Journal of Computers Vol. 35 No. 4, August 2024, pp. 109-122
doi: 10.53106/199115992024083504008

Combined Knowledge Distillation Framework: Breaking Down Knowledge Barriers

Shuiping Ni, Wendi Wang*, Mingfu Zhu, Xinliang Ma, and Yizhe Zhang

School of Computer Science and Technology, Henan Polytechnic University, Jiaozuo 454000, China

nishuiping@hpu.edu.cn, wd629@home.hpu.edu.cn, zhumingfu@zzu.edu.cn, {114086722, 1582837102}@qq.com

Received 6 November 2023; Revised 19 February 2024; Accepted 14 March 2024

Abstract. Knowledge distillation, one of the most prominent methods in model compression, has successfully balanced small model sizes and high performance. However, it has been observed that knowledge distillation predominantly focuses on acquiring knowledge concealed within the dataset and the external knowledge imparted by the teacher. In contrast, self-distillation concerns itself with the utilization of internal network knowledge. Neither approach fails to fully harness the potential of knowledge. Therefore, this paper introduces the combined knowledge c framework that combines knowledge distillation with self-distillation. Within this framework, we introduce multiple shallow classifiers, combined with an attention module, to exploit internal and external knowledge. To enhance the efficiency with which the network utilizes knowledge. Experimental results demonstrate that by comprehensively leveraging network knowledge, distillation effectiveness can be enhanced, resulting in further improvements in network accuracy. Additionally, we applied the framework to lightweight neural networks with group convolution, the framework continues to perform exceptionally well.

Keywords: model compression, knowledge distillation, self-distillation, lightweight, neural network

1 Introduction

As an emerging research field in machine learning, deep learning [1] has witnessed rapid growth in recent years. Today, it has achieved significant breakthroughs in various domains such as image classification [2, 3], object detection [4, 5], and natural language processing [6, 7]. However, with the increasing demands for performance in classification tasks, deep learning models have become progressively more complex, resulting in a rapid surge in the number of model parameters. While the increase in parameters and computational complexity has led to improvements in model accuracy, it has also introduced a series of challenges. Among these are the substantial increase in computational overhead and extended response times.

In response to these challenges, researchers have proposed two primary solutions. The first approach involves high-performance hardware acceleration [8, 9], while the second focuses on model compression [10, 11]. Complex models have the potential to excel on high-performance hardware. However, deploying these models on resource constrained devices remains a considerable challenge. Consequently, model compression has become a focal point of current research. Current model compression techniques primarily include network pruning [12], quantization [13], and knowledge distillation [14]. As one of the widely adopted model compression approaches, knowledge distillation was first introduced by Hinton in 2015 [15] and found its initial applications in image classification tasks. The core concept involves building a lightweight student model first. Then, pre-train a high-performance teacher model to guide the student model. During training, the teacher model only performs inference, and does not update parameters. However, knowledge distillation has certain drawbacks. The teacher model can only conduct offline distillation and focuses solely on outputs. It cannot utilize its internal information to guide the student. Additionally, when there's a gap in abilities between the teacher and student models, the training results may not be ideal.

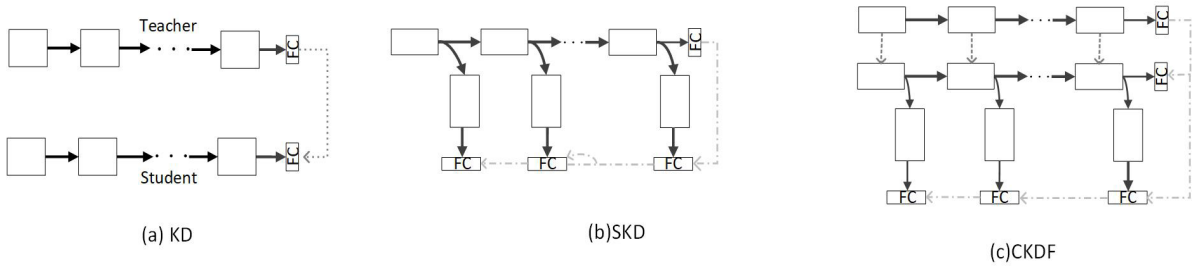
To overcome the limitations of knowledge distillation, researchers have proposed online knowledge distillation. The teacher and student models update parameters concurrently, continuously updating the acquired knowledge. While this addresses the issue of offline distillation's inability to dynamically update, it also presents

* Corresponding Author

additional challenges. While online learning enhances model performance, it also requires more computational resources. Fail to achieve the effect of model compression.

As a specific case of knowledge distillation, self-distillation [16, 18] involves a single network model serving as both the teacher and the student. By iteratively learning the knowledge within the network. Currently, there are mainly two common approaches to self-distillation. One is to perform mutual distillation through different sample information. The other involves self-distillation among different network layers within a single network. Typically, deep networks are used to guide shallow networks.

The introduction of self-distillation [17, 19] effectively addresses the gap in capabilities between teacher and student models. Additionally, it does not require a large amount of additional resources. However, it is evident that during training, the student network can only acquire knowledge from within the network. Therefore, the limitations of self-distillation are also very apparent. It excessively emphasizes the absorption of internal network knowledge, lacking the application of rich external knowledge. Unable to acquire stronger generalization capabilities.



(a) Complex model guiding simple model (b) Deep classifier guiding shallow classifier (c) Combined Knowledge Distillation scheme

Fig. 1. Comparison diagram of three distillation methods
(Black arrows indicate the direction of progress, and dashed lines represent the distillation paths.)

Based on these discoveries, the paper designs a combined knowledge distillation framework (CKDF). Introducing attention modules and shallow layers. It constructs multiple shallow classifiers to establish a self-distillation framework. Simultaneously, it employs pre-trained neural network models to guide the self-distillation framework. Achieving comprehensive utilization of knowledge within the dataset and the neural network, both internal and external. This method enhances network performance while preserving the capabilities of traditional networks. We present a structural comparison diagram of the three distillation methods in Fig. 1. To validate this methodology, we conduct a comparative analysis of knowledge distillation, self-distillation, and the combined knowledge distillation framework, evaluating their respective performance.

The primary contributions of this study are as follows:

(1) To overcome these limitations, we introduce a novel framework: combined knowledge distillation. This framework amalgamates knowledge distillation with self-distillation, offering a fresh perspective on the concept.

(2) We conduct experiments on two distinct datasets using five diverse convolutional neural networks, affirming the universality of our approach.

(3) We apply the framework to lightweight networks, which allows us to reduce network parameters while ensuring significant network performance.

2 Related Works

2.1 Knowledge Distillation

Knowledge Distillation (KD), a commonly employed technique in model compression, has extensive applications in various domains, including natural language processing and computer vision. The inception of knowledge distillation dates back to 2015. It consists of three main components: the teacher, the student, and the distillation algorithm. This approach involves two key steps. First, it designs the creation of a lightweight student

model. Second, the soft label knowledge acquired by the complex teacher model is used as supervision during the training of the student model. The distillation algorithm transfers the knowledge to the student model, guiding it to achieve performance equivalent to or better than the teacher model.

The core of knowledge distillation methods lies in the extraction and learning of knowledge. Knowledge can be categorized into label knowledge and intermediate-level knowledge based on its nature. Label knowledge refers to the softened information of the logits probability distribution in the final output of the network model. Intermediate-level knowledge, on the other hand, refers to the outputs from intermediate layers of the network. Label knowledge has always been a key focus and hot topic in the research of distillation methods. The knowledge distillation originally proposed by Hinton and others utilized label knowledge. As the research has progressed, the applications of label knowledge have become diverse. For example, improving the distillation process, removing interference, and so on. Gao et al. [20] proposed a simple staged distillation approach. Dividing the extraction process into multiple stages and updating only one module of the student network at a time. One by one, each module of the student network is updated until the whole process is completed. According to the research findings of Mirzadeh et al. [21], a high-performing teacher does not necessarily lead to a good student. When the gap between the student and the teacher is too large, it can prevent the student from improving. To address this issue, Mirzadeh et al. introduced the utilization of medium-sized neural networks as part of the distillation. The teacher first transfers knowledge to the teacher assistant, who then conveys this knowledge to the student, bridging the gap between the teacher and the student. Wu et al. [22] proposed peer collaborative distillation, training multiple network branches simultaneously. In this approach, the strong teacher transfers its knowledge of logits to peers for joint learning. This approach aims to enhance model stability and improve the quality of distillation.

After adjusting the "temperature," numerous "dark knowledge" is generated. Extensive research has shown that "dark knowledge" exerts a regularization effect on network learning through softened labels. Consequently, label smoothing regularization can simulate the distillation process. Lee et al. [23] proposed a lightweight distillation framework where the teacher generates soft targets to guide the student in learning both distillation and adversarial losses. This framework leverages the teacher's regularized outputs as prior distributions.

Label knowledge is simple and effective. However, its primary limitation is the limited information it expresses, which is insufficient for complex tasks. Researchers have turned attention to intermediate-layer knowledge in pursuit of more representative knowledge. Intermediate-layer knowledge encompasses features extracted by intermediate modules, providing a richer information set. This significantly enhances the representational power of transmitted knowledge and effectively improves training results. Chen et al. [24] proposed leveraging multi-level information from the teacher model to guide the student. They taught the student network using knowledge obtained from the earlier layers of the teacher model. While intermediate-layer knowledge is rich, it lacks integration, preventing the transmission of complementary knowledge from multiple perspectives.

2.2 Multi-Teacher Distillation

Given the outstanding performance achieved by deep neural networks based on teacher-student models, there has been growing interest in knowledge distillation. Traditional teacher-student models involve a single teacher instructing a single student, but the knowledge transferred by a single teacher may not always be reliable. As a result, researchers are not limited to a single teacher, but are beginning to explore multi-teacher approaches.

In 2017, You et al. [25] proposed using multiple teacher networks to train a single student network. They employed the triplet loss function to transfer information from the final layer logits and intermediate layers of multiple teachers to the student and consider multiple teachers' guidance as equally crucial in training a thin deep network. Subsequently, Liu et al. [26] introduced a novel framework called Adaptive Multi-Teacher Multi-Level Knowledge Distillation (AMTML-KD). In this framework, different weights are assigned to various teachers to integrate their knowledge better and provide high-quality knowledge transfer to the students. This enables students to learn multi-level knowledge from multiple teachers adaptively. In 2023, Pham et al. [27] introduced Collaborative Multi-Teacher Knowledge Distillation (CMT-KD), a method to improve learning efficiency through multi-teacher knowledge distillation and network quantization. Teachers generate shared knowledge during the collaborative learning, facilitate mutual learning among multiple teachers, and guide students. This process also encourages mutual learning between teachers and students, facilitating effective emulation of teachers by students. However, the training complexity of multi-teacher network distillation significantly increases, and different teachers may provide conflicting information or varying levels of domain expertise. This can introduce noise during the knowledge distillation process, impacting the learning of the student model.

2.3 Self-Distillation

The introduction of self-distillation has pioneered a novel distillation approach. The proposal of self-distillation addresses the time-consuming issue brought about by the pre-training of complex teacher models and the problem of mismatched capabilities between teacher and student models, significantly bridging the gap between teacher-student, and thus making a significant contribution to the academic field. Hinton et al. initially proposed the concept of self-distillation.

In 2019, Zhang et al. [28] conducted in-depth research on this concept, and provided a detailed exposition. As a specialized form of knowledge distillation, self-distillation leverages the deep knowledge within a network to guide its training, thereby enhancing model performance. The introduction of self-distillation addressed the time-consuming issue associated with knowledge distillation, where a large teacher model must be trained before the student model, as well as the problem of potential mismatched capabilities between teacher and student models that hinder learning. Furthermore, Zhang et al. [29] introduced attention mechanisms and additional shallow classifiers at different depths, utilizing knowledge transfer within the model to enhance neural network performance, significantly improving the accuracy of branches. They also redesigned the branches and attention mechanisms to enhance branch accuracy further while reducing computational complexity. In 2023, Ni et al. [19] introduced Reverse Self-Distillation (RSD), leveraging both self-distillation and the supervision provided by shallow networks. They compared different-sized self-distillation frameworks and proposed that RSD could alleviate the negative impact of the framework, thereby enhancing the efficiency of varying sizes of self-distillation frameworks. However, self-distillation predominantly focuses on internal knowledge within a single model, constraining its harness of external knowledge sources fully. This limitation may lead to decreased efficiency in knowledge transfer.

All existing works have leveraged knowledge within the network. However, most current methods only extract and learn from a single source of knowledge, neglecting the comprehensive utilization of knowledge. In this paper, we aim to enhance network performance by comprehensively utilizing knowledge both within and outside the neural network, as well as from the dataset.

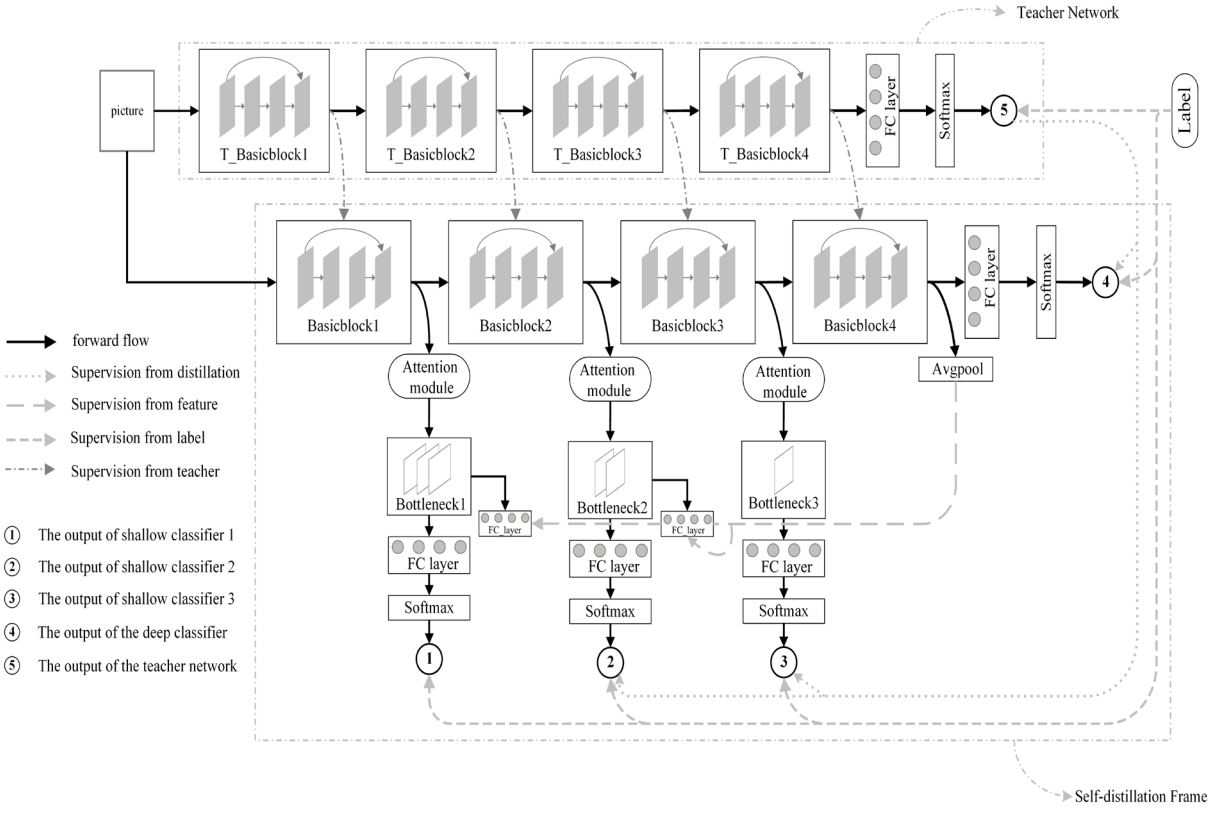


Fig. 2. The combined knowledge distillation framework

3 Proposed Methodology

3.1 Distillation Framework

In this section, we will use ResNet18 as an example to introduce the combined knowledge distillation framework. As illustrated in Fig. 2, CKDF comprises two parts: the teacher network and the self-distillation framework. In the framework, the pre-trained ResNet18 serves as the teacher network to guide the training of the self-distillation framework.

For the self-distillation framework, we consider the untrained ResNet18 as a deep classifier. We then divide it into four Basicblocks according to the network structure of ResNet18, and after each Basicblocki (i=1,2,3), we add relevant modules to construct a shallow classifier. Additionally, after Basicblock4, we append an avgpool layer. The modules related to building the shallow classifier include an attention module, a bottleneck module, and fully connected layers. Shallow classifier j (j=1,2) consists of an attention module, a bottleneck module, and two fully connected layers (FC_layer, FC layer). The output nodes of the FC_layer are equal to the number of output channels from the feature map after passing through Basicblock4 and then through avgpool. The output nodes of the other FC layer equal the number of classes in the dataset. Shallow classifier 3 consists of an attention module, a bottleneck module, and an FC layer. The teacher network, deep classifier, and all shallow classifiers, which have passed through the fully connected layers (FC layers), undergo the softmax function to obtain the respective classifier outputs.

3.2 Relevant Modules

The architecture of the shallow classifier comprises two key components: the attention module and the bottleneck module. The following will provide a detailed explanation of these components.

As depicted in Fig. 3, the structure of the attention module consists of two sets of depthwise separable convolution layers and an upsampling layer. In this module, the first set of depthwise separable convolution layers employs a stride of 2, while the remaining convolutions use a stride of 1. After the input feature map undergoes two groups of depthwise separable convolutions, perform an upsampling operation. Subsequently, the Sigmoid function is used for activation, resulting in the generation of an attention mask. Following that, the attention mask is applied to the input feature map through element-wise multiplication.

The structure of the bottleneck module, as illustrated in Fig. 4, the module is represented by groups, each containing multiple subgroup modules. Each subgroup module consists of two depthwise separable convolution layers. In the first set of depthwise separable convolution layers within each subgroup, the depthwise convolution has a stride of 2, while the stride for the remaining convolutions is 1. The number of subgroup modules depends on the shallow classifier's position. In the bottleneck module of shallow classifier i (i=1,2,3), there are subgroup modules. The output features passing through subgroup modules and are subsequently downsampled to a size of 1×1 using an avgpool layer.

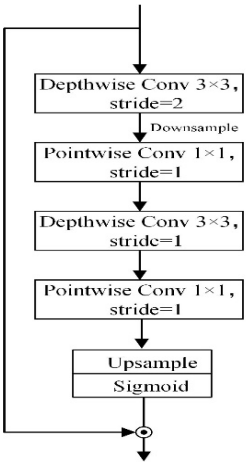


Fig. 3. The structure of the attention module

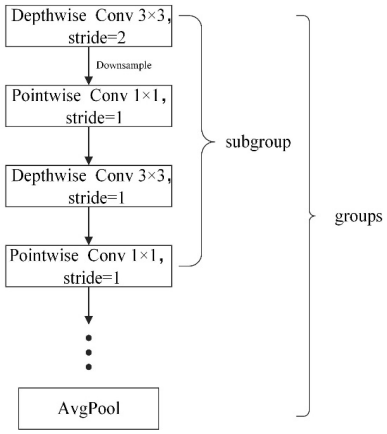


Fig. 4. The structure of the bottleneck module

3.3 The Detail of Distillation

Given N samples of M classes $X=\{x_i\}_{i=1}^{N}$, set the label set as $Y=\{y_i\}_i^M$. In the framework, the shallow classifiers is expressed as $\theta_i(i=1,2,3)$, the deep classifier as θ_4, the teacher network is represented as θ_5. The addition of a softmax layer follows each classifier. Additionally, we introduce the temperature coefficient T to adjust and soften the probability distribution of each classifier's outputs and T is by default set to 1:

$$q_m = \frac{\exp(z_m / T)}{\sum \exp(z_n / T)} , \tag{1}$$

Here z_m represents the output of class m after passing through the fully connected layer of the classifier θ_i, and q_m represents the output probability of class m.

The training process of the CKDF consists of two parts: pre-training of the teacher network and combined training of the self-distillation framework. Before proceeding with the combined knowledge distillation training, the teacher network is pre-trained, where the true labels solely supervise the pre-training process:

$$Cr(q^5, y) , \tag{2}$$

Here q^5 represents the output of the teacher network, y represents the true labels. Cr stands for the cross-entropy loss. After the pre-training of the teacher network is completed, combined training is conducted with the self-distillation framework. To facilitate the network's learning process and balance model performance, we introduce four types of loss functions [30] into the combined knowledge distillation framework, and raise two hyperparameters, α and β, to balance the losses and optimize the distillation process.

loss1: *loss1* is obtained by calculating the cross-entropy loss between the deep and shallow classifier's output, and the true labels:

$$(1-\alpha)\cdot Cr(q^4, y) + (1-\beta)\cdot \sum_{i=1}^{3} Cr(q^i, y) , \tag{3}$$

Here q^i represents the output of the shallow classifier $\theta_i(i=1,2,3)$, q^4 represents the output of the deep classifier. y represents the true labels, and Cr stands for the cross-entropy loss function. Cross-entropy loss is utilized to quantify the disparity between the classifier's output and the true labels, thereby incorporating the knowledge

hidden within the dataset into the classifiers by minimizing cross-entropy loss.

loss2: *loss2* is derived by computing the output of the classifier $\theta_i(i=2,3,4)$ within the self-distillation framework and the output of the teacher network:

$$\alpha \cdot \sum_{i=2}^{4} KL(q^i, q^5) \cdot T^2 \ , \tag{4}$$

Here q^5 represents the output of the teacher network with *T*=3, and q^i represents the output of the shallow and deep classifiers in the self-distillation framework with *T*=3. *KL* refers to the Kullback-Leibler divergence loss function. In *loss2*, we employ soft labels obtained through the teacher network for knowledge transfer, with both deep and shallow classifiers being supervised by the output of the pre-trained teacher model.

loss3: *loss3* is obtained by computing the *L2* loss between the bottleneck layer's output and the output from Basicblock4:

$$\beta \cdot \| F^i - F^k \|_2^2 \ , \tag{5}$$

Here F^k represent the output of the Basicblock4 module in the self-distillation framework after passing through the avgpool layer and $F^i(i=1,2)$ represent the output of the shallow classifier's bottleneck module after passing through the FC_layer. The *L2* loss guides the learning of shallow modules by calculating the distance between the deep and shallow classifier's feature maps. However, feature maps at different depths often have different sizes. Therefore, FC_layers are added after shallow modules to adjust the feature maps, aligning them with the dimensions of deep modules.

loss4: The teacher network is divided into four modules (T_Basicblocki (i=1,2,3,4)) based on its network structure. Similarly, the deep classifier in the self-distillation framework is also segmented into four modules (Basicblocki (i=1,2,3,4)) following the structure of the student network. *loss4* is computed by calculating the mean squared error between the outputs of each module in the teacher network and the corresponding modules in the deep classifier:

$$\beta \cdot \sum_{i=1}^{4} MSE\left(\frac{mean(X_i^2)}{\| X_i \|_2}, \frac{mean(TX_i^2)}{\| TX_i \|_2}\right) \ , \tag{6}$$

Here X_i represents the output of the deep classifier's Basicblocki (i=1,2,3,4) modules, TX_i represents the output of the teacher network's T_Basicblocki (i=1,2,3,4) modules, *MSE* stands for mean squared error and mean represents the operation of taking the mean. Through *loss4*, the output features of each module of the teacher network are introduced into the deep classifier, guiding the deep classifier in self-distillation.

In summary, the $Loss$ for the CKDF can be represented as the sum of the loss for all classifiers:

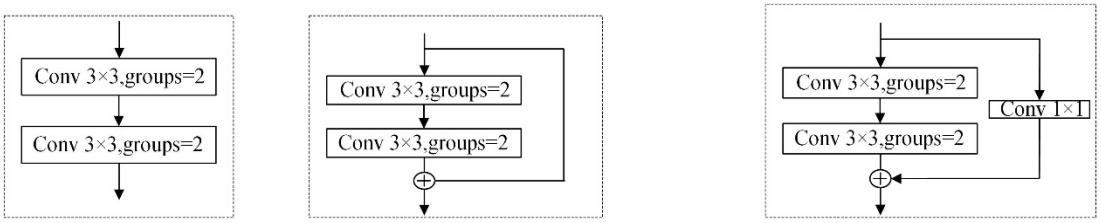
$$Loss = loss1 + loss2 + loss3 + loss4 \ . \tag{7}$$

3.4 Lightweight Module

As shown in Fig. 5, we have performed lightweight design for both the VGG and ResNet18. First, let's delve into the lightweight design for the VGG. Taking VGG11_BN as an example, as shown in Fig. 5(a), it is divided into five modules based on its network structure, with each module containing multiple convolutions. Specifically, the first and second convolution modules incorporate a single 3×3 convolution. In contrast, the third, fourth, and fifth modules contain two 3×3 convolutions. To reduce the number of parameters and computational complexity, we employ group convolution in the latter two convolution modules. This approach segments the input features into multiple groups and conducts independent convolutions within each group. Likewise, for VGG13_BN, VGG16_BN, and VGG19_BN, we divide them into five modules according to their network structure. In VGG13_BN, all five modules have two convolutions. VGG16_BN's first and second convolution modules include two sets of 3×3 convolutions, while the third, fourth, and fifth modules contain three sets of 3×3 convolutions. In VGG19_

BN, the first and second convolutions modules include two sets of 3×3 convolutions, and the third, fourth, and fifth modules contain four sets of 3×3 convolutions. We introduce group convolution for the last two modules of all networks, replacing all convolution layers in the module with group convolutions with a kernel size of 3 and a group count of 2.

Based on the structure of the ResNet18 network, we partition it into four sections and proceed with a lightweight design. The standard residual structure is shown in Fig. 5(b), and the dimension-increasing residual structure is shown in Fig. 5(c). Within the residual structures, we replace the 3×3 convolutions with group convolutions having a kernel size of 3 and a group count of 2, while the 1×1 convolutions remain unchanged.



(a) The group convolution structure of the VGG (b) The group convolution structure of the ResNet18 for the standard residual structure (c) The group convolution structure of the ResNet18 for the dimension-increasing residual structure

Fig. 5. The structure of the lightweight module

4 Experiments

4.1 Setting

In this section, we conducted experiments on the CKDF and the lightweight combined knowledge distillation framework. We trained three different neural networks, namely VGG [31], ResNet [32], and MobileNetV2 [33], on the CIFAR-10, CIFAR-100 and TinyImageNet datasets. In addition, we compared the CKDF with existing distillation methods on the CIFAR-100 dataset.

During the training process, we utilized the SGD optimizer to optimize the neural networks. The weight decay was set to *weight_decay*=5e-4, the momentum to *momentum*=0.9. On both the CIFAR-10 and CIFAR-100 datasets, the initial learning rate for the neural networks was set to 0.1, trained for 200 epochs. And the learning rate was divided by 10 at the 66th, 133rd, and 190th epochs. The Softmax temperature coefficient T had a default value of 3.0. The batch size had a default value of 128. On the TinyImageNet dataset, the initial learning rate of the neural network was set to 0.1, trained for 100 epochs. The learning rate is divided by 10 at the 33rd, 66th, and 90th epochs. T was set to 3.0, and the batch size was 128. All experiments were implemented using PyTorch 1.9.1 on GPU devices.

4.2 Dataset

CIFAR-10:

The CIFAR-10 dataset comprises a total of 60,000 images, divided into 10 classes, with each class containing 6,000 images. Each image is a 32x32 color image with three channels: R, G, and B. Among these images, 5,000 are allocated for training purposes, while the remaining 1,000 images are designated for testing.

CIFAR-100:

The CIFAR-100 dataset comprises a total of 60,000 images, divided into 100 classes, with each class containing 600 images. Each image is a 32x32 color image with three channels: R, G, and B. Among these images, 500 are allocated for training purposes, while the remaining 100 images are designated for testing.

TinyImageNet:

The TinyImageNet dataset contains a total of 120,000 images, distributed across 200 classes. Each class comprises 500 training images, 50 validation images, and 50 test images. Typically, Tiny ImageNet images are resized to smaller dimensions, often 64x64 pixels.

4.3 The Results on CIFAR-10

The student network comprehensively learns the knowledge within the network through shallow classifiers, enabling the full utilization of knowledge. Students comprehensively learn the knowledge within the network through shallow classifiers, enabling full utilization of that knowledge. To validate the effectiveness of knowledge utilization both inside and outside the neural network, we demonstrate the accuracy of various branches within the framework. The data in Table 1 demonstrates that on the CIFAR-10 dataset, as the shallow classifiers become deeper, the utilization of network knowledge is optimized to varying degrees. Taking Resnet18 as an example, the accuracy of Branch4 improved by 1.9% compared to Branch1. The VGG_BN network showed an average improvement of 0.9%.

Table 1. Experiment results of accuracy (%) on CIFAR-10
(Baseline refers to the results obtained from knowledge distillation training. SKD represents the results after self-distillation training. Branch1, Branch2, and Branch3 denote the results after distillation training of the shallow classifiers 1 to 3. Branch4 represents the output of deep classifier.)

Models	Baseline	SKD [29]	Branch1	Branch2	Branch3	Branch4
ResNet18	95.1	95.6	93.8	95.0	95.7	95.7
ResNet34	95.6	95.7	94.3	95.5	96.0	96.0
VGG11_BN	92.2	92.9	92.2	92.9	93.3	93.3
VGG13_BN	94.1	94.3	93.5	94.2	94.4	94.4
VGG19_BN	93.9	94.2	93.5	94.2	94.2	94.2
MobileNetV2	91.0	90.8	90.1	91.6	91.5	91.5

To validate the classification performance of different network structures in complex tasks, we continued to evaluate the performance of the combined knowledge distillation framework using the CIFAR-10 dataset. We present the experimental results in Table 1. From the table, we can observe that the performs well on the CIFAR-10 dataset. Compared to the Baseline, CKDF shows an average improvement of 0.53%. It achieves a 2.9% improvement on the MobileNetV2 lightweight network.

Compared to self-distillation, the proposed approach in this paper achieves an average improvement of 0.27%. It reaches a 0.7% improvement on the MobileNetV2 lightweight network. This validates the applicability of the proposed approach to lightweight networks.

4.4 The Results on CIFAR-100

Table 2. Experiment results of accuracy (%) on CIFAR-100
(Baseline refers to the results obtained from knowledge distillation training. SKD represents the results after self-distillation training. Branch1, Branch2, and Branch3 denote the results after distillation training of the shallow classifiers 1 to 3. Branch4 represents the output of deep classifier.)

Models	Baseline	SKD [29]	Branch1	Branch2	Branch3	Branch4
ResNet18	77.7	78.9	75.5	77.6	79.3	79.7
ResNet34	77.8	79.7	75.5	78.9	80.8	80.8
VGG11_BN	70.0	72.9	72.2	73.2	73.8	73.8
VGG13_BN	74.0	76.2	75.3	75.6	76.3	76.6
VGG19_BN	73.2	74.6	74.3	75.0	75.2	75.1
MobileNetV2	68.9	70.3	68.6	71.2	72.3	71.8

Table 2 displays the results of the distillation experiments on the CIFAR-100 dataset. From the table, we can observe that: Compared to the knowledge distillation, the proposed approach in this paper achieved an average improvement of 2.37%, with a 3.8% improvement on the VGG11_BN network and a 2.9% improvement on the MobileNetV2 lightweight network. Compared to self-distillation, the proposed approach in this paper achieved

an average gain of 0.86%, with a 1.5% improvement on the MobileNetV2 lightweight network. It was confirmed that the proposed method was also applicable to lightweight networks. In the VGG network, Branch1 consistently achieves higher accuracy than the baseline across all networks, and Branch4 consistently outperforms self-distillation. In the combined knowledge distillation framework, as the shallow modules delve deeper, performance gradually improves.

4.5 The Results on TinyImageNet

Table 3 displays the results of the distillation experiments on the TinyImageNet dataset. From the data in the table, we can observe that our proposed framework demonstrates varying degrees of improvement. Both when compared to knowledge distillation and when compared to self-distillation methods. Compared to knowledge distillation and self-distillation methods. In comparison to knowledge distillation, the performance of the framework showed improvements ranging from 3.1% to 5.2%. Likewise, when compared to self-distillation, it achieved an accuracy improvement ranging from 0.9% to 5.7%. These results indicate the effectiveness of the combined knowledge distillation framework on the TinyImageNet dataset.

In addition, the data in the table shows that for ResNet18, the accuracy of Branch4 improved by 4.5% compared to Branch1. And for MobileNetV2, it improved by 5.4%. This further confirms the effectiveness of deepening the shallow classifiers.

Table 3. Experiment results of accuracy (%) on TinyImageNet
(Baseline refers to the results obtained from knowledge distillation training. SKD represents the results after self-distillation training. Branch1, Branch2, and Branch3 denote the results after distillation training of the shallow classifiers 1 to 3. Branch4 represents the output of deep classifier.)

Models	Baseline	SKD [29]	Branch1	Branch2	Branch3	Branch4
ResNet18	63.0	61.3	62.5	64.1	66.0	67.0
ResNet34	59.3	63.1	59.6	63.0	64.2	64.0
VGG11_BN	57.7	59.6	61.5	61.8	61.7	61.0
VGG13_BN	59.9	61.7	64.0	63.8	64.1	63.0
VGG19_BN	59.4	62.5	61.8	64.5	65.0	64.8
MobileNetV2	55.6	57.0	54.1	58.0	60.3	59.5

4.6 Compared to Other Methods

To assess the effectiveness of our proposed method, we compared CKDF with several existing knowledge distillation approaches on the CIFAR-100 dataset. Table 4 presents the comparative results between the CKDF and knowledge distillation methods.

When applying our approach to perform knowledge distillation with a teacher-student pair of ResNet18 and MobileNetV2, our proposed solution exhibited a performance difference of only 0.1 compared to the best-performing DML method. Compared to the latest knowledge distillation solution, DKD, the performance of our framework outperforms it. In other teacher-student scenarios, our approach consistently outperformed alternative distillation methods. The experimental results demonstrate that, in contrast to knowledge distillation approaches, our method can comprehensively extract useful information from the network. This performance improvement is achieved by effectively utilizing both dataset and network knowledge.

Table 4. Comparison of different methods on CIFAR-100

Teacher Student	ResNet18 MobileNetV2	Resnet50 ResNet18	VGG16_BN VGG11_BN
KD [15]	71.3	79.0	72.8
SKD [29]	70.6	79.1	72.3
SCAN [34]	71.4	79.5	71.9
DML [35]	**71.9**	79.0	72.4
DKD [36]	70.8	78.3	68.5
CKDF	71.8	**79.7**	**73.8**

*The highest accuracy is indicated by numbers in bold.

4.7 Lightweight Network

We selected five neural networks and trained them on two different datasets. We applied the combined knowledge distillation to the neural network with group convolution, and the model lightweight was realized. The Table 5 presents the experimental results.

Using CKDF on the CIFAR-10 dataset, the lightweight ResNet18 network reduces parameters by 45.4% compared to the original neural network while improving performance. At the same time, the VGG network achieves an average parameter reduction of 16.4% with a modest 0.3% performance gain. On the CIFAR-100 dataset, improvements of varying degrees were observed compared to the original neural networks. The lightweight ResNet18 network reduces parameters by 44.7% and achieves a performance improvement of 2.2%. Meanwhile, the VGG network averages a 16.1% parameter reduction with a corresponding average performance increase of 1.25%. Experimental data bears out that our proposed lightweight CKDF achieves the same or even higher accuracy with fewer parameters than knowledge distillation. This accomplishment aligns with our original research objectives.

Table 5. Comparison of lightweight network and traditional network

Models	CIFAR-10		CIFAR-100	
	Accuracy/%	Parameters/M	Accuracy/%	Parameters/M
ResNet18	95.1	12.10	77.7	12.28
ResNet18_lightweight	95.3	6.61	79.9	6.79
VGG11_BN	92.2	30.94	70.0	31.45
VGG11_BN_lightweight	92.6	26.81	71.7	27.32
VGG13_BN	94.1	31.12	74	31.63
VGG13_BN_lightweight	94.3	27.00	75.3	27.50
VGG16_BN	93.7	36.44	73.4	36.94
VGG16_BN_lightweight	94.2	29.95	74.9	30.46
VGG19_BN	93.9	41.75	73.2	42.26
VGG19_BN_lightweight	94.0	32.90	73.9	33.41

4.8 Ablation Experiment

In the combined knowledge distillation framework, multiple loss functions are used. We conducted experiments with ResNet18 and VGG11_bn networks on the CIFAR-100 dataset. We employed various combinations of loss functions (loss1+loss2, loss1+loss3, loss2+loss3) to investigate their impact on model performance. The experimental results are shown in Table 6. The data shows that various combinations of loss functions can significantly affect model performance. For Branch4, removing loss1 or loss3 both led to a decrease in model performance. Furthermore, when loss2 was removed, the accuracy decreased by 1.0% and 1.5% for resnet18 and vgg11_bn, respectively. This significant drop in performance demonstrates the crucial role of loss2 in the framework's performance.

Table 6. Accuracy of ablation experiments on CIFAR-100 dataset (%)

Models	Classfiers	loss1+loss2	loss1+loss3	loss2+loss3	Loss
ResNet18	Branch 4	79.6	78.7	79.2	**79.7**
	Branch 3	79.1	78.5	79.1	**79.3**
	Branch 2	**77.8**	77.0	77.9	77.6
	Branch 1	75.4	75.1	75.1	**75.5**
VGG11_BN	Branch 4	73.3	72.3	73.0	**73.8**
	Branch 3	**73.8**	73.2	73.8	**73.8**
	Branch 2	72.4	72.8	73.0	**73.2**
	Branch 1	71.5	71.3	71.8	**72.2**

*The highest accuracy is indicated by numbers in bold.

4.9 Random Erasing on CIFAR-100

To validate the robustness of CKDF, we chose to perform random erasing on CIFAR-100. Random Erasing is performed with a certain probability, as shown in Fig. 6. Therefore, we conducted experiments using probability as a variable. We set the probability to 0.1, 0.2, 0.3, 0.4, and 0.5. We tested our framework and resnet18 on the dataset, and the results are shown in Table 7. After performing Random Erasing, the accuracy of our framework

decreased when compared to its performance on the original CIFAR-100 dataset. However, the accuracy still remained higher than that of resnet18.

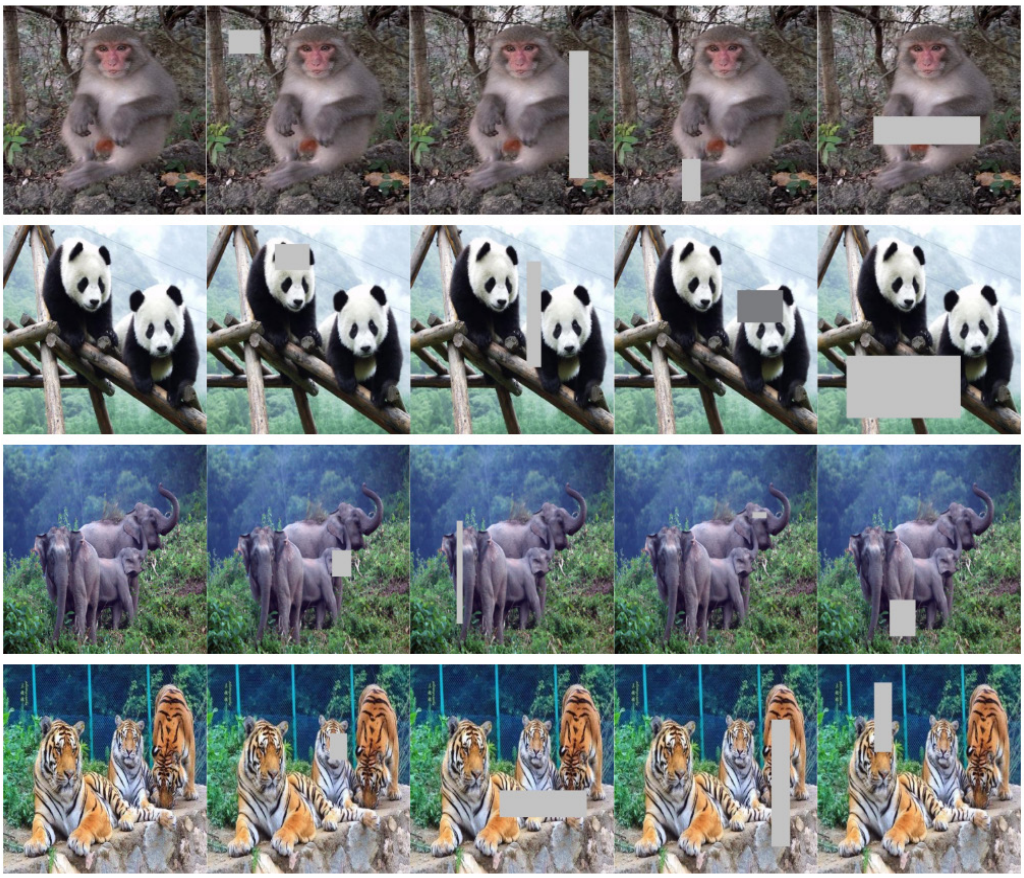


Fig. 6. Example of random erasing on images
(Randomly selecting rectangular areas to erase their pixels, thereby increasing the difficulty of model recognition.)

Table 7. The results after random erasing on CIFAR-100
("Probability" refers to the probability of random erasing. Baseline" represents the results after training resnet18. Branch1, Branch2, and Branch3 denote the results after distillation training of the shallow classifiers 1 to 3. Branch4 represents the output of deep classifier.)

Probability	Baseline	Branch1	Branch2	Branch3	Branch4
0.1	76.2	72.9	75.4	77.7	77.2
0.2	76.0	73.6	75.6	77.5	76.6
0.3	75.5	70.7	73.1	75.2	75.7
0.4	74.5	69.9	72.4	74.3	74.7
0.5	74.1	69.4	71.3	73.4	74.5

5 Conclusions

In response to the trends and demands in deep learning, this paper conducts further research on the model compression technique. We found that knowledge distillation and self-distillation cannot fully utilize knowledge. Therefore, we proposed a novel framework - combined knowledge distillation framework. Performing knowledge distillation and self-distillation simultaneously on neural network models. Comprehensively utilize previously underutilized knowledge. We conducted extensive experiments using different network models on various

datasets. In the experiments, the CKDF demonstrates excellent performance, fully proving the effectiveness of our approach. Subsequently, we applied the combined knowledge distillation to lightweight networks. Compared to knowledge distillation, we achieved similar or better performance with fewer parameters. However, along with the performance improvement, some issues arise. Our model integrates multiple shallow classifiers. It demands additional storage space and computational complexity. Next, we will research how to optimize the shallow classifiers further to reduce their complexity. Additionally, the teacher network conducts offline knowledge distillation for the self-distillation framework. The knowledge given to the student network cannot change. We will also conduct research on this aspect.

6 Acknowledgement

This work is partially supported by Key Research and Development Program in Henan Province (231111210500) and National Key R&D Program of China: Key technical equipment for disaster monitoring, warning, and information acquisition based on communication big data (2023YFC3010700).

The authors also gratefully acknowledge the helpful comments and suggestions of them reviewers, which have improved the presentation.

References

[1] Y. LeCun, Y. Bengio, G. Hinton, Deep learning, Nature 521(7553)(2015) 436-444.
[2] L. Schmarje, M. Santarossa, S.-M. Schröder, R. Koch, A survey on semi-, self-and unsupervised learning for image classification, IEEE Access 9(2021) 82146-82168.
[3] Z. Zhong, X. Sun, Z. Wu, Y. Zheng, S. Lin, L. Sato, Animation from blur: Multi-modal blur decomposition with motion guidance, in: Proc. 2022 European Conference on Computer Vision, Springer Nature Switzerland, Cham, 2022.
[4] L. Jiao, F. Zhang, F. Liu, S. Yang, L. Li, Z. Feng, R. Qu, A survey of deep learning-based object detection, IEEE Access 7(2019) 128837-128868.
[5] Y. Li, C.-Y. Wu, H. Fan, K. Mangalam, B. Xiong, J. Malik, C. Feichtenhofer, Mvitv2: Improved multiscale vision transformers for classification and detection, in: Proc. 2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2022.
[6] M. Bhan, N. Achache, V. Legrand, A. Blangero, N. Chesneau, Evaluating self-attention interpretability through human-grounded experimental protocol, in: Proc. 2023 World Conference on Explainable Artificial Intelligence, Springer Nature Switzerland, Cham, 2023.
[7] S. Chen, Y. Zhang, Q. Yang, Multi-task learning in natural language processing: An overview, ACM Computing Surveys 56(12)(2024) 1-32.
[8] M. Tamador, K. Khalil, A survey on neural network hardware accelerators, IEEE Transactions on Artificial Intelligence 4(2)(2024) 123-140.
[9] S. Mittal, A survey of FPGA-based accelerators for convolutional neural networks, Neural Computing and Applications 32(4)(2020) 1109-1139.
[10] C. Bucilua, R. Caruana, A. Niculescu-Mizil, Model compression, in: Proc. 2006 the 12th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, 2006.
[11] Z. Lyu, T. Yu, F. Pan, Y. Zhang, J. Luo, D. Zhang, Y. Chen, B. Zhang, G. Li, A survey of model compression strategies for object detection, Multimedia tools and applications, 83(16)(2024) 48165-48236.
[12] S. Vadera, S. Ameen, Methods for pruning deep neural networks, IEEE Access 10(2022) 63280-63300.
[13] A. Polino, R. Pascanu, D. Alistarh, Model compression via distillation and quantization, IEEE Transactions on Neural Networks and Learning Systems 30(6)(2019) 1876-1887.
[14] Z. Li, P. Xu, X. Chang, L. Yang, Y. Zhang, L. Yao, X. Chen, When object detection meets knowledge distillation: A survey, IEEE Transactions on Pattern Analysis and Machine Intelligence 45(8)(2023) 10555-10579.
[15] G. Hinton, O. Vinyals, J. Dean, Distilling the knowledge in a neural network, <https://arxiv.org/abs/1503.02531>, 2015 (accessed 17.11.2022).
[16] S. Yun, J. Park, K. Lee, J. Shin, Regularizing class-wise predictions via self-knowledge distillation, in: Proc. 2020 the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2020.
[17] X. Nie, Y. Li, L. Luo, N. Zhang, J. Feng, Dynamic kernel distillation for efficient pose estimation in videos, in: Proc. 2019 IEEE/CVF International Conference on Computer Vision (ICCV), 2019.
[18] Z. Yang, A. Zeng, Z. Li, T. Zhang, C. Yuan, Y. Li, From Knowledge Distillation to Self-Knowledge Distillation: A Unified Approach with Normalized Loss and Customized Soft Labels, in: Proc. 2023 International Conference on Computer Vision 2023.

[19] S. Ni, X. Ma, M. Zhu, J. Liu, L. Zhang, Reverse Self-distillation Overcoming the Self-distillation Barrier, IEEE Open Journal of the Computer Society 1(2023) 26-38.
[20] M. Gao, Y. Shen, Q. Li, J. Yan, L. Wan, D. Lin, C.-C. Loy, X. Tang, An embarrassingly simple approach for knowledge distillation, <https://arxiv.org/abs/1812.01819>, 2018 (accessed 23.11.2022).
[21] S.-I. Mirzadeh, M. Farajtabar, A. Li, N. Levine, A. Matsukawa, H. Ghasemzadeh, Improved knowledge distillation via teacher assistant, in: Proc. 2020 the AAAI Conference on Artificial Intelligence, 2020.
[22] G. Wu, S. Gong, Peer collaborative learning for online knowledge distillation, in: Proc. 2021 AAAI Conference on Artificial Intelligence, 2021.
[23] H. Lee, S.-J. Hwang, J. Shin, Self-supervised label augmentation via input transformations, in: Proc. 2020 37th International Conference on Machine Learning, Virtual Event, 2020.
[24] P. Chen, S. Liu, H. Zhao, J. Jia, Distilling knowledge via knowledge review, in: Proc. 2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2021.
[25] S. You, C. Xu, D. Tao, Learning from multiple teacher networks, in: Proc. 2017 the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, 2017.
[26] Y. Liu, W. Zhang, J. Wang, Adaptive multi-teacher multi-level knowledge distillation, Neurocomputing 415(2020) 106-113.
[27] C. Pham, T. Hoang, T.-T. Do, Collaborative Multi-Teacher Knowledge Distillation for Learning Low Bit-width Deep Neural Networks, in: Proc. 2023 the IEEE/CVF Winter Conference on Applications of Computer Vision, 2023.
[28] L. Zhang, J. Song, A. Gao, J. Chen, C Bao, K. Ma, Be your own teacher: Improve the performance of convolutional neural networks via self distillation, in: Proc. 2019 the IEEE/CVF International Conference on Computer Vision, 2019.
[29] L. Zhang, C. Bao, K. Ma, Self-distillation: Towards efficient and compact neural networks, IEEE Transactions on Pattern Analysis and Machine Intelligence 44(8)(2022) 4388-4403.
[30] S. Zagoruyko, N. Komodakis, Paying more attention to attention: Improving the performance of convolutional neural networks via attention transfer, <https://arxiv.org/abs/1612.03928>, 2016 (accessed 21.04.2023).
[31] K. Simonyan, A. Zisserman, Very deep convolutional networks for large-scale image recognition, in: Proc. 2015 International Conference on Learning Representations (ICLR), 2015.
[32] K. He, X. Zhang, S. Ren, J. Sun, Deep residual learning for image recognition, in: Proc. 2016 the IEEE Conference on Computer Vision and Pattern Recognition, 2016.
[33] M. Sandler, A. Howard, M. Zhu, A. Zhmoginov, L.-C. Chen, Mobilenetv2: Inverted residuals and linear bottlenecks, in: Proc. 2018 the IEEE Conference on Computer Vision and Pattern Recognition, 2018.
[34] L. Zhang, Z. Tan, J. Song, J. Chen, C. Bao, K. Ma, Scan: A scalable neural networks framework towards compact and efficient models, in: Proc. 2019 Advances in Neural Information Processing Systems, 2019.
[35] Y. Zhang, T. Xiang, T.-M. Hospedales, H. Lu, Deep mutual learning, in: Proc. 2018 the IEEE Conference on Computer Vision and Pattern Recognition, 2018.
[36] B. Zhao, Q. Cui, R. Song, Y. Qiu, J. Liang, Decoupled knowledge distillation, in: Proc. 2022 the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2022.