PrUE: Distilling Knowledge from Sparse Teacher Networks

Shaopu Wang[1,2], Xiaojun Chen[2(✉)], Mengzhen Kou[1,2], and Jinqiao Shi[3]

[1] School of Cyber Security, University of Chinese Academy of Sciences,
Beijing, China
[2] Institute of Information Engineering, Chinese Academy of Sciences, Beijing, China
{wangshaopu,chenxiaojun,koumengzhen}@iie.ac.cn
[3] Beijing University of Posts and Telecommunication, Beijing, China
shijinqiao@bupt.edu.cn

Abstract. Although deep neural networks have enjoyed remarkable success across a wide variety of tasks, their ever-increasing size also imposes significant overhead on deployment. To compress these models, knowledge distillation was proposed to transfer knowledge from a cumbersome (teacher) network into a lightweight (student) network. However, guidance from a teacher does not always improve the generalization of students, especially when the size gap between student and teacher is large. Previous works argued that it was due to the high certainty of the teacher, resulting in harder labels that were difficult to fit. To soften these labels, we present a pruning method termed Prediction Uncertainty Enlargement (PrUE) to simplify the teacher. Specifically, our method aims to decrease the teacher's certainty about data, thereby generating soft predictions for students. We empirically investigate the effectiveness of the proposed method with experiments on CIFAR-10/100, Tiny-ImageNet, and ImageNet. Results indicate that student networks trained with sparse teachers achieve better performance. Besides, our method allows researchers to distill knowledge from deeper networks to improve students further. Our code is made public at: https://github.com/wangshaopu/prue.

Keywords: Knowledge distillation · Network pruning · Deep learning

1 Introduction

Neural networks have gained remarkable practical success in many fields [3]. In practice, researchers usually introduce more layers and parameters to make the network deeper [37] and wider [15] for achieving better performance. However, these over-parameterized models also incur huge computational and storage overhead [5], which makes deploying them on edge devices impractical. Therefore, several methods have been proposed to shrink neural networks, e.g., network pruning [11,17], quantization [10], and knowledge distillation [13]. Among these approaches, knowledge distillation has been widely utilized in many fields [2,39].

Table 1. The test accuracy in percentage of various teachers and ResNet-8 as the student.

	ResNet-8	ResNet-20 w/o LS	ResNet-20 w/LS	ResNet-32 w/o LS
Teacher Acc.	87.56($\pm$0.20)	91.72($\pm$0.21)	92.06($\pm$0.26)	92.99($\pm$0.12)
Student Acc.	–	88.05($\pm$0.18)	86.13($\pm$0.22)	87.60($\pm$0.08)
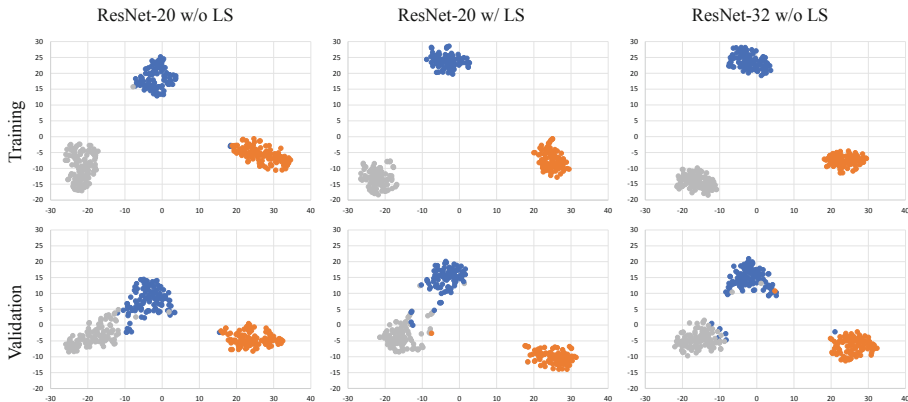
Fig. 1. Visualization of network predictions. We randomly select some training samples from the three classes of CIFAR-10 "airplane" (gray), "automobile" (blue), and "bird" (yellow), and then perform t-SNE dimensionality reduction [22] on network predictions. Note that the x-y axis has no real meaning here. (Color figure online)

Generally speaking, it utilizes a pre-trained teacher to produce supervision for students. In this way, a lightweight student network can achieve similar generalization as the teacher.

Although this paradigm of encouraging students to mimic teachers' behaviors has proven to be a promising way, some recent works [25,30] argued that knowledge distillation is not always effective. Specifically, it is found that well-behaved teachers failed to improve student generalization under certain circumstances. For instance, Müller et al. [25] discovered that teachers pre-trained with label smoothing (LS) [31], a commonly used technique to regularize models, will distill inferior students, even though the teacher's generalization has been improved. They attribute this phenomenon to the fact that LS tends to erase the relative information within a class. As a result, teachers generate harder labels that are difficult for students to fit. Meanwhile, Mirzadeh et al. [24] investigated another more common scenario. When there exists a large capacity gap between students and teachers, the former will perform worse. Coincidentally, their experiments lead to a similar conclusion that well-performed teachers fail to generate soft targets.

To investigate the relationship between network capacity and label smoothing, we train ResNet-20 and ResNet-32 on CIFAR-10 and report the results of

visualizing their predictions for the classes "airplane", "automobile" and "bird" in Fig. 1. The first row represents examples from the training set, while the second row is from the validation set. As revealed in the first column, a ResNet-20 trained without label smoothing (w/o LS) produces predictions scattered in some broad clusters. We also notice that blue dots (automobile) and gray dots (airplanes) in the validation set tend to be mixed at the boundary. A possible explanation is that these vehicles are more similar in some features than the yellow dots (birds), and it causes some misclassification. While in the second column, a ResNet-20 is trained with a label smoothing factor of 0.2. We observe that LS encourages samples in the training set to be equidistant from other classes' centers. What is striking in this figure is the third column. We train a ResNet-32 and notice that it acts in a similar pattern to LS. They both compact each class cluster. Next, we use ResNet-8 as a student to validate the effectiveness of knowledge distillation. The accuracy results, as shown in Table 1, confirm that while label smoothing and network deepening can improve the teacher network, they will degrade the generalization of students as expected.

A possible speculation is that although the generalization of the networks can be improved by the above two measures, their uncertainty about the data is also reduced. As a result, teachers tend to produce similar overconfident predictions for all intra-class samples and distill inferior students. In this work, we propose to improve knowledge distillation by increase teachers' uncertainty. Fortunately, a statistical metric, which we term prediction uncertainty, has been proposed by [29] to quantify this phenomenon. Following this work, we propose a criterion to identify the effect of weights on uncertainty in the teacher network. Then we prune those less-contributing weights before distillation. Differing from traditional pruning algorithms that focus on generalization, our method aims to reduce the generalization error of student networks by softening teacher predictions. We name our method Prediction Uncertainty Enlargement (PrUE).

We evaluate our pruning method on CIFAR-10/100, Tiny-ImageNet, and ImageNet classification datasets with some modern neural networks. Specifically, we first verify that label smoothing and network deepening reduce generalization error with a sacrifice of prediction uncertainty. The following distillation experiments show a positive correlation between the student's accuracy and the teacher's prediction uncertainty. However, the teacher's accuracy does not play a crucial role in knowledge distillation. Generally, large networks struggle to distill stronger students despite their high accuracy. To bridge this gap, we apply PrUE to the aforementioned teacher networks and distill their knowledge to students. Results show that our method can increase the teacher's prediction uncertainty, resulting in better performance improvement for students than existing distillation methods. We also compare PrUE with several other pruning schemes and observe that sparse teacher networks distill good students, but PrUE usually presents better performance.

Contributions: our contributions in this paper are as follows.

– We empirically investigate the impact of label smoothing and network capacity on knowledge distillation. Interestingly, They both prevent teachers from

generating soft labels and impair knowledge distillation, despite the improved
accuracy of teachers themselves.
– We apply a statistical metric to quantify the softness of labels. Based on this,
  PrUE is proposed to increase the teacher's prediction uncertainty.
– We perform experiments on CIFAR-10/100, Tiny-ImageNet, and ImageNet
  with widely varying CNN networks. Results suggest that sparse teacher net-
  works usually distill better students than dense ones. Besides, PrUE outper-
  forms existing distillation and pruning schemes.

2   Related Work

Network Pruning. The motivation behind network pruning is that there is a
mass of redundant parameters in the neural network [7]. Previous works have
demonstrated that these parameters can be removed safely. Therefore, Lecun
et al. [17] proposed removing parameters in an unstructured way by calculat-
ing the Hessian of the loss with respect to the weights. Furthermore, Han et
al. [11] proposed a magnitude-based pruning method to remove all weights below
s predefined threshold. Recently, Frankle et al. [8] proposed the "Lottery Ticker
Hypothesis" that there exist sparse subnetworks that, when trained in isolation,
can reach test accuracy comparable to the original network. Furthermore, Miao
et al. [23] proposed a framework that can prune neural networks to any sparsity
ratio with only one training.

Soft Labels. Theoretically, the widely used one-hot labels could lead to over-
fitting. Therefore, label smoothing was proposed to generate soft labels, thereby
delivering regularization effects. On the other hand, there were usually some
noisy labels in the dataset that mislead deep learning models, and a recent
work [20] noted that label smoothing could help mitigate label noise. However,
label smoothing could only add random noise and cannot reflect the relation-
ship between labels. Another well-known paradigm for generating soft labels
was knowledge distillation [13]. Differing label smoothing, knowledge distillation
required a pretrained teacher to produce soft labels for each training example.
Therefore, Yuan et al. [35] regarded it as a dynamic form of label smoothing.
Although the original distillation scheme focused on transferring dark knowl-
edge from large to small models, Zhang et al. [38] had found that these gener-
ated soft labels can be used for distributed machine learning. Therefore, some
recent works [2,39] proposed distillation-based communication schemes to save
bandwidth.

Pruning in Distillation. Both network pruning and knowledge distillation are
widely used model compression methods. Therefore, some recent works proposed
combining them together to achieve higher compression ratios. For instance, Xie
et al. [33] used this paradigm to customize a compression scheme for the identifi-
cation of Person re-identification (ReID). Chen et al. [4] proposed to use pruning

and knowledge distillation to train a lightweight detection model, to achieve synthetic aperture radar ship real-time detection at a lower cost. Meanwhile, Aghli et al. [1] introduced a compression scheme of convolutional neural networks, mainly exploring how to combine pruning and knowledge distillation methods to reduce the scale of ResNet with the guarantee of accuracy. Neill et al. [26] proposed a pruning-based self-distillation scheme using distillation as the pruning criterion to maximize the similarity of network representations before and after pruning. Cui et al. [6] proposed a joint model compression method that combines structured pruning and dense knowledge distillation. However, these researches focused on simplifying student networks. In fact, they amplify the capacity gap between students and teachers.

3 Background

Producing soft labels has been shown to be an effective regularizer. In practice, encouraging networks to fit soft labels prevents overfitting. In this section, we introduce a statistical metric quantifying label softness.

3.1 Preliminaries

Notations. Given a K-class classification task, We denote by $\mathcal{D}$ the training dataset, consisting of m i.i.d tuples $\{(\boldsymbol{x}_1, \boldsymbol{y}_1), \ldots, (\boldsymbol{x}_m, \boldsymbol{y}_m)\}$ where $\boldsymbol{x}_i \in \mathbb{R}^{d \times 1}$ is the input data and $\boldsymbol{y}_i \in \{0, 1\}^K$ is the corresponding one-hot class label. Let $\boldsymbol{y}[i]$ be the i-th element in $\boldsymbol{y}$, and $\boldsymbol{y}[c]$ is 1 for the ground-truth class and 0 for others.

Knowledge Distillation. For a teacher network $f(\boldsymbol{w}_{\mathcal{T}})$ parameterized by $\boldsymbol{w}_{\mathcal{T}}$, let $a(\boldsymbol{w}_{\mathcal{T}}, \boldsymbol{x}_i)$ and $f(\boldsymbol{w}_{\mathcal{T}}, \boldsymbol{x}_i)$ correspond to its logits and prediction for $\boldsymbol{x}_i$, respectively. In vanilla supervised learning, $f(\boldsymbol{w}_{\mathcal{T}})$ is usually trained on $\mathcal{D}$ with cross-entropy loss

$$\mathcal{L}_{CE} = -\sum_{i=1}^{m} \boldsymbol{y}_i \log f(\boldsymbol{w}_{\mathcal{T}}, \boldsymbol{x}_i) \tag{1}$$

where $f(\boldsymbol{w}_{\mathcal{T}}, \boldsymbol{x}_i) = softmax(a(\boldsymbol{w}_{\mathcal{T}}, \boldsymbol{x}_i))$.

As for a student network $f(\boldsymbol{w}_{\mathcal{S}})$, its logits and prediction for $\boldsymbol{x}_i$ are denoted as $a(\boldsymbol{w}_{\mathcal{S}}, \boldsymbol{x}_i)$ and $f(\boldsymbol{w}_{\mathcal{S}}, \boldsymbol{x}_i)$. In knowledge distillation, $f(\boldsymbol{w}_{\mathcal{S}})$ is usually trained with a given temperature τ and KL-divergence loss

$$\mathcal{L}_{KD} = -\sum_{i=1}^{m} \tau^2 KL(a(\boldsymbol{w}_{\mathcal{T}}, \boldsymbol{x}_i), a(\boldsymbol{w}_{\mathcal{S}}, \boldsymbol{x}_i)) \tag{2}$$

When the hyperparameter τ is set to 1, we can regard the distillation process as training $f(\boldsymbol{w}_{\mathcal{S}})$ on a new dataset $\{(\boldsymbol{x}_1, f(\boldsymbol{w}_{\mathcal{T}}, \boldsymbol{x}_1)), \ldots, (\boldsymbol{x}_m, f(\boldsymbol{w}_{\mathcal{T}}, \boldsymbol{x}_m))\}$ with soft labels provided by a teacher. The key idea behind knowledge distillation is to encourage the student $f(w_{\mathcal{S}})$ to mimic the behavior of the teacher $f(w_{\mathcal{T}})$. In

practice, researchers usually use correct labels to improve soft labels, especially when the generalization of teachers is poor. Therefore, the practical loss function for the student is modified as follows:

$$\mathcal{L}_{student} = \sum_{i=1}^{m} (1 - \lambda)\mathcal{L}_{CE} + \lambda\mathcal{L}_{KD} \tag{3}$$

where λ is another hyperparameter that controls the trade-off between the two losses. We refer to this approach as *Logits(τ)* through the paper.

Label Smoothing. Similar to knowledge distillation, label smoothing aims to replace hard labels to penalize overfitting. Instead, it does not involve a teacher network. Specifically, label smoothing modifies one-hot hard label vector $\boldsymbol{y}$ with a mixture of weighted origin $\boldsymbol{y}$ and a uniform distribution:

$$\boldsymbol{y}_c = \begin{cases} 1 - \alpha & \text{if } c = label, \\ \alpha/(K-1) & \text{otherwise.} \end{cases} \tag{4}$$

where $\alpha \in [0, 1]$ is the hyperparameter flattening the one-hot labels.

Label smoothing has been a widely used *trick* to improve network generalization. A prior work [29] observes that although the network trained with label smoothing suffers a higher cross-entropy loss on the validation set, its accuracy is better than that without label smoothing.

3.2   Prediction Uncertainty

To observe the effect of label smoothing on the penultimate layer representations, Müller et al. [25] proposed a visualization scheme based on squared Euclidean distance. Similarly, we use t-SNE in Sect. 1 to visualize the predictions. However, we cannot conduct numerical analysis on these intuitive presentations. To further measure the label softness quantitatively and address the *erasing* phenomenon caused by label smoothing, Shen et al. [29] propose a simple yet effective metric. The definition is as follows[1]:

$$\delta(\boldsymbol{w}) = \frac{1}{K} \sum_{c=1}^{K} \left(\frac{1}{\boldsymbol{n}_c} \sum_{i=1}^{\boldsymbol{n}_c} \| f(\boldsymbol{w}, \boldsymbol{x}_i)[c] - \tilde{f}(\boldsymbol{w}, \boldsymbol{x}_i)[c] \|^2 \right) \tag{5}$$

where class c contains $\boldsymbol{n}_c$ samples. $\tilde{f}(\boldsymbol{x}_i)[c]$ is the mean of in $f(\boldsymbol{x}_i)$ class c. The key idea behind this metric is to use the variance of intra-class probabilities to measure the uncertainty of network predictions.

Now we discuss how prediction uncertainty influences knowledge distillation. Assume an ideal network classifies each input precisely, and it is absolutely certain of each prediction. Correspondingly, this network is commonly regarded as

---

[1] It was called *stability* in the origin paper. We modify it for the purpose of our work.

a perfect model that achieves excellent generalization and low loss on the validation set. However, it tends to produce one-hot labels that fail to inform student networks about the similarity between classes, i.e., dark knowledge. At this point, the certainty of the teacher network downgrades the knowledge distillation to vanilla training. Applying label smoothing to the distillation process could help to moderate the teacher's overconfidence. Unfortunately, this trick merely tells students that airplanes and birds have the same probability as automobiles. Therefore, we aim to make teachers feel uncertain between the automobile and the airplane, thus improving the generalization behavior of the student network.

We next work on simplifying the teacher network to enlarge its prediction uncertainty. Specifically, we utilize network pruning to close the capacity gap between teachers and students.

4 Prediction Uncertainty Enlargement

In deep model compression, network pruning aims to deliver the regularization effect to neural networks by simply removing parameters. Following the discussion above, we introduce auxiliary indicator variables $\boldsymbol{m} \in \{0,1\}^l$ representing the pruning mask. Then the enlargement of prediction uncertainty is formulated as an optimization problem as:

$$\max_{\boldsymbol{m}} \delta(\boldsymbol{m} \odot \boldsymbol{w}) = \max_{\boldsymbol{m}} \frac{1}{K} \sum_{c=1}^{K} (\frac{1}{\boldsymbol{n}_c} \sum_{i=1}^{\boldsymbol{n}_c} \|f(\boldsymbol{m} \odot \boldsymbol{w}, \boldsymbol{x}_i)[c] - \tilde{f}(\boldsymbol{m} \odot \boldsymbol{w}, \boldsymbol{x}_i)[c]\|^2),$$
$$s.t. \quad \boldsymbol{m} \in \{0,1\}^l, \quad \|\boldsymbol{m}\|_0 \leq s,$$
(6)

where $\odot$ denotes the Hadamard product.

Solving such a combinatorial optimization problem requires computing its $\delta(\boldsymbol{m} \odot \boldsymbol{w})$ for each candidate in the solution space, that is, it requires up to $l \times l$ forward passes over the training dataset. However, the number of network parameters has increased substantially recently. Since an arms race of training large models has begun, millions of calculations $\delta(\boldsymbol{m} \odot \boldsymbol{w})$ are unacceptable.

Following [16,18], we next measure the impact of each weight on the network uncertainty and then prune less-contributing weights greedily. Since it is impractical to directly solve this optimization problem with respect to binary variables $\boldsymbol{m}$, we first relax $\boldsymbol{m}$ into real variables $\boldsymbol{m} \in [0,1]^l$. This change can be seen as a form of soft pruning, where the corresponding mask $\boldsymbol{m}[j]$ is gradually reduced from 1 to 0. In this way, the optimization problem is differentiable with respect to $\boldsymbol{m}$. We rewrite Optimization (6) as follows:

$$\max_{\boldsymbol{m}} \delta(\boldsymbol{m} \odot \boldsymbol{w}) = \max_{\boldsymbol{m}} \frac{1}{K} \sum_{c=1}^{K} (\frac{1}{\boldsymbol{n}_c} \sum_{i=1}^{\boldsymbol{n}_c} \|f(\boldsymbol{m} \odot \boldsymbol{w}, \boldsymbol{x}_i)[c] - \tilde{f}(\boldsymbol{m} \odot \boldsymbol{w}, \boldsymbol{x}_i)[c]\|^2),$$
$$s.t. \quad \boldsymbol{m} \in [0,1]^l, \quad \|\boldsymbol{m}\|_0 \leq s,$$
(7)

This modification allows us to perturb the mask instead of setting it to zero. For the weight $\boldsymbol{w}[j]$, we add an infinitesimal perturbation ϵ to the mask $\boldsymbol{m}[j]$ to obtain its influence on $\delta(\boldsymbol{m} \odot \boldsymbol{w})$. Its magnitude of differential $\triangle\delta_j(\boldsymbol{m} \odot \boldsymbol{w})$ indicates the dependence of $\delta(\boldsymbol{m} \odot \boldsymbol{w})$ on $\boldsymbol{w}[j]$. Next, we find the derivative of $\delta(\boldsymbol{m} \odot \boldsymbol{w})$ with respect to $\boldsymbol{m}[j]$ as follows:

$$\lim_{\epsilon \to 0} \frac{\delta(\boldsymbol{m} \odot \boldsymbol{w}) - \delta((\boldsymbol{1} - \epsilon\boldsymbol{e}_j)\boldsymbol{m} \odot \boldsymbol{w})}{\epsilon} = \lim_{\epsilon \to 0} \frac{\partial\delta(\boldsymbol{m} \odot \boldsymbol{w})}{\partial\boldsymbol{m}[j]} = g_j(\boldsymbol{w}). \qquad (8)$$

where $\boldsymbol{e}_j$ is a one-hot vector $[0, ..., 0, 1, 0, ..., 0]$ with a 1 at position j.

Thus, we measure the importance of the weight $\boldsymbol{w}[j]$ to the prediction uncertainty. To this end, we regard $|g_j(\boldsymbol{w})|$ as the proposed criterion. Given a desired sparsity s, we can achieve prediction uncertainty enlargement by pruning $s \times l$ weights that contribute less to the variance.

The key to our approach is to find the derivative of the uncertainty with respect to the pruning mask of each weight. However, restricted by the modern computing device, PrUE still faces some practical problems. Note that Optimization (7) calls $f(\boldsymbol{w})$ twice, which requires the automatic differentiation algorithm to perform two forward-backward pass through the computational graph. Modern deep learning frameworks like PyTorch usually free gradient tensors after the first backward pass to save memory. That is, our method consumes more resources due to retaining the computational graph.

On the other hand, our method requires computing the averaged intra-class probabilities for each class. In practice, researchers typically perform stochastic gradient descent by randomly selecting a mini-batch of training data, where the batchsize ranges from 128 to 1024. For a 10-class classification task like CIFAR-10, this batchsize is sufficient to estimate $\tilde{f}(\boldsymbol{x})[c]$, while not for ImageNet-1k containing 1000 classes. In fact, most classes in ImageNet-1k only appear once or twice in a batch, making accurate estimation of $\tilde{f}(\boldsymbol{x})[c]$ impractical.

One could take straightforward measures such as saving intermediate values of the graph or leveraging more devices, but this would result in additional overhead. Instead, we employ a simple yet effective trick to decompose the optimization into two steps. Specifically, we first compute $\tilde{f}(\boldsymbol{x})[c]$ for each class with the computational graph detached, then sort the dataset by labels, thus guaranteeing that only class c appears in each batch. Finally, $f(\boldsymbol{x})[c]$ can be estimated in the current batch. We empirically observed that this trick only slightly affects the results, but saves appreciable memory.

5   Experiments

In this section, we empirically investigate the effect of our proposed method on knowledge distillation. In addition, we compare PrUE with other distillation and pruning methods. The results show that our paradigm of distilling knowledge from sparse teacher networks tends to yield better students. Moreover, PrUE can exhibit better performance.

Table 2. Number of weights and training hyperparameters in our experiments.

Dataset	Network	# Weights	Epochs	Batchsize	Schedule
CIFAR-10	ResNet-8 [12]	78K	160	512	10× drop at 81, 122
	ResNet-20	272K			
	ResNet-32	466K			
	ResNet-56	855K			
	ResNet-110	1.7M			
CIFAR-100	ResNet-8	83.9K	160	512	10× drop at 81, 122
	ResNet-20	278K			
	ResNet-32	472K			
	ResNet-56	861K			
	ResNet-110	1.7M			
Tiny-ImageNet	ResNext-50 [34]	15.0M	200	256	5× drop at 60, 120, 160
	ShuffleNet V2 [21]	1.5M			
ImageNet	EffcientNet-B2 [32]	9.1M	90	256	10× drop at 30, 60
	MobileNet V3 Small [14]	2.5M			

Implementation Details. We conduct all experiments on 8 * NVIDIA Tesla A100 GPU. The sparsity level is defined to be $s = k/l \times 100(\%)$, where k is the number of zero weights, and l is the total number of network weights. All networks are trained with SGD with Nesterov momentum. We set the initial learning rate to 0.1, momentum to 0.9. Table 2 describes the number of parameters of all the networks and corresponding training hyperparameters. During distillation, we set λ to 1 for CIFAR-10 and 0.1 for the rest tasks.

5.1 The Effect of LS on Knowledge Distillation

We first investigate the compatibility of label smoothing and knowledge distillation on CIFAR-10 and CIFAR-100. Specifically, we train ResNet-20/32/56/100 with label smoothing turned on or turned off, then distill their knowledge into ResNet-8. Table 3 presents the accuracy of student networks supervised by various teachers. We also report the vanilla supervised training results of ResNet-8 for baseline comparison.

Table 3. The test accuracy of a fixed student with various teachers trained without (w/o) or with (w/) label smoothing. The vanilla supervised results of ResNet-8 is also reported.

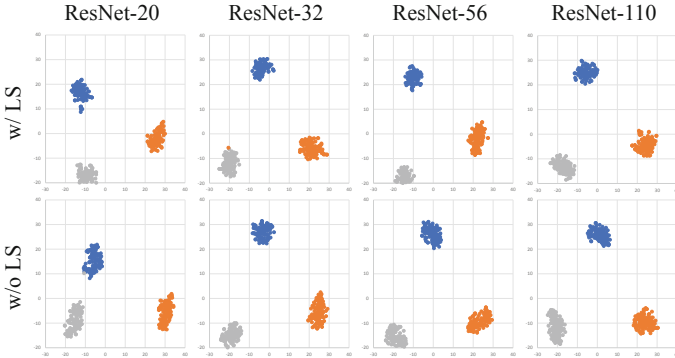|	Vanilla		ResNet-20	ResNet-32	ResNet-56	ResNet-110
CIFAR-10	87.56	w/LS	86.62(±0.21)	85.56(±0.25)	85.61(±0.03)	85.88(±0.19)
		w/o LS	88.36(±0.12)	87.48(±0.22)	87.50(±0.10)	87.47(±0.17)
CIFAR-100	59.36	w/LS	58.75(±0.23)	58.73(±0.16)	59.14(±0.14)	58.52(±0.25)
		w/o LS	59.81(±0.19)	59.50(±0.24)	59.47(±0.17)	59.76(±0.09)

Fig. 2. Visualization of predictions of more network structures.

Although deep neural networks are well known for their generalization ability, they fail to bring proportional improvement for students. In particular, ResNet-20 tends to distill better students than other well-generalized teachers. Similarly, teachers trained with hard labels achieve better distillation results compared to those trained with label smoothing. To demonstrate this phenomenon, we provide visualizations of these teachers' predictions in Fig. 2. As we can see, network deepening and label smoothing compacts each cluster and thus impairs knowledge distillation in Table 3.

5.2   Comparison with Other Distillation Methods

Intuitively, improved teachers are overconfident in each sample, thus producing harder predictions containing low information. To enlarge teacher uncertainty without sacrificing generalization, we apply PrUE to prune them, and then fine-tune them to restore accuracy.

Table 4. The test accuracy (%) and uncertainty (1e−2) of teacher networks with varying sparsity.

| Dataset	Sparsity	ResNet-20		ResNet-32		ResNet-56		ResNet-110	
		Acc.	Uncer.	Acc.	Uncer.	Acc.	Uncer.	Acc.	Uncer.
CIFAR-10	$s = 0$	91.72	6.40	93.17	3.34	93.40	2.04	93.38	1.55
	$s = 20\%$	92.82	6.12	93.54	3.19	93.75	2.12	94.14	1.31
	$s = 50\%$	91.97	8.19	93.08	4.23	93.77	2.39	93.75	1.74
	$s = 90\%$	87.98	22.87	90.63	17.49	91.64	13.00	92.13	7.28
CIFAR-100	$s = 0$	68.61	26.01	69.65	19.16	71.29	10.72	71.84	5.72
	$s = 20\%$	69.04	25.83	70.44	18.77	72.01	10.47	73.36	5.55
	$s = 50\%$	68.26	27.97	69.18	22.55	71.33	14.56	72.83	7.18
	$s = 90\%$	54.30	28.33	60.49	30.53	62.92	30.78	62.16	30.52
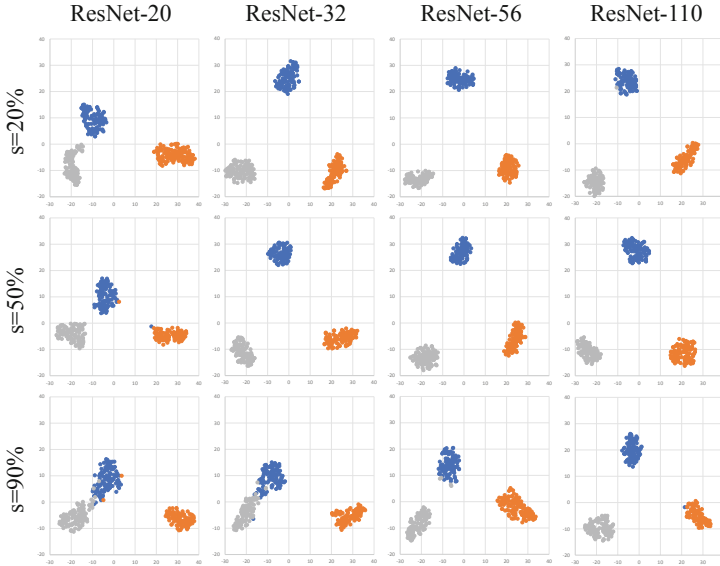
Fig. 3. Predictive visualization of networks with varying sparsity *s*. As the network deepens, the predictions get tighter. While the increasing sparsity spreads the predictions into broad clusters.

Figure 3 visualizes these sparse teacher networks. As the sparsity *s* increases, the teacher's predictions are scattered into wider clusters. We also observe that a higher sparsity is appropriate for deep networks such as ResNet-110. On the other hand, Table 4 provides quantitative results. It suggests that PrUE can effectively improve teachers' uncertainty with slight loss in performance.

Next, we distill knowledge from these sparse teachers to a ResNet-8. Meanwhile, we compare our method with other distillation methods. Table 5 and Table 6 depicts the results of students performance on CIFAR-10 and CIFAR-100, respectively. It is worth noting that λ is set to 0 on CIFAR-10, which means that our method can only obtain the teacher's prediction, while the others can receive the ground truth. Although this is an unfair comparison, PrUE still outperforms existing distillation methods notably. Another interesting observation is that teachers with high uncertainty distill better students, even when their accuracy is hurt by pruning. Therefore, we conclude that teacher uncertainty plays an important role in knowledge distillation, rather than accuracy.

5.3 Comparison with Other Pruning Methods

With promising results on distillation, we further compare PrUE with other pruning methods. In particular, we first train the teacher from scratch and apply several one-shot pruning algorithms (Magnitude [11,19], SNIP [18], Random [9], PrUE) to remove a portion of weights of the trained network, then fine-tune these

Table 5. The test accuracy of ResNet-8 on CIFAR-10 using different distillation methods. TA(20), TA(32) refers to using ResNet-20 and ResNet-32 as a teacher assistant, respectively.

| | CIFAR-10 | | | |
	ResNet-20	ResNet-32	ResNet-56	ResNet-110
Logit ($\tau = 1$) [13]	88.36($\pm$0.16)	87.48($\pm$0.22)	87.50($\pm$0.29)	87.47($\pm$0.28)
Logit ($\tau = 4$)	88.72($\pm$0.26)	88.39($\pm$0.17)	88.66($\pm$0.21)	88.34($\pm$0.29)
FitNet [28]	87.00($\pm$0.24)	86.83($\pm$0.27)	86.68($\pm$0.08)	86.62($\pm$0.10)
AT [36]	86.64($\pm$0.14)	86.37($\pm$0.16)	86.71($\pm$0.09)	86.76($\pm$0.17)
PKT [27]	87.41($\pm$0.04)	87.30($\pm$0.24)	87.26($\pm$0.13)	87.08($\pm$0.19)
TA(20) [24]	–	87.55($\pm$0.26)	87.87($\pm$0.19)	87.66($\pm$0.16)
TA(32)	–	–	87.83($\pm$0.10)	87.37($\pm$0.28)
PrUE ($s = 20\%$)	88.89($\pm$0.11)	88.30($\pm$0.06)	88.49($\pm$0.19)	88.47($\pm$0.26)
PrUE ($s = 50\%$)	**89.17**($\pm$0.19)	**88.39**($\pm$0.07)	88.68($\pm$0.23)	89.22($\pm$0.15)
PrUE ($s = 90\%$)	87.01($\pm$0.20)	87.95($\pm$0.24)	**89.08**($\pm$0.24)	**89.27**($\pm$0.18)

pruned networks until convergence. We use ResNet-8 as a student to evaluate the distillation performance of these sparse teachers.

Table 6. The test accuracy of ResNet-8 on CIFAR-100 using different distillation methods.

| | CIFAR-100 | | | |
	ResNet-20	ResNet-32	ResNet-56	ResNet-110
Logit ($\tau = 1$)	59.51($\pm$0.10)	59.25($\pm$0.23)	59.09($\pm$0.08)	59.56($\pm$0.26)
Logit ($\tau = 4$)	59.81($\pm$0.12)	59.50($\pm$0.05)	59.47($\pm$0.21)	59.76($\pm$0.12)
FitNet	58.92($\pm$0.20)	58.53($\pm$0.49)	58.59($\pm$0.07)	58.37($\pm$0.11)
AT	58.52($\pm$0.17)	58.74($\pm$0.09)	58.60($\pm$0.07)	57.87($\pm$0.24)
PKT	58.57($\pm$0.17)	58.74($\pm$0.05)	58.96($\pm$0.27)	58.81($\pm$0.06)
TA(20)	–	59.60($\pm$0.25)	59.45($\pm$0.09)	59.14($\pm$0.18)
TA(32)	–	–	59.68($\pm$0.12)	59.65($\pm$0.09)
PrUE ($s = 20\%$)	59.54($\pm$0.06)	59.66($\pm$0.12)	59.95($\pm$0.05)	59.56($\pm$0.14)
PrUE ($s = 50\%$)	**59.9**($\pm$0.30)	**59.71**($\pm$0.25)	**60.03**($\pm$0.37)	59.85($\pm$0.27)
PrUE ($s = 90\%$)	58.89($\pm$0.32)	59.44($\pm$0.19)	59.85($\pm$0.20)	**60.17**($\pm$0.08)

As illustrated in Fig. 4, our strategy of distilling knowledge from sparse networks can effectively improve the generalization behavior of student networks. Even if the weights in the network are randomly removed, students can still benefit from it. We also notice that PrUE could only exhibit similar performance to other pruning methods on shallower networks. Such as on 90% sparse ResNet-32, PrUE exhibits lower distillation performance (87.95%) than Magnitude (88.53%) and SNIP (88.14%). But as the network grows, our method

achieves better results (up to 89.27%). This result suggests that while previous work has argued that the large capacity gap between teachers and students results in lower performance gains, our approach allows researchers to break the restriction and use deeper networks to obtain further improve student accuracy.
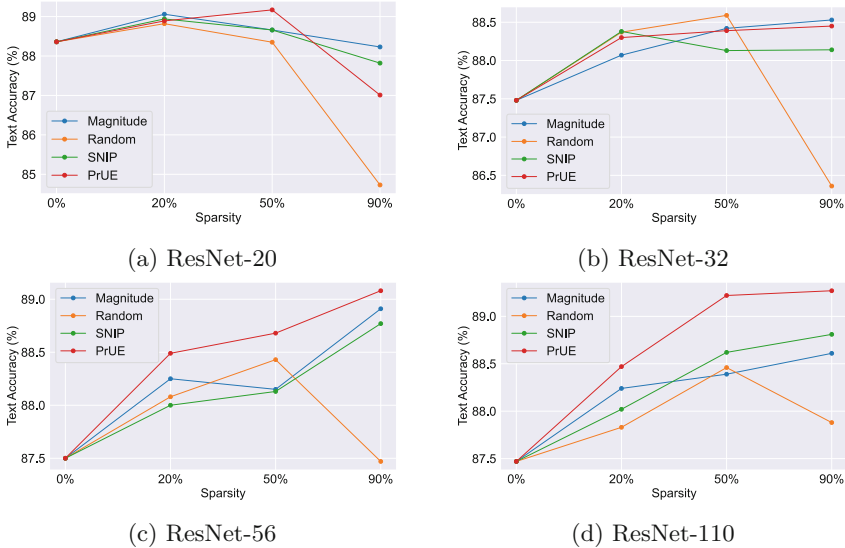

Fig. 4. Distillation accuracy of sparse teacher networks obtained using different pruning methods.

The Impact of Sparsity. We also find that inappropriate sparsity affects the distillation results of all pruning algorithms. For instance, ResNet-20 with 90% sparsity could face a 1–2% drop in distillation accuracy, although this result still outperforms traditional distillation methods in Table 5. While networks with more parameters like ResNet-110 can endure a higher sparsity ratio. Overall, if the size of teachers is much larger than that of students, we suggest a higher sparsity to bridge the capacity gap.

5.4   Distillation on Large-Scale Datasets

In this section, we consider practical applications on more challenging datasets. In practice, some large convolutional networks have been proposed to achieve better results on ImageNet tasks. On the other hand, researchers designed some lightweight networks to reduce overhead and accelerate inference. We aim to answer whether PrUE still works between these two different network structures.

We train ResNext-50 on Tiny-ImageNet as teacher network, while ShuffleNetV2 serves as the student. As for ImageNet, we distill knowledge from EfficientNet-B2 into MobileNetV3. Table 7 and Table 8 reports their own accuracy and distillation performance, respectively. Our method manages to improve

Table 7. The test accuracy (%) and uncertainty (1e−2) of sparse teacher networks on Tiny-ImageNet and ImageNet.

	$s = 0$		$s = 20\%$		$s = 50\%$		$s = 90\%$	
	Acc.	Uncer.	Acc.	Uncer.	Acc.	Uncer.	Acc.	Uncer.
ResNext-50	65.32	1.35	66.24	1.34	68.07	1.29	64.09	3.24
EfficientNet-B2	72.32	34.66	72.62	34.70	72.45	34.85	71.03	35.14

Table 8. The test accuracy of student network distilled by sparse teachers.

Teacher	Student	Vanilla	$s = 0$	$s = 20\%$	$s = 50\%$	$s = 90\%$
ResNext-50	ShuffleNet V2	62.09	63.28	63.45	64.09	64.65
EfficientNet-B2	MobileNet V3 Small	60.85	60.88	61.18	61.22	62.12

student generalization on real-world datasets. More interestingly, we observed on Tiny-ImageNet that the accuracy of the student network can sometimes exceed that of the teacher network. We believe this suggests that PrUE can be extended to a wider range of settings. Furthermore, we still lack understanding of knowledge distillation, and our proposed method could be a potential tool to shed light on it.

6   Conclusion

In this paper, we provided a data-dependent pruning method called PrUE to soften the network predictions, thereby improving its distillation performance. In particular, we proposed a computationally efficient criterion to estimate the effect of weights on uncertainty, and removed those less-contribution weights. We first showed a positive relationship between the uncertainty of the teacher network and its distillation effect through a visualization scheme. The following empirical experiments suggested that PrUE managed to increase the teacher uncertainty, thereby improving the distillation performance. Extensive experiments showed that our method notably outperformed traditional distillation methods. We also found that our strategy of distilling knowledge from sparse teacher networks could improve the generalization behavior of the student network, but the teacher pruned by PrUE tended to exhibit better performance.

References

1. 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)

2. Anil, R., Pereyra, G., Passos, A., Ormandi, R., Dahl, G.E., Hinton, G.E.: Large scale distributed neural network training through online distillation. In: International Conference on Learning Representations (2018)

3. Brown, T.B., et al.: Language models are few-shot learners. arXiv preprint arXiv:14165 (2020)

4. Chen, S., Zhan, R., Wang, W., Zhang, J.: Learning slimming SAR ship object detector through network pruning and knowledge distillation. IEEE J. Sel. Top. Appl. Earth Obs. Remote Sens. **14**, 1267–1282 (2020)

5. Cheng, J., Wang, P., Li, G., Hu, Q., Lu, H.: Recent advances in efficient computation of deep convolutional neural networks. Front. Inf. Technol. Electron. Eng. **19**(1), 64–77 (2018). https://doi.org/10.1631/FITEE.1700789

6. Cui, B., Li, Y., Zhang, Z.: Joint structured pruning and dense knowledge distillation for efficient transformer model compression. Neurocomputing **458**, 56–69 (2021)

7. Denil, M., Shakibi, B., Dinh, L., Ranzato, M., De Freitas, N.: Predicting parameters in deep learning. In: Advances in Neural Information Processing Systems, vol. 26 (2013)

8. Frankle, J., Carbin, M.: The lottery ticket hypothesis: finding sparse, trainable neural networks. In: International Conference on Learning Representations (2019)

9. Frankle, J., Dziugaite, G.K., Roy, D., Carbin, M.: Pruning neural networks at initialization: why are we missing the mark? In: International Conference on Learning Representations (2021)

10. Gong, Y., Liu, L., Yang, M., Bourdev, L.: Compressing deep convolutional networks using vector quantization. In: Computer Vision and Pattern Recognition (2014). arXiv

11. Han, S., Pool, J., Tran, J., Dally, W.J.: Learning both weights and connections for efficient neural networks. In: Neural Information Processing Systems, pp. 1135–1143 (2015)

12. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: Computer Vision and Pattern Recognition, pp. 770–778 (2016)

13. Hinton, G.E., Vinyals, O., Dean, J.: Distilling the knowledge in a neural network. In: Machine Learning (2015). arXiv

14. Howard, A., et al.: Searching for MobileNetV3. In: Proceedings of the IEEE/CVF International Conference on Computer Vision, pp. 1314–1324 (2019)

15. Ioffe, S., Szegedy, C.: Batch normalization: accelerating deep network training by reducing internal covariate shift. In: International Conference on Machine Learning, pp. 448–456. PMLR (2015)

16. Koh, P.W., Liang, P.: Understanding black-box predictions via influence functions. In: International Conference on Machine Learning, pp. 1885–1894. PMLR (2017)

17. Lecun, Y., Denker, J.S., Solla, S.A.: Optimal brain damage. In: Neural Information Processing Systems, vol. 2, pp. 598–605 (1989)

18. Lee, N., Ajanthan, T., Torr, P.H.S.: SNIP: single-shot network pruning based on connection sensitivity. In: International Conference on Learning Representations (2019)

19. Liu, N., et al.: Lottery ticket preserves weight correlation: is it desirable or not? In: International Conference on Machine Learning, pp. 7011–7020. PMLR (2021)

20. Lukasik, M., Bhojanapalli, S., Menon, A., Kumar, S.: Does label smoothing mitigate label noise? In: International Conference on Machine Learning, pp. 6448–6458. PMLR (2020)

21. Ma, N., Zhang, X., Zheng, H.-T., Sun, J.: ShuffleNet V2: practical guidelines for efficient CNN architecture design. In: Ferrari, V., Hebert, M., Sminchisescu, C.,

Weiss, Y. (eds.) Computer Vision – ECCV 2018. LNCS, vol. 11218, p. Shufflenet v2: Practical guidelines for efficient cnn architecture design-138. Springer, Cham (2018). https://doi.org/10.1007/978-3-030-01264-9_8

22. Van der Maaten, L., Hinton, G.: Visualizing data using t-SNE. J. Mach. Learn. Res. **9**(11), 2579–2605 (2008)

23. Miao, L., Luo, X., Chen, T., Chen, W., Liu, D., Wang, Z.: Learning pruning-friendly networks via frank-wolfe: one-shot, any-sparsity, and no retraining. In: International Conference on Learning Representations (2021)

24. Mirzadeh, S.I., Farajtabar, M., Li, A., Levine, N., Matsukawa, A., Ghasemzadeh, H.: Improved knowledge distillation via teacher assistant. In: Proceedings of the AAAI Conference on Artificial Intelligence, vol. 34, pp. 5191–5198 (2020)

25. Müller, R., Kornblith, S., Hinton, G.E.: When does label smoothing help? In: Advances in Neural Information Processing Systems, vol. 32 (2019)

26. Neill, J.O., Dutta, S., Assem, H.: Deep neural compression via concurrent pruning and self-distillation. arXiv preprint arXiv:2109.15014 (2021)

27. Passalis, N., Tefas, A.: Learning deep representations with probabilistic knowledge transfer. In: Ferrari, V., Hebert, M., Sminchisescu, C., Weiss, Y. (eds.) ECCV 2018. LNCS, vol. 11215, pp. 283–299. Springer, Cham (2018). https://doi.org/10.1007/978-3-030-01252-6_17

28. Romero, A., Ballas, N., Kahou, S.E., Chassang, A., Gatta, C., Bengio, Y.: FitNets: hints for thin deep nets. arXiv preprint arXiv:1412.6550 (2014)

29. Shen, Z., Liu, Z., Xu, D., Chen, Z., Cheng, K.T., Savvides, M.: Is label smoothing truly incompatible with knowledge distillation: an empirical study. In: International Conference on Learning Representations (2021)

30. Stanton, S., Izmailov, P., Kirichenko, P., Alemi, A.A., Wilson, A.G.: Does knowledge distillation really work? arXiv preprint arXiv:2106.05945 (2021)

31. Szegedy, C., Vanhoucke, V., Ioffe, S., Shlens, J., Wojna, Z.: Rethinking the inception architecture for computer vision. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 2818–2826 (2016)

32. Tan, M., Le, Q.V.: EfficientNet: rethinking model scaling for convolutional neural networks. In: International Conference on Machine Learning, pp. 6105–6114 (2019)

33. Xie, H., Jiang, W., Luo, H., Yu, H.: Model compression via pruning and knowledge distillation for person re-identification. J. Ambient Intell. Human. Comput. **12**(2), 2149–2161 (2021). https://doi.org/10.1007/s12652-020-02312-4

34. Xie, S., Girshick, R., Dollár, P., Tu, Z., He, K.: Aggregated residual transformations for deep neural networks. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 1492–1500 (2017)

35. Yuan, L., Tay, F.E., Li, G., Wang, T., Feng, J.: Revisit knowledge distillation: a teacher-free framework (2019)

36. Zagoruyko, S., Komodakis, N.: Paying more attention to attention: improving the performance of convolutional neural networks via attention transfer. arXiv preprint arXiv:1612.03928 (2016)

37. Zagoruyko, S., Komodakis, N.: Wide residual networks. arXiv preprint arXiv:1605.07146 (2016)

38. Zhang, Y., Xiang, T., Hospedales, T.M., Lu, H.: Deep mutual learning. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 4320–4328 (2018)

39. Zhu, Z., Hong, J., Zhou, J.: Data-free knowledge distillation for heterogeneous federated learning. arXiv preprint arXiv:10056 (2021)