Complex & Intelligent Systems (2022) 8:709–718
https://doi.org/10.1007/s40747-020-00248-y

ORIGINAL ARTICLE

Knowledge from the original network: restore a better pruned network with knowledge distillation

Liyang Chen[1] · Yongquan Chen[3] · Juntong Xi[1] · Xinyi Le[2,3]

Received: 9 October 2020 / Accepted: 2 December 2020 / Published online: 10 January 2021

Abstract

To deploy deep neural networks to edge devices with limited computation and storage costs, model compression is necessary for the application of deep learning. Pruning, as a traditional way of model compression, seeks to reduce the parameters of model weights. However, when a deep neural network is pruned, the accuracy of the network will significantly decrease. The traditional way to decrease the accuracy loss is fine-tuning. When over many parameters are pruned, the pruned network's capacity is reduced heavily and cannot recover to high accuracy. In this paper, we apply the knowledge distillation strategy to abate the accuracy loss of pruned models. The original network of the pruned network was used as the teacher network, aiming to transfer the dark knowledge from the original network to the pruned sub-network. We have applied three mainstream knowledge distillation methods: response-based knowledge, feature-based knowledge, and relation-based knowledge (Gou et al. in Knowledge distillation: a survey. arXiv:200605525, 2020), and compare the result to the traditional fine-tuning method with grand-truth labels. Experiments have been done on the CIFAR100 dataset with several deep convolution neural network. Results show that the pruned network recovered by knowledge distillation with its original network performs better accuracy than it recovered by fine-tuning with sample labels. It has also been validated in this paper that the original network as the teacher performs better than differently structured networks with same accuracy as the teacher.

Keywords Model compression · Network pruning · Knowledge distillation · Deep neural networks

Introduction

Deep neural networks have achieved excellent results that traditional machine learning can difficultly match in various fields, such as computer vision [20], point cloud processing [15,34], medical data processing [19], speech recognition [17], and so on. With the continuous development of deep learning, the artificial neural networks are becoming more and more deep, wide and complicated. Then the amount of neural network parameters is also explosively growing. However edge devices, like microphones and embedded systems, have limited computing resources, running memories, and storage space. The contradiction between the tremendous resource requirement of new deep learning technology and limited resource of hardware devices is hindering the application of deep learning technology. In order to enable deep learning applications deployed on these devices, various neural network compression strategies have been proposed including network pruning [9,22], quantization [16], and knowledge distillation [14].

It is considered that very deep neural networks are often over-parameterized [1] with many redundancy. The redundancy in over-parameterized network improves the generalization performance, but also leads to low efficiency and difficulties on edge deployment. To reduce the redundancy and obtain efficient neural networks, the method of pruning was proposed. The main idea of pruning is to remove the unimportant parameters of deep learning models. As the most effective model compression strategy, network pruning can reduce network parameters to less than a tenth. To a certain extent, pruning can reduce the parameters without impacting the accuracy of the model. However, when the percentage of pruned parameters is too large, the accuracy

✉ Xinyi Le
lexinyi@sjtu.edu.cn

[1] School of Mechanical Engineering, Shanghai Jiao Tong University, Shanghai, China

[2] Department of Automation, Shanghai Jiao Tong University, Shanghai, China

[3] Shenzhen Institue of Artificial Intelligence and Robotics for Society(AIRS), Shenzhen, China

of the model will inevitably decrease to a deficient level. In traditional methods, pruned models are retrained by the fine-tuning method after it is pruned, or in other case models are iteratively pruned and trained, to recover the model's accuracy. However this process below is often difficult to recover the accuracy of excessively pruned networks.

Knowledge distillation, as another compression strategy, aims to transfer dark knowledge in logits outputs [14], feature maps [13,18], and relationship diagrams [26] from a larger pre-trained teacher network to a smaller student network, allowing the student network to mimic the teacher network performance. The strategy of knowledge distillation can better improve some smaller networks' accuracy than directly training them with one-hot labels.

In this paper, we motivate to solve the accuracy loss problem in pruning, and merge the two pruning method. We propose a new strategy to recover the pruned neural network: we replace the fine-tuning procedure in pruning pipeline to knowledge distillation and transfer the knowledge from the original un-pruned network to the pruned network to increase the accuracy of the pruned network. We pruned an over-parameterized network and then used the original network as the teacher, and the pruned network as a student for knowledge distillation. As the teacher network and the student network have the same structure (the student network can be seen as a sub-network of the teacher network), the student network can better fit the representation of the teacher network [5,31]. So the proposed method is more effective than simply mechanically combining network pruning and knowledge distillation.

Our new method combines the advantages of several model compression methods. Compared to the latest knowledge distillation methods [2,33], our method focuses on generating a student model from the original model by pruning. Therefore, we get a generated student network better suits the teacher network than manually selected network in simple knowledge distillation methods. Then compared to the simple pruning methods [7,22,37], the idea of knowledge transfer is used to retraining the pruning network in our proposed method. With the help of effective knowledge distillation methods, we can significantly improve the pruned model performance. To maximize the use of both methods, We carefully designed the method framework and the training pipeline. An end-to-end high efficiency model compression method is proposed in the paper.

Contributions:

- We proposed a new pruning pipeline combined with knowledge distillation, in which knowledge is transferred from un-pruned network to pruned network.
- We demonstrated the results of choosing different networks as the teacher when practicing knowledge distillation. It is verified that original model as the teacher in our method perform better than other irrelevant models as the teacher.
- We verified the effectiveness of the proposed pipeline with different knowledge distillation methods and pruning methods. Experiments show that our proposed method increased accuracy of several pruned networks by 0.5% to 1.5%, compared with traditional methods.

The rest of this paper is organized as follows. In "Related works", various researches related to our work are introduced. Then detailed methodologies and the proposed framework are described in "Preliminary" and "Main method" sections. In the 'Experiment' section, we introduce the experiment and demonstrate the result of the proposed method. Finally, in 'Discussion' and 'Conclusion', we make a conclusion and present further thinking about this work.

Related works

Model compression

With the increasing amount of network parameters, various model compression strategies have been proposed: model pruning, parameter sharing, knowledge distillation, quantization, and low-rank decomposition [3]. And these various strategies have achieved excellent results. In recent years, in the study of model compression to reach higher performance, methods combining different strategies have been gradually proposed. [10] combines pruning, quantization, and Huffman encoding to compress the model parameters by over 40×. Wei[32] applies knowledge distillation to quantization to improve accuracy. However, they did not consider improving the performance of pruning by incorporating knowledge distillation into the pruning process.

Network pruning

As early as 1990, Lecun [21] proposed the idea of pruning parameters, and various pruning methods and pipelines were brought out. The most used strategy is the saliency-based method [8], which is to, sort parameters by importance, and then remove the less important parts of parameters. The evaluation method of "the importance of parameters" has become the object of study by scholars. A simple method is to define the absolute value of weight as a measure of its importance [22]. Another essential issue in pruning is that when the model is undergoes pruning, it generally brings accuracy loss, so we also need to consider the recovery of accuracy while pruning. The traditional one-shot pruning process generally recovers the accuracy by the three-stage method "training, pruning, and fine-tuning," as shown in Fig. 1. Whereas it is difficult to recover the accuracy when too many parameters

Training
Pruning
Fine-Tuning

Fig. 1 A typical three stage pruning pipeline. A network is first well trained in a target dataset, and then the pre-trained model is pruned with certain method. After pruning, the pruned model is fine-tuned, in which the model is retrained with lower learning rate on the dataset

are removed at one time. The iterative pruning method is then proposed, in which the pruning is not done at once. In iterative pruning, every step, the model is slightly pruned and retrained for a few times. The pruning and retraining process is iteratively done many times to recover a better accuracy, such as in [37] Zhu proposes automated gradual pruning with an iterative process.

Knowledge distillation

Knowledge distillation is a knowledge transfer technology widely used in computer vision [23], natural language processing [24], and other deep learning fields. The vanilla knowledge distillation strategy was proposed by Hinton in 2015 [14]. In the vanilla method, the softened outputs of the logits layer of a robust, high-accuracy, and well pre-trained network, are used to guide and supervise the outputs of the student network (often a smaller network). It is considered that the dark knowledge hidden in the output of the teacher network's logits layers is used to improve the student network's performance. Knowledge distillation has achieved outstanding results. In the continuous development, response-based knowledge, feature-based knowledge, relation-based knowledge [6], and other knowledge distillation methods based on different knowledge have been gradually proposed. Despite the different knowledge definitions and distillation methods, the goal is similarly to approximate the representation of the student network to the teacher network. When it comes to the effects of knowledge definition, the structural differences between the networks are very important. [25] also finds that networks with similar structures are easier to transfer knowledge. Therefore, in this paper, the knowledge distillation between the sub-networks of the original network is used to minimize the structural differences.

Preliminary

In this section, we will introduce the detailed preliminary methodology used in our method and experiment. This section starts from two parts: the methodology used for pruning, and the methodology of knowledge distillation.

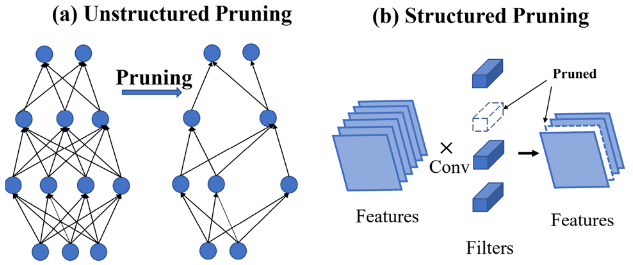

Fig. 2 The schematic illustration of two types of network pruning methods according to granularity. **a** In unstructured pruning, neuron leveled connections are pruned. **b** A type of structured pruning on the right shows a example that filters are pruned

Pruning methods

The goal of pruning is to reduce the maximum amount of parameters with a certain small loss of precision, or the least loss of precision with a small specific amount of parameters. When parameters are pruned, they are removed from a model and will not participate in the reference process. Generally more parameters are pruned, the model becomes more efficient and small. However when more parameters are pruned, the model performance are more affected and the accuracy decreases more. So it is important to balance the degree of pruning and loss of accuracy. The optimization target is shown as:

$$\min_{\omega} \mathcal{L}\left(\mathcal{D};\omega\right) \quad \text{s.t.} \quad \|\omega\|_0 < \kappa, \tag{1}$$

where ω denotes the weight of the network. $\mathcal{D}$ denotes the dataset. $\mathcal{L}$ is the loss function of the network. κ is a hyperparameter given by us to limit the number of non-zero weight parameters of a model. When κ is set small, the model becomes more efficient and has a smaller size. Besides, a sparsity ratio, which means the proportion of parameters with a value of 0 to the total parameters, is also used to describe the sparsity of the model.

According to the granularity, pruning can be mainly divided into structured pruning and unstructured pruning as shown in Fig. 2. In unstructured pruning, the connections at the level of individual neurons are pruned. In structured pruning, on the other hand, larger parts, such as filters, channels, and layers are pruned.

A rigorously done research [4] found that unstructured pruning can find better sub-networks, and the unstructured pruning preserves more local structure information of the original network than structured pruning. Noting that the remaining structure of pruned sub-network is essential in our experiment, only unstructured pruning experiment was implemented in our paper.

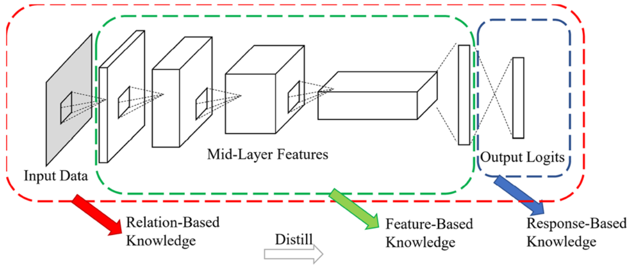

Fig. 3 Three types of knowledge in knowledge distillation. Response-based knowledge is distilled from outputs logits layers; feature-based knowledge is distilled from middle layers; relation-based knowledge is often distilled from some relational representation of the whole network

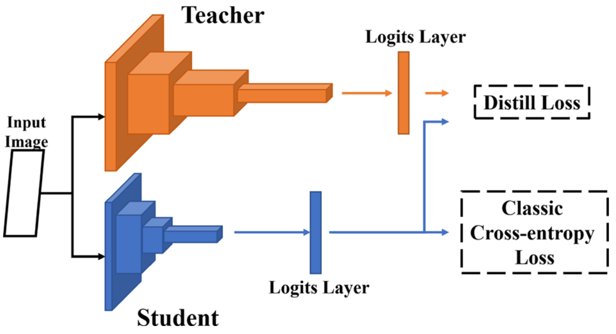

Fig. 4 Schematics of vanilla knowledge distillation for a pre-trained teacher network and a smaller student network. The same image data is input into the two networks, and then the different results generated are used to calculate the loss function

In our experiment, for unstructured pruning, we first obtain a sparse model by training with ℓ_2 regularization. Then we used magnitude-based weight pruning to perform global unstructured pruning with specified sparsity, and remove the weight parameters with a small ℓ_1-norm.

Knowledge distillation methods

Based on different knowledge, knowledge distillation are mainly divided into three categories [6]: response-based knowledge, feature-based knowledge, and relation-based knowledge [6]. An illustration of three different knowledge is shown in Fig. 3. We applied the three types of knowledge distillation methods separately.

Response-based knowledge

Response-based knowledge makes the most of the knowledge from neural units response. The main goal is to let the output of the student network to mimic the output of the teacher network. For this method, we apply the vanilla and also very effective knowledge distillation method by Hinton [14], as shown in Fig. 4.

The training loss for knowledge distillation usually has two parts: classic cross-entropy loss and distill loss. Classic cross-entropy loss is the classic loss used for classification model training and can be calculated as the cross-entropy of softmax outputs of neural networks and grand truth labels. Distill loss is the knowledge distillation loss between the student network and the teacher network. Distill loss is carefully designed and varies with the definition of knowledge. The two parts loss functions for vanilla knowledge distillation are defined below:

Classic cross-entropy loss:

$$\mathcal{L}(\mathbf{W}_\mathrm{S}, x) = \mathrm{CrossEntropy}(z_\mathrm{S}, y_\mathrm{true}) \tag{2}$$

Distill loss:

$$\mathcal{L}_\mathrm{Distill} = \tau^2 \cdot \mathrm{KLdiv}(Q_\mathrm{S}, Q_\mathrm{T}) \tag{3}$$

In classic cross-entropy loss, x and y_true respectively denote the input image data and the corresponding ground truth labels, and W_S denotes the student network weight parameters, z_S denotes the logits output of the student networks. In distill loss, a smoothed probability output Q_s of the student network is first calculated:

$$Q_\mathrm{S} = \frac{\exp(z_\mathrm{S}/\tau)}{\sum_{i=1}^{n} \exp\left(z_\mathrm{S}^{(i)}/\tau\right)} \tag{4}$$

In which $z_\mathrm{S} \in \mathbf{R}^n$ is the logits output of the student network. Then Eq. 4 is a softmax function with additional parameter: temperature τ. τ controls the degree of smoothness, and when τ=1 Eq. 4 becomes a simple softmax function. Similarly, the smoothed probability output of the teacher network Q_T can be calculated. Here smoothed probability Q_S and Q_T are used instead of direct outputs of networks to better transferring knowledge. In the two loss functions, the cross-entropy loss helps the student network learn to predict the correct label of input images, then the distill loss is to make sure the outputs probability distribution of the student network imitates the outputs of the teacher network.

Then the total loss for vanilla knowledge distillation is the weighted sum of the two losses below. α denotes the weight hyperparameter and the total loss can be defined as:

$$\mathcal{L}_\mathrm{KD} = \alpha\mathcal{L}(\mathbf{W}_\mathrm{S}, x) + (1-\alpha)\tau^2 \cdot \mathrm{KLdiv}(Q_\mathrm{S}, Q_\mathrm{T}) \tag{5}$$

Feature-based knowledge

Feature-based knowledge regards the feature layer of the middle layer of the deep neural network as the knowledge that needs to be transferred, and aims that the student network directly fits the feature layer of the teacher network. For such methods, we use an attention-map based method

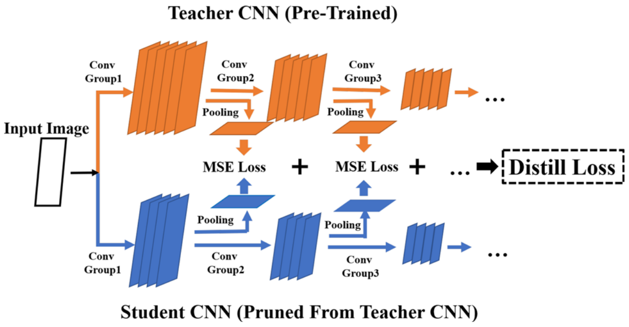

Fig. 5 Schematics of a feature-based knowledge distillation for two convolutional neural networks. The same image is input into two networks, and networks generate several feature maps after different convolution layer groups. The attention maps composited by the feature maps are used to calculate MSE (mean square error) loss

that Zagoruyko and Komodakis [35] proposed, as shown in Fig. 5.

In this method, the MSE of attention maps at different convolution stages between two networks are defined as distill loss. The MSE loss can be given by:

$$\mathcal{L}_{\text{attention\_MSE}} = \frac{1}{2}\sum_{j\in\mathcal{I}}\left\| \frac{Q_{\mathrm{S}}^{(j)}}{\|Q_{\mathrm{S}}^{(j)}\|_2} - \frac{Q_{\mathrm{T}}^{(j)}}{\|Q_{\mathrm{T}}^{(j)}\|_2} \right\|_p, \tag{6}$$

where $Q_{\mathrm{S}}^{(j)} = \text{vec}\left(F\left(A_{\mathrm{S}}^{(j)}\right)\right)$ and $Q_{\mathrm{T}}^{(j)} = \text{vec}\left(F\left(A_{\mathrm{T}}^{(j)}\right)\right)$ are respectively the j-th pair of student and teacher attention maps in vectorized form, and $A_{\mathrm{S}}^{(j)}$ and $A_{\mathrm{T}}^{(j)}$ respectively denote the activation tensors of student and teacher network. Then Eq. 6 is to calculate L_2 distance for the mid-layer features of two networks. Therefore to minimize this distance, the student network will learn modeling the same feature as the teacher network. As the same $\mathcal{L}(\mathbf{W}_{\mathrm{S}}, x)$ denotes the cross-entropy loss with one-hot labels, which is defined in Eq. 2. Let β be the weight hyperparameter, and the total attention transfer knowledge distillation loss can be defined as:

$$\mathcal{L}_{\mathrm{AT}} = \mathcal{L}(\mathbf{W}_{\mathrm{S}}, x) + \frac{\beta}{2}\sum_{j\in\mathcal{I}}\left\| \frac{Q_{\mathrm{S}}^{(j)}}{\|Q_{\mathrm{S}}^{(j)}\|_2} - \frac{Q_{\mathrm{T}}^{(j)}}{\|Q_{\mathrm{T}}^{(j)}\|_2} \right\|_p \tag{7}$$

Relation-based knowledge

Relation-based knowledge does not focus on the value of certain layers but explores the relationship between different sample data or network feature layers. In our paper, similarity-preserving knowledge distillation method [30] is used, as is shown in Fig. 6. In this method a mini-batch of images is input into networks. Every input image generates several feature maps. Then similarities are calculated between feature maps from every two input images of a mini-batch and a $b \times b$ (b denotes the batch size) similarity matrix was generated. The teacher network and student network has respective similarity matrices, and the MSE of these two matrices is considered as the distill loss.

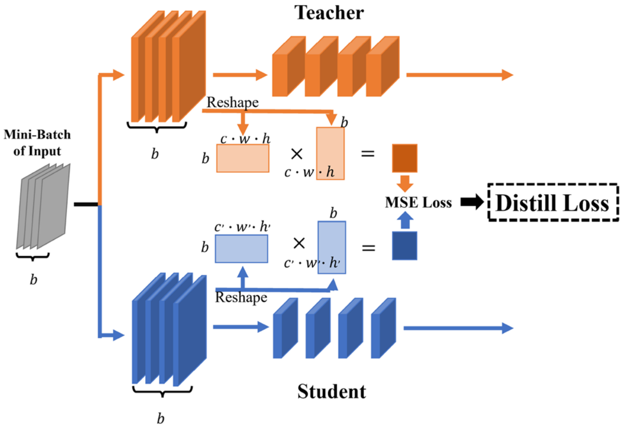

Fig. 6 Schematics of transferring knowledge from a pre-train teacher network to a small student network with similarity-preserving knowledge distillation method. A mini-batch of image are input into two networks. The mid-layer activation tensors are reshaped to a feature matrix with a size of $b \times (c \cdot w \cdot h)$. Matrix multiplication is done between the feature matrix, and its transposition and a similarity matrix is obtained. The MSE Loss is calculated between similarity matrices from two networks

The loss of similarity-preserving knowledge distillation can be defined as:

$$\begin{aligned} &\mathcal{L}_{\mathrm{SP}}(Q_{\mathrm{T}}, Q_{\mathrm{S}}) \\ &\quad = \mathcal{L}(\mathbf{W}_{\mathrm{S}}, x) \\ &\qquad + \beta \cdot \frac{1}{b^2}\sum_{(l,l')\in\mathcal{I}}\left\| Q_{\mathrm{S}}^{(l)} \cdot Q_{\mathrm{S}}^{(l)\top} - Q_{\mathrm{T}}^{(l)} \cdot Q_{\mathrm{T}}^{(l)\top} \right\|_F^2 \end{aligned} \tag{8}$$

where $Q_{\mathrm{T}}^{(l)} \in \mathbf{R}^{b\times(c\cdot w\cdot h)}$ is a reshaping of $A_{\mathrm{T}}^{(l)}$, and $A_{\mathrm{T}}^{(l)} \in \mathbf{R}^{b\times c\times w\times h}$ denote the after-activation feature map of a particular layer l. b denotes the batch size while training. $\mathcal{L}(\mathbf{W}_{\mathrm{S}}, x)$ denotes the cross entropy loss defined in Eq. 2.

Main method

In this section, we introduce the proposed framework of recovering pruned models with knowledge distillation. A schematic illustration is shown in Fig. 7. In the part of network pruning, an over-parameterized network is well-trained on a given dataset, and then the pre-trained model is pruned to a smaller model. In the part of knowledge distillation, the pre-trained model is seen as the teacher and the pruned model as the student. Different knowledge distillation methods are

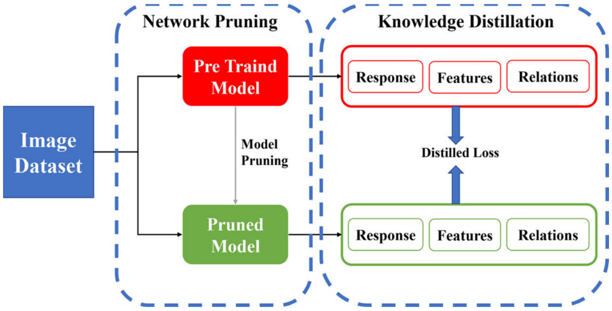

Fig. 7 Proposed framework of recovering pruned model with knowledge distillation

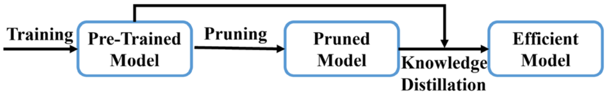

Fig. 8 The pipeline of one-shot process pruning with knowledge distillation framework

used and different knowledge is transferred from the teacher model to the student model. To apply our proposed method to the retraining process of the pruned model and recover the accuracy, we add the distill loss to the original loss of fine-tuning.

The proposed method can be seen as a self knowledge distillation framework with pruning. The whole process starts with one pre-trained model, and no other models are used. All the intermediate models are generated from the same original model, so they have high similarity and compatibility in structure and parameter weights. Then the knowledge distillation is practiced between the models at different stages of pruning, and knowledge at different time steps in the process is transferred. Due to the similarity of the intermediate models, the dark knowledge from the original model helps to recover a better final model more easily.

There are two kinds of training strategies: one-shot pruning and iterative pruning. In this paper, both the one-shot process and iterative process are applied in our experiment. The pipeline of one-shot process is shown in Fig. 8. A model is well trained, and pruned to obtain a small student model with certain sparsity. Then we retrain the pruned model with knowledge distillation, in which the original model is the teacher. The pipeline of iterative process is shown as algorithm flow in Algorithm 1. In iterative process, pruning and retraining are iteratively performed. Each time a part of the parameters is pruned, the model will be retrained to recover before the next pruning step. These two steps are performed several times until the model reaches the target sparsity.

Algorithm 1 Framework of Iterative Pruning and Knowledge Distillation.

```
Input: Pre-trained model $T$;
   student model sparsity ratio at step $t$: $s_t\%$;
   final sparsity ratio $s_f\%$;
   training epoch between two pruning step $n_1$;
   additional retraining epoch after pruning $n_2$;
Output: efficient model $S$
# Initialization
$S \leftarrow T$
while sparsity of S: $s_t\% > s_f\%$ do
    prune $(s_{t+1} - s_t)\%$ weight parameters of $S$;
    for $i = 1$ to $n_1$ do
        retraining model $S$ with distillation loss with $T$
    end for
    $t \leftarrow t + 1$
end while
for $epoch = 1$ to $n_2$ do
    retraining model $D$ with distillation loss with $T$
end for
return $S$;
```

Table 1 baseline Top-1 accuracy of ResNet32x4, VGG13, and WRN_40_2 on CIFAR100 dataset

Model	ResNet32x4	VGG13	WRN_40_2
Top-1 acc. (%)	79.42	74.64	75.61

Experiment

Knowledge distillation compared to fine-tuning in retraining

We choose three networks to prune: simple CNNs (VGG13), residual network v1 (ResNet32x4), and Wide ResNet (WRN _40_2) [11,28,36]. We use the pre-trained model of these networks from [29]. We perform all the experiments on CIFAR100 dataset as all the networks above can't reach extremely high accuracy on this dataset, and the Top-1 accuracy is listed in Table 1. Different target sparsity is used as a goal to verify the completeness. To validate the effectiveness , we chose three very high sparsity: 0.9, 0.95, and 0.975. In these sparsity, the network capacity is definitely abated. On the retraining process, the step decay learning rate is used. The learning rate is initialized as 0.001, and reduced by 50% every 40 epochs. The max retraining epoch is set to 120. The optimizer is SGD with weight-decay=5e−4 and momentum = 0.9. All the experiments are done in Pytorch 1.5, CUDA 10.1 and CUDNN 7.6.5, with NVIDIA GeForce GTX 1080Ti(Pascal) GPU and Intel i7 9700k CPU.

For the hyperparameters in knowledge distillation loss function, we choose them with common used values based on previous studies [27,29]. In Eq. 5 the weight hyperparameter α is set to 0.9, and temperature τ is set to 4. In Eq. 7 we set β = 100. In Eq. 8 we set β = 1000.

Table 2 The Top-1 accuracy on CIFAR100 dataset of three different networks, the networks were one-shot pruned to a sparsity of 90%, 95% and 97.5% (percentage of weight parameters pruned), and then retrained by 4 different methods: fine-tuning with grand truth labels (FT), vanilla knowledge distillation (KD), attention-map based knowledge distillation (AT), similarity-preserving knowledge distillation (SP)

Models	Top-1 acc.(%)		
	Sparsity = 90%	Sparsity = 95%	Sparsity = 97.5%
ResNet32x4 pruned, FT	77.270	75.990	73.770
ResNet32x4 pruned, KD	78.550	76.270	**74.260**
ResNet32x4 pruned, AT	**78.610**	**77.000**	73.900
ResNet32x4 pruned, SP	78.100	76.390	73.940
VGG13 pruned, FT	73.890	72.540	68.510
VGG13 pruned, KD	73.430	72.700	**69.490**
VGG13 pruned, AT	73.460	71.270	64.630
VGG13 pruned, SP	**74.070**	**73.120**	67.190
WRN_40_2 pruned, FT	73.050	71.140	65.490
WRN_40_2 pruned, KD	**73.650**	**71.300**	**66.240**
WRN_40_2 pruned, AT	73.440	70.720	64.700
WRN_40_2 pruned, SP	73.230	70.500	63.890

And FT is the traditional fine-tuning method. The results in bold are the best result of 4 methods for the same pruned model and sparsity

One shot pruning

In one-shot pruning, we pruned the pre-train models to a certain sparsity (0.9, 0.95, 0.975) once, and then retrained the pruned model with fine-tuning method (FT), vanilla knowledge distillation (KD), attention-map based knowledge distillation (AT), similarity-preserving knowledge distillation (SP). The overall results are shown in Table 2.

Results showed that in different CNNs, the pruned models retraining with knowledge distillation strategy outperformed the pruned models only fine-tuned (FT). The vanilla knowledge distillation method (KD) had stable performance on improving accuracy. However attention-based knowledge distillation method (AT) and similarity-preserving knowledge distillation (SP) respectively fitted ResNet32x4 and VGG13, and performed poorly when models were heavily pruned (sparsity over 97%).

We also visualized the activation attention maps of the mid-layer in Fig. 12. The graph shows that the activation attention maps of pruned-retrained models evidently changed from the original model. The model pruned and retrained with fine-tuning seems less likely to focus on the momentous information on the input images. However when retraining with knowledge distillation strategy, the new attention maps more closely resembled the original attention maps. The attention maps reconstruction effect performs the best on the AT method as the original attention maps are directly used to guide the retraining of pruned model.

Iterative pruning

In iterative pruning, we performed the pruning every 5 epochs for 30 times in total, and the density for parameters exponential declined. So the sparsity can be calculated as:

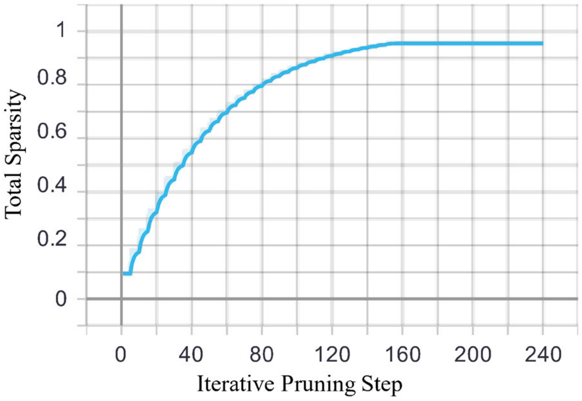

Fig. 9 The model sparsity during iterative pruning process. The sparsity is finally stabilized at 0.95 at step 150

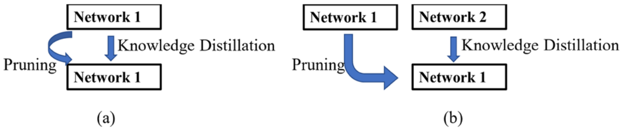

Fig. 10 **a** The proposed method of training a pruned model with knowledge distillation from the original model. **b** Simply combine pruning and knowledge distillation

$$s_{\mathrm{t}} = 1 - (1 - s_{\mathrm{f}})^{t-\alpha} \tag{9}$$

s_{f} donates the final sparsity ,which is set to 95%, s_{t} is the model sparsity after pruned for t times. the course of sparsity during the iterative pruning is shown in Fig. 9. The sparsity ratio increases following the Eq. 9, and reaches the highest value 0.95 at epoch 150. Then sparsity does not change in the remaining training process.

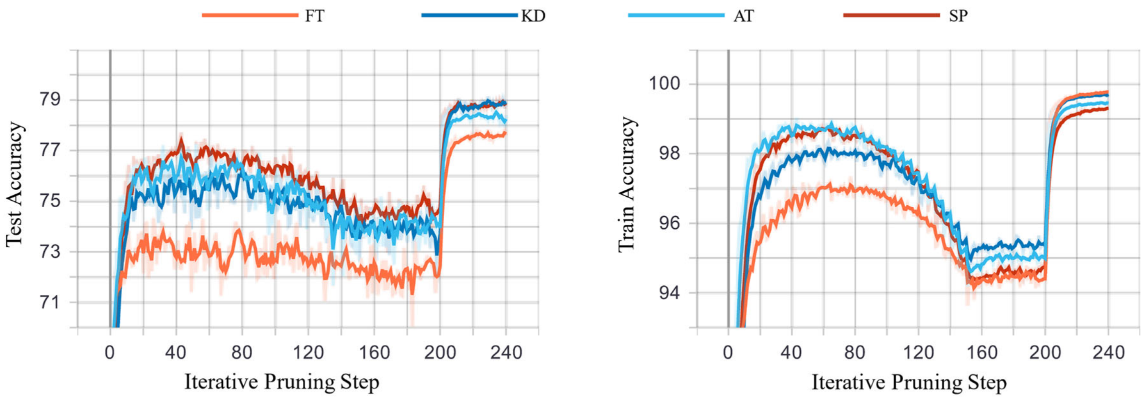

Fig. 11 The validation Top-1 accuracy and training accuracy of retrain with fine-tuning (FT), vanilla knowledge distillation (KD), attention-based knowledge distillation (AT), similarity-preserving knowledge distillation (SP) in iterative pruning process

Table 3 The baseline accuracy of two different residual network on CIFAR100

Model	ResNet32x4	ResNet50
Top-1 acc. (%)	79.42	79.34

Figure 11 shows the Top-1 accuracy results of training-dataset and testing-dataset during iterative pruning. At the last step, the test accuracy of three knowledge distillation methods all achieves higher results than the test accuracy with fine-tuning. However, fine-tuning strategy top the training accuracy at the last step. It is considered that with knowledge distillation in iterative pruning, the degree of overfitting is reduced, and it is because the models relearned the dark knowledge lost in the pruning process by knowledge distillation.

Knowledge distillation with different teacher model

In this part, we experiment on the selection of the teacher network. In the previous sections, we mentioned that in the proposed pipeline, the original model with the original parameters is used as the teacher network in knowledge distillation, because with a similar network structure and weight parameters, the pruned model can better recover. The difference of proposed method and simply combine pruning and knowledge distillation sketched in Fig. 10.

To validate the effect of different teacher models, two different residual network was chosen: (1) ResNet32x4, (2) ResNet50. Resnet32x4 is ResNet v1 with 32 layers and channels widen to 4x, and ResNet50 is ResNet v2 [12]. These two networks have almost the same accuracy as shown in Table 3. Then the student networks are both pruned from ResNet32x4. Iterative pruning method was used, and all the experimental hyperparameter configurations are the same as the iterative pruning in "Knowledge distillation compared to fine-tuning in retraining" subsection. The results are shown in Table 4.

Table 4 The Top-1 accuracy(%) on CIFAR100 dataset with different models as the teacher

Training method	Teacher model	
	ResNet32x4 (Original model)	ResNet50
KD	78.960	78.640
AT	78.400	77.500
SP	78.940	78.330

We can notice that the pruned model retrained with original pre-pruned models as teacher performs better. When retraining with other models, the accuracy significantly decreased, and especially with attention-based method, accuracy reduced to even lower than retraining by fine-tuning. The pruned models performed to be more difficult to mimic a teacher model which has a different structure than the original model.

Discussion

Knowledge distillation in our proposed method generally helps to recover the pruned models' accuracy. However, results interestingly showed that different knowledge distillation methods suited different architecture of CNNs. For example, attention-based knowledge distillation performs poorly on CNNs without residual connections. Besides, feature-based and relation-based knowledge distillation neither work well on extreme sparsity. The theory for the unexpected result was not thoroughly studied in this paper. Another work to be done is when the teacher and the student have the same structure, whether model weight parameters affect retraining performance. As the Lottery Ticket Hypothesis [4] shows that the initial weight parameters lead the training process, carefully selected parameters may also improve the result of pruned models.

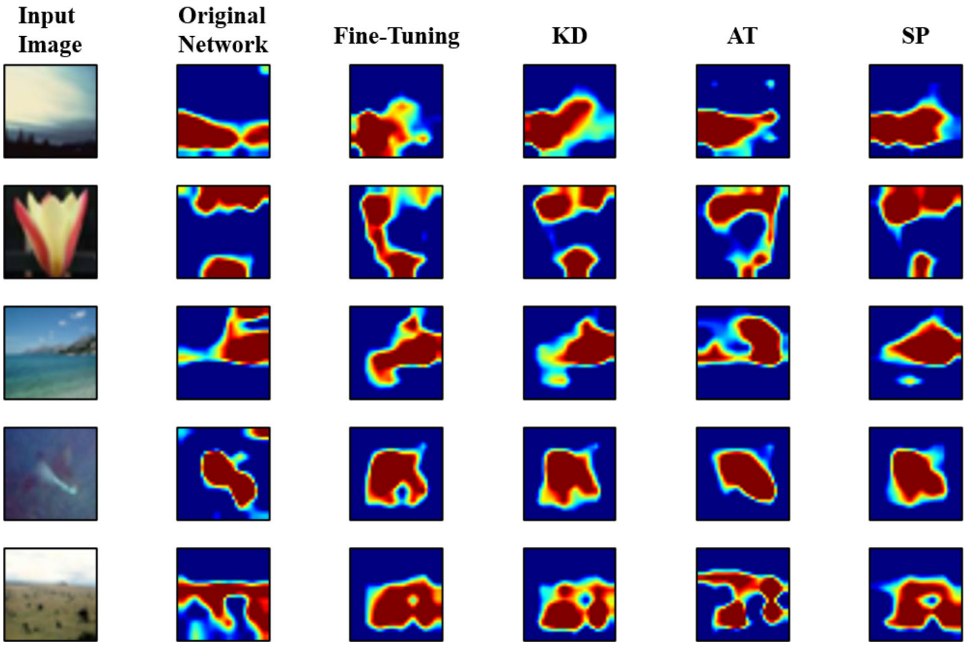

Fig. 12 The activation attention maps of the mid-layer in VGG13 (sparsity = 95%). The first column shows input images, and results of the original unpruned network corresponded to column2. Results of retrained by fine-tuning are corresponded to column3, and the last 3 columns are results of the three knowledge distillation methods. All the attention maps are converted into thermal maps and then displayed

Conclusion

In this work, we focused on the enduring question of how to recover pruned models. Knowledge distillation is used as a perfect tool for transferring knowledge, increasing accuracy, and compressing models. We proposed an improved integrated framework of pruning combining knowledge distillation strategy, and experimented on different image classification CNNs with three knowledge distillation methods. The result showed that knowledge from the original network of various forms helped the pruned network recover with higher accuracy. We also observe that different knowledge distillation methods suited different architecture of CNNs. We hope the future work of knowledge distillation interpretability could explain the mechanism and extend the application of knowledge distillation in other fields.

Acknowledgements This work was supported by the Open Fund of Shenzhen Institute of Artificial Intelligence and Robotics for Society(AIRS) (Grant no. AC01202005016), Aeronautical Science Foundation of China (Grant no. 2019ZE057001) and Shanghai Rising-Star ProgramShanghai Rising-Star Program (Grant no. 20QC1401100). Liyang Chen and Yongquan Chen are co-first authors.

Compliance with ethical standards

Conflict of interest On behalf of all authors, the corresponding author states that there is no conflict of interest.

References

1. Belkin M, Hsu D, Ma S, Mandal S (2019) Reconciling modern machine-learning practice and the classical bias-variance trade-off. Proc Natl Acad Sci 116(32):15849–15854
2. Chen H, Wang Y, Xu C, Xu C, Tao D (2020) Learning student networks via feature embedding. IEEE Trans Neural Netw Learn Syst 30:1928–1942
3. Denil M, Shakibi B, Dinh L, Ranzato M, De Freitas N (2013) Predicting parameters in deep learning. Adv Neural Inf Process Syst 2013:2148–2156
4. Frankle J, Carbin M (2018) The lottery ticket hypothesis: finding sparse, trainable neural networks. In: International conference on learning representations (ICLR)
5. Furlanello T, Lipton ZC, Tschannen M, Itti L, Anandkumar A (2018) Born again neural networks. In: International conference on machine learning (ICML)
6. Gou J, Yu B, Maybank SJ, Tao D (2020) Knowledge distillation: a survey. arXiv:2006.05525
7. Guo Y, Yao A, Chen Y (2016) Dynamic network surgery for efficient DNNs. Adv Neural Inf Process Syst 2016:1379–1387
8. Hagiwara M (1993) Removal of hidden units and weights for back propagation networks. In: International joint conference on neural networks (IJCNN)
9. Han S, Pool J, Tran J, Dally W (2015) Learning both weights and connections for efficient neural network. Adv Neural Inf Process Syst 2015:1135–1143

10. Han S, Mao H, Dally WJ (2016) Deep compression: Compressing deep neural networks with pruning, trained quantization and huffman coding. In: International conference on learning representations (ICLR), pp 1–14
11. He K, Zhang X, Ren S, Sun J (2016a) Deep residual learning for image recognition. In: Proceedings of the IEEE conference on computer vision and pattern recognition (CVPR), pp 770–778
12. He K, Zhang X, Ren S, Sun J (2016b) Identity mappings in deep residual networks. In: Proceedings of the IEEE conference on computer vision and pattern recognition (CVPR)
13. Heo B, Kim J, Yun S, Park H, Choi JY (2019) A comprehensive overhaul of feature distillation. In: Proceedings of the IEEE international conference on computer vision (ICCV), pp 1921–1930
14. Hinton G, Vinyals O, Dean J (2014) Distilling the knowledge in a neural network. arXiv:1503.02531
15. Huang Z, Yu Y, Xu J, Ni F, Le X (2020) PF-Net: point fractal network for 3D point cloud completion. In: Proceedings of the IEEE conference on computer vision and pattern recognition (CVPR), pp 7662–7670
16. Hubara I, Courbariaux M, Soudry D, El-Yaniv R, Bengio Y (2017) Quantized neural networks: training neural networks with low precision weights and activations. J Mach Learn Res 18(1):6869–6898
17. Jia Y, Chen X, Yu J, Wang L, Wang Y (2020) Speaker recognition based on characteristic spectrograms and an improved self-organizing feature map neural network. Complex Intell Syst 2020:1–9
18. Kim J, Park S, Kwak N (2018) Paraphrasing complex network: network compression via factor transfer. Adv Neural Inf Process Syst 2018:2760–27693
19. Kollias D, Tagaris A, Stafylopatis A, Kollias S, Tagaris G (2018) Deep neural architectures for prediction in healthcare. Complex Intell Syst 4(2):119–131
20. Le X, Mei J, Zhang H, Zhou B, Xi J (2020) A learning-based approach for surface defect detection using small image datasets. Neurocomputing 408:112–120
21. LeCun Y, Denker JS, Solla SA (1990) Optimal brain damage. Adv Neural Inf Process Syst 1990:598–605
22. Li H, Kadav A, Durdanovic I, Samet H, Graf HP (2017) Pruning filters for efficient convnets. In: International conference on learning representations (ICCV)
23. Li Z, Hoiem D (2017) Learning without forgetting. IEEE Trans Pattern Anal Mach Intell 40(12):2935–2947
24. Liu J, Chen Y, Liu K (2019) Exploiting the ground-truth: an adversarial imitation based knowledge distillation approach for event detection. Proc Conf AAAI Artif Intell 33:6754–6761
25. Mirzadeh SI, Farajtabar M, Li A, Levine N, Matsukawa A, Ghasemzadeh H (2019) Improved knowledge distillation via teacher assistant. arXiv:1902.03393
26. Park W, Kim D, Lu Y, Cho M (2020) Relational knowledge distillation. In: 2019 IEEE/CVF conference on computer vision and pattern recognition (CVPR)
27. Ruffy F, Chahal K (2019) The state of knowledge distillation for classification. arXiv:1912.10850
28. Simonyan K, Zisserman A (2015) Very deep convolutional networks for large-scale image recognition. In: International conference on learning representations (ICLR)
29. Tian Y, Krishnan D, Isola P (2019) Contrastive representation distillation. arXiv:1910.10699
30. Tung F, Mori G (2019) Similarity-preserving knowledge distillationn. In: Proceedings of the IEEE international conference on computer vision (ICCV), pp 1365–1374
31. Turc I, Chang MW, Lee K, Toutanova K (2019) Well-read students learn better: the impact of student initialization on knowledge distillation. arXiv:1908.08962
32. Wei Y, Pan X, Qin H, Ouyang W, Yan J (2018) Quantization mimic: towards very tiny cnn for object detection. In: Proceedings of the European conference on computer vision (ECCV), pp 267–283
33. Wu X, He R, Hu Y, Sun Z (2020) Learning an evolutionary embedding via massive knowledge distillation. Int J Comput Vis 128:2089–2106
34. Yu Y, Huang Z, Li F, Zhang H, Le X (2020) Point encoder gan: a deep learning model for 3D point cloud inpainting. Neurocomputing 384:192–199
35. Zagoruyko S, Komodakis N (2016) Paying more attention to attention: improving the performance of convolutional neural networks via attention transfer. arXiv:1612.03928
36. Zagoruyko S, Komodakis N (2017) Wide residual networks. In: Proceedings of the IEEE conference on computer vision and pattern recognition (CVPR)
37. Zhu M, Gupta S (2018) To prune, or not to prune: Exploring the efficacy of pruning for model compression. In: International conference on learning representation (ICLR)