResNet [16] for the teacher network, where VGG is trained on the CIFAR100 dataset [24] and ResNet is trained on the TinyImageNet dataset [25]. The TinyImageNet dataset is a subset of resized ($3 \times 64 \times 64$) ImageNet dataset [6]. We reserve 10% of the data as a validation set in all training. We apply unstructured pruning that removes more weights, more precisely LR rewinding [39], to prune the teacher model. In LR rewinding, we set the ratio of epochs by 0.65 for VGG-CIFAR100. In other words, we train the VGG19 for 200 epochs initially, then rewind the learning rates and retrains (fine-tuning) the network for 130 epochs (65%). Note that the different ratios from 0.6 to 0.9 do not make significant differences in pruning, and we use the ratio of 0.5 for ResNet-TinyImageNet. For a fair comparison, we train (and distill) networks with enough epochs and halt the training at their best performance

Table 1: Knowledge distillation from VGG19 to VGG11 on CIFAR100 with teacher pruning. VGG19DBL is the VGG19 with 2× more filters per layer. Teacher "None" indicates the student is trained without a teacher, while the pruning ratio "None" means the distillation from the unpruned teacher.

Teacher	Pruning Ratio	Teacher Accuracy	Student	Student Accuracy
None	-	-	VGG11	69.51 ± 0.24
VGG19	None	73.13	VGG11	72.02 ± 0.27
	36%	73.30	VGG11	72.76 ± 0.10
	59%	72.25	VGG11	72.59 ± 0.32
	79%	73.43	VGG11	72.67 ± 0.34
VGG19DBL	None	74.44	VGG11	71.81 ± 0.29
	36%	73.46	VGG11	72.01 ± 0.11
	59%	73.24	VGG11	72.40 ± 0.25
	79%	73.50	VGG11	72.48 ± 0.19

on validation dataset. All test accuracies are the average of three independent experiments, and we also provide the standard deviation.

For simplicity, we use the vanilla KD [20] with appropriate balancing parameter α and temperature τ. The balancing parameter α represents the ratio of two objectives (distill loss and hard-target loss). The temperature τ is a softening parameter, where higher τ produces a softer target. In the experiment, we fix the parameters by $\alpha = 0.95$ and $\tau = 10$. More detailed training parameters are provided in Appendix. Note that the purpose of experiments is not achieving the best test accuracy but to compare between *distilling from a pruned network* and *distilling from an unpruned network*. Thus, hyperparameters, as well as network architectures, are not optimized for test accuracies. Instead, we use as-is settings for a *fair* comparison between the pruned teacher and the unpruned teacher. For example, we follow default settings for MobileNetV2 [42] and ResNet18 [16] optimized for ImageNet dataset [6], while we use TinyImageNet dataset [25].

Distill VGG19 to VGG11: We set VGG19 [43] as a teacher network and VGG11 as a student network. The network architecture of VGG is unchanged except for the number of fully connected (FC) layers, where our VGG has a single FC layer (which is commonly used for CIFAR10 data). Then, we compare the KD results on the CIFAR100 dataset [24] between the regular VGG19 teacher and the pruned VGG19 teacher. We prune the teacher network with three sparsity levels: 36% sparsity (36% of weights are removed), 59% sparsity, and 79% sparsity.

Surprisingly, as shown in Table 1, VGG11 with pruned VGG19 consistently outperforms the one with the unpruned teacher. Table 1 also provides results when the teacher network is VGG19DBL, with 2× many channels in each layer. In both cases, the pruned teacher shows better performance.

Self distillation: Motivated by [11, 54], we conduct the self distillation experiment, where the teacher and the student share the same model. We con-

Table 2: Self distillation of VGG19 and ResNet18 with teacher pruning. DBL model has the same model structure with 2× more filters per layer. Teacher "None" indicates the student is trained without a teacher, while the pruning ratio "None" means the distillation from the unpruned teacher.

Teacher	Pruning Ratio	Teacher Accuracy	Student	Student Accuracy
None	-	-	VGG19	72.76 ± 0.33
VGG19	None	73.13	VGG19	73.74 ± 0.20
	36%	73.30	VGG19	74.10 ± 0.26
	59%	72.25	VGG19	74.26 ± 0.37
	79%	73.43	VGG19	74.35 ± 0.10
None	-	-	VGG19DBL	74.62 ± 0.21
VGG19DBL	None	74.44	VGG19DBL	74.78 ± 0.37
	36%	73.46	VGG19DBL	75.16 ± 0.44
	59%	73.24	VGG19DBL	75.26 ± 0.77
	79%	73.50	VGG19DBL	75.05 ± 0.92
None	-	-	ResNet18	57.75 ± 0.24
ResNet18	None	57.75	ResNet18	57.97 ± 0.10
	36%	57.66	ResNet18	59.39 ± 0.21
	59%	57.58	ResNet18	58.99 ± 0.26
	79%	57.32	ResNet18	59.33 ± 0.18
None	-	-	ResNet18DBL	60.21 ± 0.24
ResNet18DBL	None	60.46	ResNet18DBL	61.35 ± 0.02
	36%	61.97	ResNet18DBL	63.03 ± 0.38
	59%	61.80	ResNet18DBL	63.19 ± 0.21
	79%	61.66	ResNet18DBL	63.16 ± 0.02

sider VGG19 and VGG19DBL with CIFAR100 dataset, where ResNet18 and ResNet18DBL are trained on the TinyImageNet dataset. Table 2 shows the test accuracies of 1) the model without KD, 2) the model learned from the unpruned teacher, and 3) the model learned from the pruned teacher. Similar to other experiments, we also observe the consistent result where the pruned model teaches better than the unpruned teacher. Note that learning from unpruned network also increases the test accuracy (compared to the one without a teacher); however, the gain with the pruned teacher is more significant.

Distill ResNet18 to VGG and MobileNet: We also investigate the KD from the pruned teacher when the student and the teacher have different network architectures. Specifically, we consider the TinyImageNet dataset, where the teacher is ResNet18 and students are VGG16 and MobileNetV2 [42]. Table 3 compares the test accuracies of 1) student without a teacher, 2) student learned from the unpruned teacher, and 3) student learned from the pruned teacher. Consistently, we observe the better KD performance when the teacher is pruned. This implies that the better distillation is not limited to the case of the similar architecture between teacher and student networks.

Table 3: Distillation from ResNet18 to MobileNetV2 and VGG16 with teacher pruning. Teacher "None" indicates the student is trained without a teacher, while the pruning ratio "None" means the distillation from the unpruned teacher.

Teacher	Pruning Ratio	Teacher Accuracy	Student	Student Accuracy
None	-	-	VGG16	53.31 ± 0.45
ResNet18	None	57.75	VGG16	54.75 ± 0.29
	36%	57.66	VGG16	56.35 ± 0.35
	59%	57.58	VGG16	55.86 ± 0.04
	79%	57.32	VGG16	56.49 ± 0.15
None	-	-	MobileNetV2	50.79 ± 0.44
ResNet18	None	57.75	MobileNetV2	56.10 ± 0.23
	36%	57.66	MobileNetV2	56.73 ± 0.24
	59%	57.58	MobileNetV2	56.73 ± 0.43
	79%	57.32	MobileNetV2	57.20 ± 0.25

Remark: One might suspect that better distillation result is due to higher accuracy of the teacher, where the pruned model often achieves better accuracy [9]. However, the higher accuracy of the teacher network does not guarantee better results in distillation [45]. Also, the pruned teacher works better even when test accuracy is lower than the unpruned teacher. For example, pruning decreases the test accuracy of the teacher network in ResNet18-TinyImageNet, where we observe that the pruned teacher transfers the knowledge better. This implies that the pruned teacher is better not because it has higher accuracy, but it provides better transferable knowledge.

We also investigate the agreement between the teacher and the student's prediction (details provided in Appendix). As shown by Stanton et al. [45], we observe that the agreement and the accuracy behave independently. For example, in VGG19 self distillation experiments, the pruned teacher provides a higher agreement, and the corresponding student has a higher accuracy; however, in ResNet18 self distillation, the pruned teacher shows lower agreement although the student's accuracy is higher. It implies that some students mimic the teacher better but perform worse. This result supports our theory that distillation indirectly helps the training student models with additional regularization.

3.2 Pruned Teacher as a Regularizer

In this section, we provide a theoretical analysis on the pruned teacher in KD. We first point out that the teacher trained with a regularizer provides an additional regularization during distillation.

Let $\{(x_i, y_i)\}_{i=1}^N$ be the dataset where the label y_i takes value from the set $\{1, 2, \ldots, K\}$. We are interested in a classification model which outputs a K-dimensional probability distributions. Let $f_{true}(x_i) \in \mathbb{R}^K$ be the one-hot encoded vector where $f_{true}(x_i)[y_i] = 1$ for the ground-truth label y_i and $f_{true}(x_i)[y'] = 0$

for all $y' \neq y_i$. We further let $f_t(x; w)$ be the output of the teacher network when the input is x and the weight is w. Then, we train the teacher $f_t(\cdot; w)$ and achieve w_t that minimizes the cross entropy loss

$$L_{CE}(w) = \frac{1}{N}\sum_{i=1}^{N} H(f_{true}(x_i), f_t(x_i; w)), \tag{1}$$

where the cross-entropy loss is defined by $H(p_1, p_2) = -\sum_{k=1}^{K} p_1[k] \log p_2[k]$.

Similarly, $f_s(x; \tilde{w})$ is the output of the student network when the input is x and the weight is $\tilde{w}$. For the temperature $\tau = 1$, the knowledge distillation loss is given by

$$L_{KD}(\tilde{w}) = \frac{1}{N}\sum_{i=1}^{N}(1-\alpha)H(f_{true}(x_i), f_s(x_i; \tilde{w})) + \alpha H(f_t(x; w_t), f_s(x; \tilde{w})). \tag{2}$$

Yuan et al. [54] showed that the KD is equivalent to label smoothing regularization (LSR). More precisely, the author showed that

$$L_{KD}(\tilde{w}) = \frac{1}{N}\sum_{i=1}^{N} H(f_m^{(\alpha)}(x_i; w_t), f_s(x_i; \tilde{w})), \tag{3}$$

where $f_m^{(\alpha)}(x; w_t) = (1-\alpha) f_{true}(x) + \alpha f_t(x; w_t)$, and therefore KD is equivalent to label smoothing regularization with smoothed label distribution $f_m^{(\alpha)}(x; w_t)$.

We then consider the case where the teacher is trained with a regularizer $R(w)$. The regularized teacher $f_t(\cdot; w_p)$ is obtained by minimizing

$$L_{REG}(w) = \frac{1}{N}\sum_{i=1}^{N} H(f_{true}(x_i), f_t(x_i; w)) + R(w), \tag{4}$$

i.e., $L_{REG}(w_p) = \min_w L_{REG}(w)$. Since $L_{CE}(w_t) = \min_w L_{CE}(w)$, we have

$$\frac{1}{N}\sum_{i=1}^{N} H(f_{true}(x_i), f_t(x_i; w_t)) \leq \frac{1}{N}\sum_{i=1}^{N} H(f_{true}(x_i), f_t(x_i; w_p)) \tag{5}$$

$$\frac{1}{N}\sum_{i=1}^{N} H(f_{true}(x_i), f_t(x_i; w_p)) + R(w_p) \leq \frac{1}{N}\sum_{i=1}^{N} H(f_{true}(x_i), f_t(x_i; w_t)) + R(w_t) \tag{6}$$

which implies

$$0 \leq \frac{1}{N}\sum_{i=1}^{N} \log \frac{f_t(x_i; w_t)[y_i]}{f_t(x_i; w_p)[y_i]} \leq R(w_t) - R(w_p) \tag{7}$$

Thus, $f_t(x_i; w_t)[y_i]$ is larger than $f_t(x_i; w_p)[y_i]$ on average.

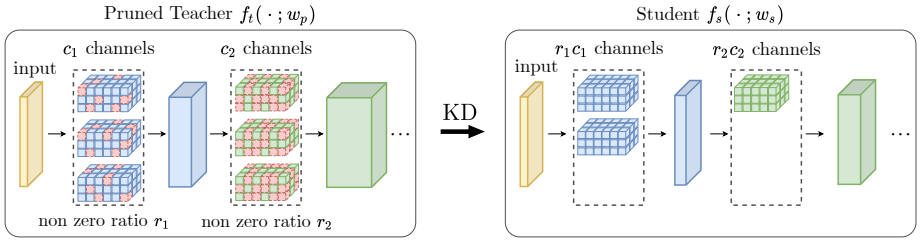

Fig. 2: Student network design. The number of channels of the student network is adjusted so that each layer's parameters match the number of nonzero parameters in each layer of the pruned teacher.

Recall that the distillation from $f_t(x_i; w_t)$ is equivalent to label smoothing regularization with smoothed label distribution $f_m^{(\alpha)}(x; w_t) = (1-\alpha)f_{true}(x) + \alpha f_t(x, w_t)$. If we distill from $f_t(x_i; w_p)$ to the student, then it is essentially label smoothing regularization with a new smoothed label distribution $f_m^{(\alpha)}(x; w_p) = (1-\alpha)f_{true}(x) + \alpha f_t(x, w_p)$. Since Eq. (7) implies that the new smoothed distribution $f_m^{(\alpha)}(x_i; w_p)$ has a smaller weight at the true label y_i on average, we can conclude that $f_m^{(\alpha)}(x_i; w_p)$ is *smoother*[1] than $f_m^{(\alpha)}(x_i; w_t)$. In other words, the regularization in teacher training also regularizes student distillation further. Note that Eq. (7) provides an upper bound of the ratio between the teacher's output and the regularized teacher's output at the true label. This effectively measures the smoothness of a smoothed label in label smoothing regularization.

The pruning can be viewed as a solution of the empirical risk minimization problem with sparsity-inducing regularization [28]. Thus, the distillation from the pruned teacher is a label smoothing regularization with smoother label distribution, which reduces a generalization error.

4 Transferring Knowledge of Sparsity

Based on the observation that the pruned teacher transfers the better knowledge, we propose a novel network compression framework that learns from the (unstructured) pruned network. The critical challenge is a student network architecture design to learn effectively from the pruned teacher.

More formally, let $f_t(\cdot; w_t)$ be a cumbersome network to compress, and the goal is to compress it to a smaller network $f_s(\cdot; w_s)$. In the previous section, we considered the distillation to a given student network. On the other hand, in this section, we provide a detailed architecture design for a student network f_s based on the pruned teacher $f_t(\cdot; w_p)$.

On top of the "prune, then distill" as described in Figure 1, we add student network architecture design. The key idea of student network design is that the

[1] Instead of label's self-entropy, we measure the smoothness with true label's weight.

pruned teacher can also provide sparsity knowledge. We construct the narrower student where each layer matches the corresponding layer of the pruned teacher. More precisely, the student network has the same depth, but the number of channels per layer is reduced so that the number of weights is (approximately) equal to the number of remaining parameters in the pruned teacher (as described in Figure 2). The intuition is to build a student network where each layer has enough capacity to learn from the pruned teacher. The rigorous construction of the student network is described in Appendix. Thus, the proposed compression algorithm has four steps:

1. Train the original network and obtain $f_t(\cdot; w_t)$.
2. Apply the unstructured pruning and obtain pruned network $f_t(\cdot; w_p)$.
3. Construct f_s based on each layer's sparsity of the pruned network $f_t(\cdot; w_p)$.
4. Distill the pruned network $f_t(\cdot; w_p)$ to the student $f_s(\cdot; w_s)$.

Note that the above framework does not depend on the specific choice of distillation or pruning method. In Section 5, we apply LR rewinding [39] to prune the model, and apply the vanilla KD [20] to distill the pruned teacher.

The proposed scheme transfers knowledge from the sparse network (from unstructured pruning) to a network with fewer channels to reduce the number of channels further. This is similar to residual distillation [30] which removes unwanted parts (residual connections) of residual networks. In our setting, we remove unwanted parts (more channels) of unstructured pruning by merging sparse filters into fewer filters via KD.

Note that our compression framework can be viewed as structured pruning since it effectively removes neurons (channels) of a given network. Since structured pruning is nearly an architecture search algorithm [34], the proposed framework suggests a novel network architecture search algorithm that learns from unstructured pruning. Recall that recent global unstructured pruning algorithms [27] (where the pruning scheme actively determines the pruning ratio for each layer) outperform precisely designed layerwise sparsity selection schemes.

5 Experiments

In this section, we present our experimental results verifying the proposed algorithm. Similar to Section 3, we compare test accuracies of three scenarios: 1) train student network without a teacher, 2) distill the pruned teacher to the student network, and 3) distill the original (unpruned) teacher to the student network. To maintain the consistency of experiments, we use the same training, pruning, and distillation procedure and the same network hyperparameters for all three scenarios (mostly from Section 3). All test accuracies are the average of three independent experiments, and we also provide the standard deviation.

5.1 Results

For the VGG-CIFAR100 experiment, we use VGG19 with batch normalization as a teacher. In the proposed framework, we apply LR rewinding to obtain

Table 4: Performance of the proposed compression algorithm on VGG19 with CIFAR100. VGG19-ST(X) is the constructed student network based on the proposed algorithm from X% pruned teacher. Teacher "None" indicates the student is trained without a teacher, while the pruning ratio "None" means the distillation from the unpruned teacher.

Teacher	Pruning Ratio	Teacher Accuracy	Student	Student Accuracy
None	-	-	VGG19-ST36	72.32 ± 0.12
VGG19	None	73.13	VGG19-ST36	73.52 ± 0.20
	36%	73.30	VGG19-ST36	73.77 ± 0.16
None	-	-	VGG19-ST59	71.80 ± 0.18
VGG19	None	73.13	VGG19-ST59	73.18 ± 0.10
	59%	72.25	VGG19-ST59	73.81 ± 0.10
None	-	-	VGG19-ST79	70.89 ± 0.14
VGG19	None	73.13	VGG19-ST79	72.42 ± 0.16
	79%	73.43	VGG19-ST79	73.39 ± 0.11
None	-	-	VGG19DBL-ST36	74.39 ± 0.02
VGG19DBL	None	74.44	VGG19DBL-ST36	74.62 ± 0.34
	36%	73.46	VGG19DBL-ST36	75.40 ± 0.18
None	-	-	VGG19DBL-ST59	74.06 ± 0.22
VGG19DBL	None	74.44	VGG19DBL-ST59	74.67 ± 0.24
	59%	73.24	VGG19DBL-ST59	75.09 ± 0.23
None	-	-	VGG19DBL-ST79	73.81 ± 0.45
VGG19DBL	None	74.44	VGG19DBL-ST79	74.16 ± 0.04
	79%	73.50	VGG19DBL-ST79	75.19 ± 0.31

the pruned VGG19s with target sparsity 36%, 59%, and 79%. The test accuracy of the pruned teacher is similar to the baseline model (VGG19) or slightly higher. We construct the student network as described in the previous section. Let VGG19-ST36, VGG19-ST59, and VGG19-ST79 denote the student networks with fewer channels that correspond to pruned teachers with pruning ratios 36%, 59%, and 79%, respectively. We also run the same experiment with VGG19DBL (with 2× more channels per layer). Similar to VGG19, let VGG19DBL-ST36, VGG19DBL-ST59, and VGG19DBL-ST79 denote student networks that correspond to pruned teachers with pruning ratios 36%, 59%, and 79%, respectively.

For the ResNet-TinyImageNet experiment, we use ResNet18 as a teacher. The base ResNet18 is an unpruned teacher model where the test accuracy is 57.75%. The pruned ResNet18 is a teacher in the proposed framework where we apply LR rewinding with target sparsity 36%, 59%, and 79%. Notably, the pruned teacher's test accuracy is lower than the unpruned network, unlike the VGG-CIFAR100 setup. Similar to VGG-CIFAR100, let ResNet18-ST36, ResNet18-ST59, and ResNet18-ST79 denote the student networks that correspond to the pruned teacher with pruning ratios 36%, 59%, and 79%, respectively.

Table 5: Performance of the proposed compression algorithm on ResNet18 with TinyImageNet. ResNet18-ST(X) is the constructed student network based on the proposed algorithm from X% pruned teacher. Teacher "None" indicates the student is trained without a teacher, while the pruning ratio "None" means the distillation from the unpruned teacher.

Teacher	Pruning Ratio	Teacher Accuracy	Student	Student Accuracy
None	-	-	ResNet18-ST36	56.44 ± 0.26
ResNet18	None	57.75	ResNet18-ST36	57.74 ± 0.22
	36%	57.66	ResNet18-ST36	58.75 ± 0.19
None	-	-	ResNet18-ST59	55.93 ± 0.32
ResNet18	None	57.75	ResNet18-ST59	56.70 ± 0.35
	59%	57.58	ResNet18-ST59	57.76 ± 0.31
None	-	-	ResNet18-ST79	54.48 ± 0.53
ResNet18	None	57.75	ResNet18-ST79	55.65 ± 0.24
	79%	57.32	ResNet18-ST79	56.23 ± 0.16
None	-	-	ResNet18DBL-ST36	59.88 ± 0.30
ResNet18DBL	None	60.46	ResNet18DBL-ST36	61.02 ± 0.15
	36%	61.97	ResNet18DBL-ST36	62.33 ± 0.21
None	-	-	ResNet18DBL-ST59	58.81 ± 0.28
ResNet18DBL	None	60.46	ResNet18DBL-ST59	60.99 ± 0.27
	59%	61.80	ResNet18DBL-ST59	62.41 ± 0.52
None	-	-	ResNet18DBL-ST79	57.79 ± 0.14
ResNet18DBL	None	60.46	ResNet18DBL-ST79	60.60 ± 0.26
	79%	61.66	ResNet18DBL-ST79	61.87 ± 0.27

Table 4 and Table 5 show the test accuracies of the student network. For comparison, we also provide test accuracies when the same student network is trained without a teacher. In all settings, the proposed scheme outperforms the student network trained from scratch by huge margin.

5.2 Ablation Study

Learning from the unpruned teacher: Table 4 and Table 5 also provide the KD result from the unpruned teacher with the same student networks. Similar to Section 3, it is consistent that the pruned teacher (with matching sparsity) provide better KD.

Alternative student network design: For VGG19(DBL) teacher, we manually designed students VGG19-CL1 and VGG19-CL2. These networks have the same depth, but the number of channels is adjusted, where the number of network parameters is (approximately) half of the original network. VGG19-CL1 removes channels uniformly across the layer, and VGG19-CL2 removes channels unevenly. The detailed network architecture is provided in Appendix.

Table 6: Knowledge distillation to manually designed student networks. VGG19DBL is the VGG19 with 2× more filters per layer. Teacher "None" indicates the student is trained without a teacher, while the pruning ratio "None" means the distillation from the unpruned teacher.

Teacher	Pruning Ratio	Teacher Accuracy	Student	Student Accuracy
None	-	-	VGG19-CL1	69.51 ± 0.24
VGG19	None	73.13	VGG19-CL1	70.47 ± 0.25
	36%	73.30	VGG19-CL1	71.52 ± 0.50
	59%	72.25	VGG19-CL1	71.43 ± 0.24
	79%	73.43	VGG19-CL1	71.82 ± 0.16
None	-	-	VGG19-CL1	69.51 ± 0.24
VGG19DBL	None	74.44	VGG19-CL1	70.38 ± 0.25
	36%	73.46	VGG19-CL1	70.84 ± 0.23
	59%	73.24	VGG19-CL1	70.52 ± 0.03
	79%	73.50	VGG19-CL1	71.00 ± 0.34
None	-	-	VGG19-CL2	71.36 ± 0.29
VGG19	None	73.13	VGG19-CL2	72.75 ± 0.60
	36%	73.30	VGG19-CL2	73.52 ± 0.22
	59%	72.25	VGG19-CL2	73.39 ± 0.21
	79%	73.43	VGG19-CL2	73.67 ± 0.09
None	-	-	VGG19-CL2	71.36 ± 0.29
VGG19DBL	None	74.44	VGG19-CL2	72.29 ± 0.12
	36%	73.46	VGG19-CL2	72.73 ± 0.41
	59%	73.24	VGG19-CL2	72.94 ± 0.37
	79%	73.50	VGG19-CL2	72.88 ± 0.20

Table 6 compares the test accuracies of student networks with pruned and unpruned teachers. The number of parameters of VGG19-CL1 and VGG19-CL2 are 11.0M and 9.9M, respectively, which are comparable to VGG19-ST59 that has 8.2M parameters (see Appendix for details). However, the test accuracy of VGG19-ST69 with the proposed framework is higher than accuracies of VGG19-CL1 and VGG19-CL2. The result justifies the proposed student network construction based on the pruned teacher.

Also, the student network with pruned teachers outperforms the student with the unpruned teacher. This implies that the surprising performance of pruned teachers does not rely on the architecture of the student. Note that VGG19DBL has better test accuracy compared to VGG19, where the margin is about 1%. There is no significant difference in test accuracy when unpruned VGG19 and unpruned VGG19DBL are being used as teacher networks in KD. However, in KD, pruned VGG19 teaches better than pruned VGG19DBL with the same sparsity. It coincides with what we observed in the previous section, where the teacher with better accuracy does not guarantee better KD.

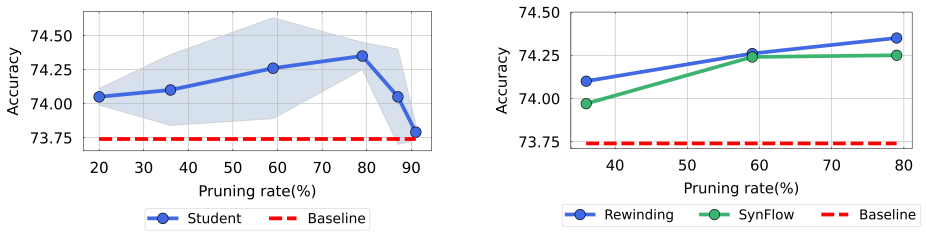

Fig. 3: Effect of pruning ratios and algorithms. The left plot shows the student's accuracies with various pruning ratios of pruned teachers. The right plot shows the student's accuracies when different pruning algorithms (LR rewinding [39] and SynFlow [47]) are applied to the teacher. In both cases, baseline is the student distilled from unpruned teacher.

5.3 Discussions

Effect of pruning ratio and pruning algorithm: Figure 3 shows the effect of pruning ratio and pruning algorithm. For VGG19 on CIFAR100, we apply the proposed scheme with additional pruning ratio 20%, 87%, and 91%. In the current setting, the 79% point is the optimal pruning ratio, and the student's performance is degraded if the pruning ratio is too high. We also applied another pruning algorithm, SynFlow [47]. Our result shows that the effectiveness of proposed compression scheme does not depend on the choice of pruning algorithm.
Large Scale Experiments: We also applied the proposed idea to the larger model (ResNet50) and the larger dataset (ImageNet). We consistently observe that the "prune, then distill" strategy is effective in large scale setups as well. We refer to the Appendix for a detailed setup and results of large-scale experiments.

6 Conclusion

Our experiments showed that the pruned teacher can be more effective than the original teacher in KD. We further showed theoretically that the pruned teacher provides an additional regularization in distillation. Based on this observation, we proposed a novel network compression scheme that distills a pruned teacher network to the student network whose architecture is based on an (unstructured) pruned network. The proposed network compression is effectively a structured pruning algorithm that utilizes the knowledge of sparsity from unstructured pruning, and therefore our work bridges two main pruning approaches.

Acknowledgments

JP and AN were supported by Basic Science Research Program through the National Research Foundation of Korea (NRF) funded by the Ministry of Education (2021R1F1A1059567). We thank Minhyeok Cho for giving valuable comments. We also thank anonymous reviewers for providing constructive feedback.

References

1. Anwar, S., Hwang, K., Sung, W.: Structured pruning of deep convolutional neural networks. ACM Journal on Emerging Technologies in Computing Systems (JETC) **13**(3), 1–18 (2017)
2. Banner, R., Hubara, I., Hoffer, E., Soudry, D.: Scalable methods for 8-bit training of neural networks. In: NeurIPS (2018)
3. Chen, T., Kornblith, S., Swersky, K., Norouzi, M., Hinton, G.E.: Big self-supervised models are strong semi-supervised learners. In: NeurIPS (2020)
4. Cho, J.H., Hariharan, B.: On the efficacy of knowledge distillation. In: ICCV (2019)
5. Czarnecki, W.M., Osindero, S., Jaderberg, M., Swirszcz, G., Pascanu, R.: Sobolev training for neural networks. In: NeurIPS (2017)
6. Deng, J., Dong, W., Socher, R., Li, L.J., Li, K., Fei-Fei, L.: Imagenet: A large-scale hierarchical image database. In: CVPR (2009)
7. Devlin, J., Chang, M.W., Lee, K., Toutanova, K.: Bert: Pre-training of deep bidirectional transformers for language understanding. In: NAACL-HLT (2019)
8. Floridi, L., Chiriatti, M.: Gpt-3: Its nature, scope, limits, and consequences. Minds and Machines **30**(4), 681–694 (2020)
9. Frankle, J., Carbin, M.: The lottery ticket hypothesis: Finding sparse, trainable neural networks. In: ICLR (2019)
10. Frankle, J., Dziugaite, G.K., Roy, D., Carbin, M.: Linear mode connectivity and the lottery ticket hypothesis. In: ICML (2020)
11. Grill, J.B., Strub, F., Altché, F., Tallec, C., Richemond, P., Buchatskaya, E., Doersch, C., Avila Pires, B., Guo, Z., Gheshlaghi Azar, M., Piot, B., kavukcuoglu, k., Munos, R., Valko, M.: Bootstrap your own latent - a new approach to self-supervised learning. In: NeurIPS (2020)
12. Han, S., Liu, X., Mao, H., Pu, J., Pedram, A., Horowitz, M.A., Dally, W.J.: Eie: Efficient inference engine on compressed deep neural network. ACM SIGARCH Computer Architecture News **44**(3), 243–254 (2016)
13. Han, S., Mao, H., Dally, W.J.: Deep compression: Compressing deep neural network with pruning, trained quantization and huffman coding. In: ICLR (2016)
14. Havasi, M., Peharz, R., Hernandez-Lobato, J.M.: Minimal random code learning: Getting bits back from compressed model parameters. In: ICLR (2019)
15. He, K., Fan, H., Wu, Y., Xie, S., Girshick, R.: Momentum contrast for unsupervised visual representation learning. In: CVPR (2020)
16. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: CVPR (2016)
17. He, Y., Kang, G., Dong, X., Fu, Y., Yang, Y.: Soft filter pruning for accelerating deep convolutional neural networks. In: IJCAI (2018)
18. He, Y., Zhang, X., Sun, J.: Channel pruning for accelerating very deep neural networks. In: ICCV (2017)
19. Heo, B., Lee, M., Yun, S., Choi, J.Y.: Knowledge transfer via distillation of activation boundaries formed by hidden neurons. In: AAAI (2019)
20. Hinton, G., Vinyals, O., Dean, J.: Distilling the knowledge in a neural network. In: NeurIPS Workshop (2015)
21. Huang, Y., Cheng, Y., Bapna, A., Firat, O., Chen, D., Chen, M., Lee, H., Ngiam, J., Le, Q.V., Wu, Y., et al.: Gpipe: Efficient training of giant neural networks using pipeline parallelism. In: NeurIPS (2019)
22. Idelbayev, Y., Carreira-Perpinan, M.A.: Low-rank compression of neural nets: Learning the rank of each layer. In: CVPR (2020)

23. Jing, Y., Yang, Y., Wang, X., Song, M., Tao, D.: Amalgamating knowledge from heterogeneous graph neural networks. In: CVPR (2021)
24. Krizhevsky, A., Hinton, G., et al.: Learning multiple layers of features from tiny images. (Technical Report) (2009)
25. Le, Y., Yang, X.: Tiny imagenet visual recognition challenge. (Technical Report) (2015)
26. LeCun, Y., Denker, J.S., Solla, S.A.: Optimal brain damage. In: NeurIPS (1990)
27. Lee, J., Park, S., Mo, S., Ahn, S., Shin, J.: Layer-adaptive sparsity for the magnitude-based pruning. In: ICLR (2021)
28. LeJeune, D., Javadi, H., Baraniuk, R.: The flip side of the reweighted coin: Duality of adaptive dropout and regularization. In: NeurIPS (2021)
29. Li, F., Zhang, B., Liu, B.: Ternary weight networks. arXiv:1605.04711 (2016)
30. Li, G., Zhang, J., Wang, Y., Liu, C., Tan, M., Lin, Y., Zhang, W., Feng, J., Zhang, T.: Residual distillation: Towards portable deep neural networks without shortcuts. In: NeurIPS (2020)
31. Li, H., Kadav, A., Durdanovic, I., Samet, H., Graf, H.P.: Pruning filters for efficient convnets. In: ICLR (2017)
32. 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: ICCV (2019)
33. Liu, Z., Li, J., Shen, Z., Huang, G., Yan, S., Zhang, C.: Learning efficient convolutional networks through network slimming. In: ICCV (2017)
34. Liu, Z., Sun, M., Zhou, T., Huang, G., Darrell, T.: Rethinking the value of network pruning. In: ICLR (2018)
35. Luo, J.H., Wu, J., Lin, W.: Thinet: A filter level pruning method for deep neural network compression. In: ICCV (2017)
36. Mirzadeh, S.I., Farajtabar, M., Li, A., Levine, N., Matsukawa, A., Ghasemzadeh, H.: Improved knowledge distillation via teacher assistant. In: AAAI (2020)
37. Park, D.Y., Cha, M.H., Jeong, C., Kim, D., Han, B.: Learning student-friendly teacher networks for knowledge distillation. In: NeurIPS (2021)
38. Patterson, D., Gonzalez, J., Le, Q., Liang, C., Munguia, L.M., Rothchild, D., So, D., Texier, M., Dean, J.: Carbon emissions and large neural network training. arXiv:2104.10350 (2021)
39. Renda, A., Frankle, J., Carbin, M.: Comparing rewinding and fine-tuning in neural network pruning. In: ICLR (2020)
40. Romero, A., Ballas, N., Kahou, S.E., Chassang, A., Gatta, C., Bengio, Y.: Fitnets: Hints for thin deep nets. In: ICLR (2015)
41. Sainath, T.N., Kingsbury, B., Sindhwani, V., Arisoy, E., Ramabhadran, B.: Low-rank matrix factorization for deep neural network training with high-dimensional output targets. In: ICASSP (2013)
42. Sandler, M., Howard, A., Zhu, M., Zhmoginov, A., Chen, L.C.: Mobilenetv2: Inverted residuals and linear bottlenecks. In: CVPR (2018)
43. Simonyan, K., Zisserman, A.: Very deep convolutional networks for large-scale image recognition. In: ICLR (2015)
44. Srinivas, S., Fleuret, F.: Knowledge transfer with jacobian matching. In: ICML (2018)
45. Stanton, S., Izmailov, P., Kirichenko, P., Alemi, A.A., Wilson, A.G.: Does knowledge distillation really work? In: NeurIPS (2021)
46. Su, X., You, S., Wang, F., Qian, C., Zhang, C., Xu, C.: Bcnet: Searching for network width with bilaterally coupled network. In: CVPR (2021)
47. Tanaka, H., Kunin, D., Yamins, D.L., Ganguli, S.: Pruning neural networks without any data by iteratively conserving synaptic flow. In: NeurIPS (2020)

48. Tian, Y., Krishnan, D., Isola, P.: Contrastive representation distillation. In: ICLR (2019)
49. Wen, W., Wu, C., Wang, Y., Chen, Y., Li, H.: Learning structured sparsity in deep neural networks. In: NeurIPS (2016)
50. Wiedemann, S., Kirchhoffer, H., Matlage, S., Haase, P., Marban, A., Marinc, T., Neumann, D., Nguyen, T., Schwarz, H., Wiegand, T., Marpe, D., Samek, W.: Deepcabac: A universal compression algorithm for deep neural networks. IEEE Journal of Selected Topics in Signal Processing **14**(4), 700–714 (2020)
51. Xu, Z., Hsu, Y.C., Huang, J.: Training shallow and thin networks for acceleration via knowledge distillation with conditional adversarial networks. In: ICLR Workshop (2017)
52. Yang, Y., Qiu, J., Song, M., Tao, D., Wang, X.: Distilling knowledge from graph convolutional networks. In: CVPR (2020)
53. 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)
54. Yuan, L., Tay, F.E., Li, G., Wang, T., Feng, J.: Revisiting knowledge distillation via label smoothing regularization. In: CVPR (2020)
55. Zagoruyko, S., Komodakis, N.: Wide residual networks. In: BMVC (2016)
56. Zagoruyko, S., Komodakis, N.: Paying more attention to attention: Improving the performance of convolutional neural networks via attention transfer. In: ICLR (2017)
57. Zhou, H., Alvarez, J.M., Porikli, F.: Less is more: Towards compact cnns. In: ECCV (2016)
58. Zhou, H., Song, L., Chen, J., Zhou, Y., Wang, G., Yuan, J., Zhang, Q.: Rethinking soft labels for knowledge distillation: A bias–variance tradeoff perspective. In: ICLR (2021)
59. Zhuang, T., Zhang, Z., Huang, Y., Zeng, X., Shuang, K., Li, X.: Neuron-level structured pruning using polarization regularizer. In: NeurIPS (2020)