Feature fusion-based collaborative learning for knowledge distillation

Yiting Li\textsuperscript{1}, Liyuan Sun\textsuperscript{1}, Jianping Gou\textsuperscript{1,}\textsuperscript{a}, Lan Du\textsuperscript{2} and Weihua Ou\textsuperscript{3}

Abstract
Deep neural networks have achieved a great success in a variety of applications, such as self-driving cars and intelligent robotics. Meanwhile, knowledge distillation has received increasing attention as an effective model compression technique for training very efficient deep models. The performance of the student network obtained through knowledge distillation heavily depends on whether the transfer of the teacher’s knowledge can effectively guide the student training. However, most existing knowledge distillation schemes require a large teacher network pre-trained on large-scale data sets, which can increase the difficulty of knowledge distillation in different applications. In this article, we propose a feature fusion-based collaborative learning for knowledge distillation. Specifically, during knowledge distillation, it enables networks to learn from each other using the feature/response-based knowledge in different network layers. We concatenate the features learned by the teacher and the student networks to obtain a more representative feature map for knowledge transfer. In addition, we also introduce a network regularization method to further improve the model performance by providing a positive knowledge during training. Experiments and ablation studies on two widely used data sets demonstrate that the proposed method, feature fusion-based collaborative learning, significantly outperforms recent state-of-the-art knowledge distillation methods.

Keywords
Model compression, knowledge distillation, collaborative learning, feature fusion, deep learning

Date received: 28 August 2021; accepted: 8 October 2021

Handling Editor: Yanjiao Chen

Introduction
Recently, since deep neural networks (DNNs) have shown breakthrough results in the visual recognition tasks, the number of deep learning applications in real-world scenarios has exploded\textsuperscript{1,3} - These deep learning-based methods have been widely used in self-driving cars, cancer detection, and intelligent robotics. However, the high performance of DNNs mainly comes at the cost of the high computational complexity. Therefore, it is usually very difficult to deploy large-scale DNNs on mobile and embedded devices due to their limited computational power. To overcome those issues, several model compression methods have been developed to improve the model efficiency without significantly sacrificing the model accuracy, such as

\textsuperscript{1}School of Computer Science and Communication Engineering, Jiangsu University, Zhenjiang, China
\textsuperscript{2}Department of Data Science and Artificial Intelligence, Faculty of Information Technology, Monash University, Clayton, VIC, Australia
\textsuperscript{3}School of Big Data and Computer Science, Guizhou Normal University, Guiyang, China

Corresponding authors:
Jianping Gou, School of Computer Science and Communication Engineering, Jiangsu University, 301 Xuefu Road, Zhenjiang 212013, China.
Email: goujianping@ujs.edu.cn

Liyuan Sun, School of Computer Science and Communication Engineering, Jiangsu University, 301 Xuefu Road, Zhenjiang 212013, China.
Email: 2211908031@stmail.ujs.edu.cn

Creative Commons CC BY: This article is distributed under the terms of the Creative Commons Attribution 4.0 License (https://creativecommons.org/licenses/by/4.0/) which permits any use, reproduction and distribution of the work without further permission provided the original work is attributed as specified on the SAGE and Open Access pages (https://us.sagepub.com/en-us/nam/open-access-at-sage).
network pruning,\textsuperscript{4,5} low-rank decomposition,\textsuperscript{6,7} and knowledge distillation (KD).\textsuperscript{8,9} Among different model compression schemes, KD has received a lot of attention because of its great flexibility in teacher–student network architectures. Specifically, KD is first formally introduced by Hinton et al.,\textsuperscript{8} where the teacher network transfers the knowledge in the output layer to the student network. Furthermore, Romero et al.\textsuperscript{9} develop the idea of FitNets, that is, the middle layer of DNNs is introduced by Hinton et al.,\textsuperscript{8} where the teacher network, and thus avoids the problem caused by the large-scale pre-trained teacher network.\textsuperscript{13–17} Specifically, Zhang et al.\textsuperscript{13} proposed a distillation method without indicating the teacher network, that is, two peer student networks learn from each other. Guo et al.\textsuperscript{14} utilizes the collaborative learning to ensemble the output of all student networks to improve the performance of student networks. However, the above two methods only consider the knowledge of the output layer of the student network, making it possible for further improvement using feature knowledge. For example, Hou et al.\textsuperscript{15} fuse the features of the middle layer in two parallel student networks using the fusion module formed by a simple “SUM” operation, while two parallel student networks must share the same network structure. Kim et al.\textsuperscript{16} proposed a feature fusion learning method to fuse the features of the student network and devise an ensemble classifier to work together to improve the model performance.

To further improve the performance of the student network with a more effective KD scheme, we propose a feature fusion-based collaborative learning (FFCL) for KD in this article. Specifically, in the process of distillation, two parallel peer (or student) networks improve their performance in a collaborative manner. Since the parameters in the network are randomly initialized and different student networks may have different abilities to learn knowledge, there will be a gap between the performance of two parallel student networks with the same architecture during training. In this case, two networks learn from each other, while the network with a poor performance may affect the network with a good performance, which then affects the final results. Therefore, before the distillation process, we first pre-train the network that will participate in the training process. During the distillation process, the pre-trained network will guide the corresponding network. We refer to this step as the network regularization, which can enable the student network to obtain the correct knowledge from the pre-trained network, reduce the negative impact of the wrong knowledge among the peer networks, and further avoid the issue caused by training a large-scale teacher network. Moreover, to further utilize more feature knowledge from the middle layers during collaborative learning, we fuse the features from peer networks to obtain more representative features, which can then be used to further improve network performance. Consequently, through the collaborative learning between peer networks, network regularization for each network and feature fusion between peer networks, the distilled knowledge could be more informative for training each peer network. The main contributions of this article can be summarized as follows:

- A novel collaborative learning framework for KD: not only can it improve the performance of parallel student networks, but also improve the performance of the fusion module in an end-to-end trainable manner.
- The network architecture of parallel student networks can be different, and in order to reduce the impact of incorrect knowledge between networks on performance, a regularization process is introduced.

### Related work

**KD**

Due to the excellent performance of DNNs in computer vision, speech recognition, and natural language processing, a variety of KD schemes have been proposed to train small networks with high performances. Existing KD schemes can usually be divided into three types,\textsuperscript{18} (1) offline distillation,\textsuperscript{8–12} (2) online distillation,\textsuperscript{13–17} and (3) self-distillation.\textsuperscript{19–23} As a classic KD scheme, offline distillation can effectively improve the performance of the student network. However, it takes a lot of time to pre-train a large-scale teacher network. Therefore, how to choose a proper teacher network is also a difficult problem. Compared with the offline KD, online distillation does not require a pre-trained teacher network. All peer networks in online distillation are trained from scratch by transferring the knowledge from each other, but a poor-performance network may affect the performance of other networks during collaborative learning. Self-distillation indicates that the network improves its performance in a self-learning way, where one network is trained again using its pre-trained network as the regularization during learning. For instance, Yuan et al.\textsuperscript{19} proposed a teacher-free knowledge distillation (Tf-KD) method, which is a...
special self-learning framework. Therefore, an effective distillation method should not only improve the performance of the student network but also save training time and storage space. To overcome the weakness existed in each distillation strategy, we explore a new distillation scheme, where multiple different schemes can work together to further improve the performance of the model.

**Collaborative learning**

Recently, many KD schemes based on collaborative learning have been proposed. Specifically, Zhang et al. proposed a deep mutual learning (DML) strategy, where a group of student networks learn from each other and guide each other throughout the training process, instead of using a pre-defined one-way conversion path between teacher and student networks. Guo et al. proposed knowledge distillation via collaborative learning (KDCL), which ensembles the output of different networks and then use the ensemble results to teach each individual network through collaborative learning. Lan et al. constructed a multi-branch structure through the network hierarchy. The authors regard each branch as a student network, and merge these branches to generate a better-performing teacher network. Then, through the joint online learning of the teacher–student network, a single-branch model or a multi-branch fusion model with superior performance can be obtained. Chen et al. proposed a two-level scheme for the online KD with a group leader and multiple auxiliary peers. Hou et al. extracted and fused the features of two student networks to obtain more meaningful feature maps, and then input the fused features into the fusion module. Although the final classifier can achieve better performance, two student networks must share the same network structure, because the authors adopt a simple “SUM” operation in the process of feature fusion. Kim et al. made a further improvement on this basis of using feature fusion learning (FFL) to improve the performance of the fusion classifier. This scheme is also suitable for two student networks with different network structures. Different from the above methods, our proposed FFCL scheme learns different knowledge among the peer networks in the distillation process of collaborative learning, and uses a network regularization for each peer model to improve the performance.

**The proposed method**

In this section, we introduce our FFCL framework in detail. We first describe how to perform collaborative learning between peer student networks, and then introduce how to use regularization to improve the network performance during the learning process.

During the collaborative learning, different networks learn from each other using the feature knowledge in the middle layer and the response knowledge in the output layer. We extract features from the middle layers of two networks, and fuse these features to get more meaningful feature maps. We then input the feature maps into the fusion module, and the output from the fused classifier will be used to guide each peer network to improve the performance. In this way, the network can learn the feature knowledge in the middle layer of other networks. At the same time, the networks also learn the response knowledge in the output layer from each other. All networks are trained from scratch during the distillation process of collaborative learning. Furthermore, to make each peer network to be learned with more positive knowledge from its other networks, a model regularization process is introduced during collaborative learning. Before the training process of the peer student networks, the model regularization is first carried out by pre-training all peer networks that can participate in the training process. During the training process, the knowledge is extracted from the pre-trained network and transferred to the corresponding network. The overview framework of FFCL is illustrated in Figure 1.

**Notations**

Given $n$ samples $X = \{x_1, x_2, \ldots, x_n\}$ from $m$ classes, we denote the corresponding label set as $Y = \{y_1, y_2, \ldots, y_n\}$. Among them, $y_i \in \{1, 2, \ldots, m\}$. For a peer network $\theta_k$, the feature in any one middle layer is denoted as $f_k$, and the logit output from network $\theta_k$ is denoted as $z_k$. The fused module is denoted as $\theta_k$, which is a small network. We denote the logit output of the fused module as $z_f$. In addition, we denote the network $\theta_k$ which has completed the pre-training process as $\theta_k^0$.

**Fused feature-based collaborative distillation**

In the process of collaborative learning, the network learns two parts of knowledge from each other, namely, fused feature knowledge from the middle layers of the peer networks and response knowledge of their output layers. First, we introduce the feature knowledge during collaborative distillation. The features extracted from the middle layers of the given peer student network $\theta_k$ and $\theta_{k'}$ are $f_k$ and $f_{k'}$, respectively. We concatenate $f_k$ and $f_{k'}$ along the channel dimension to get the fused feature $f$. And then, the feature $f$ is inputted into the fused module, which has the logit output $z_f$. The distillation loss function of knowledge transfer for the peer student
network \( \theta _k \) with the fused feature knowledge from the networks \( \theta _k \) and \( \theta _k' \) is defined as

\[
L^f _\theta = \sum _{x \in X} \sum _{i=1}^m \sigma _i (z_f ; T) \log \left( \frac{\sigma _i (z_f ; T)}{\sigma _i (z_{k'} ; T)} \right)
\]

where \( T \) indicates the temperature parameter and \( \sigma \) stands for the softmax function. Here, we use the last feature map of the peer student networks for feature fusion. Similarly, the distillation loss function for the peer student network \( \theta _k' \) with the fused feature knowledge is defined as

\[
L^f _{\theta _k'} = \sum _{x \in X} \sum _{i=1}^m \sigma _i (z_f ; T) \log \left( \frac{\sigma _i (z_f ; T)}{\sigma _i (z_{k'} ; T)} \right)
\]

In the distillation process with fusing features, the fused model is always very small network and its used structure is chosen as in Figure 2. In the proposed FFCL, the used network structure of fused module \( \theta _f \) is composed of a 3x3 depthwise convolution and a pointwise convolution layer. For example, for the ResNet18 network, we set the batch size to 128, and then the dimension of the final feature map is [128, 512, 4, 4]. In the case of two peer networks with ResNet18, we concatenate two features along the channel dimension, and the feature dimension output by the fused module is [128, 1024, 4, 4].

Next, we introduce the response knowledge during collaborative distillation. That is, one peer student network learns the knowledge from the output layer of the other peer student network during the training process.

The distillation loss function of transferring response knowledge from the network \( \theta _k' \) to \( \theta _k \) is defined as

\[
L^{r} _\theta = \sum _{x \in X} \sum _{i=1}^m \sigma _i (z_{k'} ; 1) \log \left( \frac{\sigma _i (z_{k'} ; 1)}{\sigma _i (z_k ; 1)} \right)
\]
where the value 1 means the temperature parameter \( T = 1 \), and in fact, peer network \( \theta_k \) is the student and \( \theta_{k'} \) is the teacher. In a similar way, the distillation loss from the network \( \theta_k \) to \( \theta_{k'} \) is defined as

\[
L^s_{\theta_k} = \sum_{x \in X} \sum_{i = 1}^m \sigma_i(z_{ki}; T) \log \left( \frac{\sigma_i(z_{ki}; T)}{\sigma_i(z_{ki}; T)} \right)
\]

Similarly, the distillation loss function as a network regularization for the peer network \( \theta_{k'} \) is defined as

\[
L^s_{\theta_{k'}} = \sum_{x \in X} \sum_{i = 1}^m \sigma_i(z_{ki'}; T) \log \left( \frac{\sigma_i(z_{ki'}; T)}{\sigma_i(z_{ki'}; T)} \right)
\]

In the ablation experiment, it is obviously that Case G, which only represents regularization-based distillation, is the most effective method among all the cases (Cases E–G). From this, we affirm the validity of network regularization and ensure the effectiveness of those pre-trained networks.

**FFCL loss**

In the distillation process, each step has its own favorable effect, and they work together to improve the network performance. In the framework of collaborative learning between two peer networks \( \theta_k \) and \( \theta_{k'} \), the overall distillation loss function for simultaneously training the peer student network \( \theta_k \) and the fused module \( \theta_f \) is formulated as follows

\[
L^D_{\theta_k} = L^{col}_{\theta_k} + \beta T^2 L^f_{\theta_k} + \gamma_1 L^{CE}_{\theta_k} + \gamma_2 L^{CE}_{\theta_f}
\]

where \( \beta, \gamma_1, \) and \( \gamma_2 \) are the hyper-parameters. \( L^{CE} \) indicates the cross-entropy function between the network logits and the ground true labels, that is

\[
L^{CE}_{\theta_k} = -\sum_{x \in X} \sum_{i = 1}^m y_i \log(\sigma_i(z_{ki}; 1))
\]

\[
L^{CE}_{\theta_f} = -\sum_{x \in X} \sum_{i = 1}^m y_i \log(\sigma_i(z_{ki}; 1))
\]

Similarly, for training network \( \theta_{k'} \) with the fused module \( \theta_f \), the overall distillation loss function is defined as

\[
L^D_{\theta_{k'}} = L^{col}_{\theta_{k'}} + \beta T^2 L^f_{\theta_{k'}} + \gamma_1 L^{CE}_{\theta_{k'}} + \gamma_2 L^{CE}_{\theta_f}
\]

where \( L^{CE}_{\theta_{k'}} \) is defined as

\[
L^{CE}_{\theta_{k'}} = -\sum_{x \in X} \sum_{i = 1}^m y_i \log(\sigma_i(z_{ki}; 1))
\]

In summary, using the overall distillation loss function above, the collaborative learning of the two peer networks is performed by KD, and the proposed FFCL is shown in Algorithm 1. As a result, the proposed collaborative KD between two peer student networks can further improve the performance, which is verified in the experimental section.

**FFCL extension**

The formulation of the proposed FFCL above is based on two peer networks and can be a standard FFCL framework. In fact, it can be extended more than two peer networks. Given \( N (N > 2) \) peer networks denoted as the network set \( D = \{ \theta_1, \theta_2, \ldots, \theta_n \} \), the feature in middle layer of an arbitrary peer network \( \theta_p \) is denoted as \( f_p \) and the logit output of the network \( \theta_p \) is represented as \( z_p \). The fused module for all the peer networks is still set as \( \theta_f \) and its logit output is indicated as \( z_f \). In
Algorithm 1. The proposed FFCL.

Require: input samples X with labels Y, epoch t, hyper-parameters α, β, γ, and γ.
1: Initialize: initialize peer student networks θ̂ and θ1, and the fused module θf under different conditions.
2: Stage 1: pre-train network θ̂ and network θ1 for use of the process of network regularization.
3: t = 0.
4: Repeat:
5: for i = 1 to 2 do
6: Compute stochastic gradient of cross-entropy loss and update
7: θ̂ ← θ̂ + η \frac{∂L_{\text{CE}}}{∂θ̂}.
8: t = t + 1.
9: end for
10: Until: t = T.
11: Stage 2: train peer student networks θ1 and θ2 collaboratively.
12: t = 0.
13: for k = 1 to 2 do
14: Repeat:
15: Compute stochastic gradient of L^D_{\theta_k} in equation (9) and update θ_k and θ_f;
16: θ̂ ← θ̂ + η \frac{∂L_{\text{CE}}}{∂θ̂}, and θ_f ← θ_f + η \frac{∂L_{\text{CE}}}{∂θ_f}.
17: t = t + 1.
18: end for
19: Until: t = T.
20: Return θ̂ and θ_f.

During the collaborative learning among the multiple peer student networks, each network learns from the other N−1 peer networks via knowledge transfer. That is, each peer student network is trained by distilling the knowledge from the other peer networks. The distillation loss function of knowledge transfer for the network θ_u with the fused feature knowledge is defined as

\[ L^D_{\theta_u} = \sum_{x \in X} \sum_{i=1}^m \sigma_i(z_i; T) \log \left( \frac{\sigma_i(z_i; T)}{\sigma_i(z_i; T)} \right) \]  \hspace{1cm} (14)

where \( z_i \) is the logit output from network \( \theta_i \).

Finally, the overall distillation loss function of the general FFCL for learning the network \( \theta_u \) is formulated as follows

\[ L^D_{\theta_u} = \frac{1}{N-1} \sum_{a=1,a\neq u}^N \sum_{i=1}^m \sigma_i(z_i; 1) \log \left( \frac{\sigma_i(z_i; 1)}{\sigma_i(z_i; 1)} \right) \]  \hspace{1cm} (15)

where \( z_i \) is the logit output from network \( \theta_i \).

Experiments

We conducted favorable experiments to verify the effectiveness of FFCL on CIFAR-10 and CIFAR-100 data set,\textsuperscript{27} while comparing our FFCL framework with some

Figures 3 and 4. In the figures, the green arrow represents the feature after combining the features of peer networks, and \( z_f \) represents the features output by the fused module \( \theta_f \).

\[ L^D_{\theta_u} = \alpha T^2 L^D_{\theta_u} + (1-\alpha)T^2 L^D_{\theta_u} + \beta T^2 L^D_{\theta_u} + \gamma_1 L^CE_{\theta_u} + \gamma_2 L^CE_f \]  \hspace{1cm} (16)

where \( L^D_{\theta_u} \) is the network regularization function computed as in equation (7), and \( L^CE_{\theta_u} \) and \( L^CE_f \) are the cross-entropy functions, respectively, computed as equations (10) and (11).

To intuitively understand the proposed FFCL, we provide the overview diagrams of the standard FFCL between two peer student networks and the general FFCL among three peer student networks, shown in Figures 3 and 4. In the figures, the green arrow represents the process of network regularization, and the yellow arrow represents the process of feature fusion.
classic and recent state-of-the-art methods including KD, DML, KDCL, FFL, and Tf-KD. The architecture of each peer network was chosen from ResNet, WideResNet (WRN), and ShuffleNet. The baselines were trained for 200 epochs with batch size 128. The initial learning rate is 0.1 and then divided at the 60th, 120th, and 160th epochs. We used SGD optimizer with momentum of 0.9, and weight decay was set as $5e^{-4}$. The average top-1 accuracy (%) of baselines for different networks is reported in Table 1.

**Results on CIFAR-10**

We first compare our proposed FFCL based on two peer networks on CIFAR-10 data set with DML, KDCL, FFL, and Tf-KD. We considered five pairs of peer networks, selected from ResNet and ShuffleNet. The top-1 accuracies over three individual runs with the corresponding standard deviations derived by each model with different architecture settings are reported in Table 2.

Equipped with the feature fusion mechanism during collaborative learning, our FFCL achieved the highest top-1 accuracies across the five architecture settings, gaining improvement on the runner up by 0.24%–0.8%. Comparing the two peer networks (i.e. $\theta_1$ and $\theta_2$) separately, one can also observe that FFCL, FFCL, and FFCL outperformed their counterparts with DML, FFL, and KDCL. Consequently, FFCL performed consistently better than all the other competitors considered on the CIFAR-10 data set. The favorable performance of FFCL can be attributed to the following facts: (1) the fused features contain more comprehensive and informative knowledge, (2) the fused features are more expressive than the features from the middle layer of a single network, and (3) the network regularization process also promotes the improvement of model performance.

**Results on CIFAR-100**

We further compared FFCL based on two peer networks using CIFAR-100 with KD, DML, KDCL, FFL, and Tf-KD. Similarly, we considered six pairs of peer networks that were selected from ResNet, WRN, and ShuffleNet.

Table 3 shows the performance of all the models over the six architecture settings. Similar to what we observed on CIFAR-10, FFCL outperformed all the other state-of-the-art methods with a notable margin, which further demonstrates the effectiveness of FFCL in fusing features. Specifically, with network $\theta_1$ in the peer networks of ResNet18-ResNet18, our FFCL method gained an improvement on KD, Tf-KD, DML, FFL, and KDCL by 0.54%, 0.73%, 1.4%, 2.22%, and 1.04%, respectively. And with network $\theta_2$, FFCL

---

**Data sets and settings**

CIFAR-10 contains a total of 60,000 samples, including 50,000 training samples and 10,000 testing samples. Those samples are divided into 10 classes. CIFAR-100 is similar to CIFAR-10, but contains 100 classes. And each class has the same numbers of training and testing samples.

In the experiments, we used stochastic gradient descent (SGD) optimizer with momentum 0.9 and weight decay $5e^{-4}$. For the hyper-parameters $\alpha_1$, $\alpha_2$, $\beta$, $\gamma_1$, $\gamma_2$ and $T$, they were determined by grid-search as follows: $\alpha_1 = 0.46$, $\alpha_2 = 0.15$, $\beta = 0.57$, $\gamma_1 = 0.18$ and $\gamma_2 = 0.05$, and $T = 6$.

All the reported accuracies were averaged over three runs that are randomly initialized. It is noteworthy that the compared Tf-KD method was trained with the network architecture $\theta_1$ when the collaborative learning was performed under the situations of two peer networks with different architectures. For KD, $N_2$ and $N_1$ were respectively used as the teacher and student networks during the distillation process when they have different architectures, and the proper teacher network was used for the student network $N_1$ when two peer networks $N_1$ and $N_2$ have the same architectures. For three peer networks $\theta_1$, $\theta_2$, and $\theta_3$, the proposed FFCL was respectively indicated as FFCL, FFCL, FFCL, and FFCL. DML, FFL, and KDCL with collaborative learning were indicated in the same way. And both FFCL and FFL using fused models were indicated as FFCL and FFL, respectively.

**Baselines**

To highlight the improvements of the competitors, we provide the baselines of the used networks without KD. On CIFAR-10 and CIFAR-100, we used baseline models including ShuffleNet, ResNet18 and ResNet34. Besides, WRN-28-10 was also used for CIFAR-100. The baselines were trained for 200 epochs with batch size 128. The initial learning rate is 0.1 and then divided at the 60th, 120th, and 160th epochs. We used SGD optimizer with momentum of 0.9, and weight decay was set as $5e^{-4}$. The average top-1 accuracy (%) of baselines for different networks is reported in Table 1.

| Data sets | ShuffleNet | ResNet18 | ResNet34 | WRN-28-10 |
|-----------|------------|----------|----------|-----------|
| CIFAR-10  | 91.03      | 95.23    | 95.39    | –         |
| CIFAR-100 | 70.55      | 76.19    | 77.91    | 79.86     |

WRN: WideResNet.
method beat those competitors by 0.41%, 0.6%, 1.48%, 2%, and 0.75%, respectively. Compared with FFL_θf that also use a feature fusion mechanism, the performance of FFCL_θf won by 1.81%. This set of experimental results demonstrates that the performance of the peer networks can be improved through the network regularization process, and then the peer networks can output features with stronger expression, which can be input into the fused classifier to obtain better performance.

Further experiments

To further investigate the classification performance of the proposed FFCL, we conducted the comparative experiments on CIFAR-100 among the collaborative learning methods using multiple peer networks (i.e. more than two peer networks) including DML, KDCL, FFL, and our FFCL. For easy implementation in the experiments, the collaborative learning was carried out on three different peer architectures composed of three peer networks.

The top-1 accuracies over three individual runs with the corresponding standard deviations derived by each peer model with different architecture settings are reported in Table 4. It can be seen that FFCL_θf significantly outperforms its counterparts with FFL and DML, and our FFCL_θf performs very better than FFL_θf. Moreover, through the experimental results in Tables 3 and 4, our FFCL_θf with three peer networks

| Table 2. The top-1 accuracy (%) over three individual runs with the corresponding standard deviations on CIFAR-10 data set. |
| Network1 (θ1) | ResNet18 | ResNet18 | ResNet34 | ShuffleNet | ShuffleNet |
| Network2 (θ2) | ResNet18 | ResNet34 | ResNet34 | ShuffleNet | ShuffleNet |
|---|---|---|---|---|---|
| TF-KD | 95.26 ± 0.10 | 95.26 ± 0.10 | 95.59 ± 0.08 | 92.29 ± 0.21 | 92.29 ± 0.21 |
| KD | 95.35 ± 0.08 | 95.32 ± 0.04 | 95.49 ± 0.16 | 91.79 ± 0.08 | 91.79 ± 0.08 |
| DML_θ1 | 95.06 ± 0.05 | 95.10 ± 0.06 | 95.45 ± 0.04 | 92.01 ± 0.12 | 91.68 ± 0.16 |
| DML_θ2 | 95.07 ± 0.10 | 95.43 ± 0.21 | 95.48 ± 0.19 | 91.92 ± 0.13 | 91.57 ± 0.11 |
| FFL_θ1 | 94.93 ± 0.11 | 94.80 ± 0.12 | 95.15 ± 0.23 | 90.67 ± 0.07 | 97.26 ± 0.15 |
| FFL_θ2 | 94.86 ± 0.12 | 95.33 ± 0.20 | 95.27 ± 0.03 | 90.96 ± 0.18 | 95.05 ± 0.11 |
| KDCL_θ1 | 95.15 ± 0.08 | 95.03 ± 0.10 | 95.25 ± 0.17 | 92.06 ± 0.12 | 91.76 ± 0.13 |
| KDCL_θ2 | 95.11 ± 0.07 | 95.34 ± 0.18 | 95.19 ± 0.02 | 92.12 ± 0.06 | 95.02 ± 0.18 |
| FFCL_θf (ours) | 95.43 ± 0.11 | 95.36 ± 0.18 | 95.64 ± 0.09 | 92.80 ± 0.17 | 92.94 ± 0.10 |
| FFCL_θf2 (ours) | 95.30 ± 0.15 | 95.75 ± 0.14 | 95.66 ± 0.08 | 92.68 ± 0.08 | 95.48 ± 0.15 |
| FFCL_θf (ours) | 95.72 ± 0.13 | 96.01 ± 0.18 | 96.05 ± 0.04 | 93.30 ± 0.15 | 95.50 ± 0.16 |

KD: knowledge distillation; DML: deep mutual learning; FFL: feature fusion learning; KDCL: knowledge distillation via collaborative learning.

In each column, the highest score is in boldface, and the score with underline is the highest among the competing methods except ours.

| Table 3. The average top-1 accuracy (%) over three individual runs with the corresponding standard deviations on CIFAR-100 data set. |
| Network1 (θ1) | ResNet18 | ResNet18 | ResNet34 | ShuffleNet | ShuffleNet |
| Network2 (θ2) | ResNet18 | ResNet34 | ResNet34 | ShuffleNet | ShuffleNet |
|---|---|---|---|---|---|
| KD | 77.11 ± 0.12 | 77.45 ± 0.11 | 78.17 ± 0.14 | 72.11 ± 0.45 | 72.11 ± 0.45 | 80.12 ± 0.22 |
| TF-KD | 76.92 ± 0.03 | 76.92 ± 0.03 | 77.77 ± 0.28 | 72.24 ± 0.06 | 72.24 ± 0.06 | 80.33 ± 0.24 |
| DML_θ1 | 76.25 ± 0.16 | 76.22 ± 0.23 | 77.51 ± 0.08 | 70.54 ± 0.20 | 70.67 ± 0.36 | 79.93 ± 0.08 |
| DML_θ2 | 76.04 ± 0.24 | 77.27 ± 0.30 | 77.62 ± 0.05 | 70.28 ± 0.19 | 76.32 ± 0.22 | 80.25 ± 0.07 |
| FFL_θ1 | 75.43 ± 0.29 | 75.51 ± 0.13 | 76.54 ± 0.31 | 69.19 ± 0.20 | 70.18 ± 0.17 | 79.23 ± 0.19 |
| FFL_θ2 | 75.52 ± 0.33 | 77.11 ± 0.13 | 77.00 ± 0.33 | 69.70 ± 0.33 | 75.93 ± 0.32 | 79.08 ± 0.18 |
| FFL_θf | 77.53 ± 0.14 | 78.37 ± 0.12 | 78.83 ± 0.18 | 75.54 ± 0.33 | 75.87 ± 0.09 |
| KDCL_θ1 | 76.61 ± 0.18 | 76.69 ± 0.17 | 78.11 ± 0.08 | 70.40 ± 0.22 | 70.37 ± 0.29 | 79.90 ± 0.17 |
| KDCL_θ2 | 77.75 ± 0.17 | 77.93 ± 0.19 | 77.97 ± 0.18 | 70.31 ± 0.31 | 76.99 ± 0.22 | 80.19 ± 0.02 |
| FFCL_θf (ours) | 77.65 ± 0.26 | 77.34 ± 0.10 | 77.90 ± 0.04 | 70.39 ± 0.03 | 71.18 ± 0.17 | 79.04 ± 0.11 |
| FFCL_θf2 (ours) | 77.52 ± 0.17 | 78.05 ± 0.19 | 78.24 ± 0.18 | 70.31 ± 0.19 | 77.50 ± 0.01 | 79.19 ± 0.17 |
| FFCL_θf (ours) | 79.34 ± 0.18 | 79.53 ± 0.07 | 79.98 ± 0.10 | 72.36 ± 0.14 | 77.52 ± 0.10 | 81.37 ± 0.17 |

WRN: WideResNet; KD: knowledge distillation; DML: deep mutual learning; FFL: feature fusion learning; KDCL: knowledge distillation via collaborative learning.

In each column, the highest score is in boldface, and the score with underline is the highest among the competing methods except ours.
obtains the counterpart with two peer networks. Thus, the comparative experiments on the peer architectures of two and three student networks show that our proposed feature fusion-based collaborative distillation is very effective.

Ablation study

Our FFCL framework transfers variety kinds of knowledge while training the peer networks through collaborative learning. We verify the importance of each kind of knowledge with a set of ablation studies in this section. As shown in Table 5, we carried out experiments under seven settings. The two peer networks were set to ResNet18. RK means the response knowledge, namely, the knowledge of the output layer transferred between networks, it corresponds to \( \theta_{k_{u}} \) in equation (3) or \( \theta_{k_{v}} \) in equation (4). FK indicates the feature knowledge, that is, the feature knowledge of the middle layers transferred between networks via the fused module, which corresponds to \( L^{f}_{k_{u}} \) in equation (1) or \( L^{f}_{k_{v}} \) in equation (2). NRK represents the network regularization knowledge corresponding to \( L^{r}_{k_{u}} \) in equation (7) or \( L^{r}_{k_{v}} \) in equation (8). We excluded one or two of those three components of FFCL in turn, which gives us seven variations indicated by Cases A–G in Table 5. Specifically, these cases are described as follows:

- Case A represents our proposed FFCL scheme with the objective function \( L^{D}_{\theta_{u}} \) in equation (9) for network \( \theta_{k} \) and \( L^{D}_{\theta_{v}} \) in equation (12) for network \( \theta_{k'_{v}} \).
- Case B only retains the feature knowledge and the network regularization knowledge, where the objective function for the peer network \( \theta_{k} \) is formulated as \( aL^{f}_{\theta_{u}} + bL^{r}_{\theta_{u}} + cL^{E}_{\theta_{u}} + dL^{E}_{\theta_{v}} \).

### Table 5. Ablation study of FFCL in terms of the average top-1 accuracy over three individual runs with the corresponding standard deviations on CIFAR-100. ResNet18-ResNet18 was selected as the peer architecture.

| Case | RK | FK | NRK | FFCL\_\( \theta_{1} \) | FFCL\_\( \theta_{2} \) | FFCL\_\( \theta_{f} \) |
|------|----|----|-----|----------------|----------------|----------------|
| A    | ✓  | ✓  | ✓   | 77.65 ± 0.26 | 77.52 ± 0.17 | 79.34 ± 0.18 |
| B    | ×  | ✓  | ✓   | 75.58 ± 0.22 | 75.99 ± 0.14 | 78.38 ± 0.10 |
| C    | ✓  | ×  | ✓   | 77.23 ± 0.23 | 77.14 ± 0.05 | —              |
| D    | ✓  | ×  | ×   | 76.91 ± 0.15 | 76.88 ± 0.02 | 78.84 ± 0.23 |
| E    | ✓  | ×  | ×   | 76.25 ± 0.16 | 76.04 ± 0.24 | —              |
| F    | ×  | ✓  | ×   | 75.43 ± 0.29 | 75.52 ± 0.33 | 77.53 ± 0.14 |
| G    | ×  | ×  | ✓   | 76.64 ± 0.07 | —              | —              |

RK: response knowledge; FK: feature knowledge; NRK: network regularization knowledge; FFCL: feature fusion-based collaborative learning. The bold values are indicate that in the ablation experiments, the complete model we proposed can achieve the best performance in the experiments.

### Table 4. The average top-1 accuracy (%) of FFL, DML, and our FFCL using three peer student networks over three individual runs with the corresponding standard deviations on CIFAR-100 data set.

| Network1 (\( \theta_{1} \)) | ShuffleNet | ResNet18 | ShuffleNet |
|-----------------------------|------------|----------|------------|
| Network2 (\( \theta_{2} \)) | ResNet18  | ResNet18 | ResNet18   |
| Network3 (\( \theta_{3} \)) | ResNet18  | ResNet18 | ResNet34   |
| DML\_\( \theta_{1} \)       | 70.64 ± 0.24 | 76.60 ± 0.17 | 70.79 ± 0.10 |
| DML\_\( \theta_{2} \)       | 76.47 ± 0.12 | 76.34 ± 0.19 | 76.31 ± 0.49 |
| DML\_\( \theta_{3} \)       | 76.54 ± 0.14 | 76.56 ± 0.06 | 78.05 ± 0.33 |
| FFL\_\( \theta_{1} \)       | 70.19 ± 0.34 | 75.91 ± 0.23 | 76.07 ± 0.20 |
| FFL\_\( \theta_{2} \)       | 75.42 ± 0.05 | 75.82 ± 0.33 | 75.65 ± 0.18 |
| FFL\_\( \theta_{3} \)       | 75.83 ± 0.18 | 75.33 ± 0.37 | 76.90 ± 0.25 |
| KDCL\_\( \theta_{1} \)      | 77.73 ± 0.19 | 78.41 ± 0.12 | 78.41 ± 0.23 |
| KDCL\_\( \theta_{2} \)      | 70.40 ± 0.04 | 76.85 ± 0.11 | 76.04 ± 0.30 |
| KDCL\_\( \theta_{3} \)      | 77.02 ± 0.26 | 77.19 ± 0.13 | 77.14 ± 0.24 |
| FFCL\_\( \theta_{1} \) (ours)| 71.26 ± 0.37 | 77.37 ± 0.13 | 71.04 ± 0.21 |
| FFCL\_\( \theta_{2} \) (ours)| 77.43 ± 0.34 | 77.21 ± 0.12 | 77.25 ± 0.18 |
| FFCL\_\( \theta_{3} \) (ours)| 77.29 ± 0.34 | 77.44 ± 0.15 | 77.85 ± 0.16 |
| FFCL\_\( \theta_{f} \) (ours)| 79.21 ± 0.28 | 79.86 ± 0.13 | 79.50 ± 0.13 |

DML: deep mutual learning; FFL: feature fusion learning; KDCL: knowledge distillation via collaborative learning. The highest score are in bold.
• Case C keeps the response knowledge and the network regularization knowledge, where the objective function for the peer network $\theta_k$ is formulated as $aL_{\theta_k} + bL_{\theta_k} + cL_{\theta_k}^{CE}$.
• Case D excludes the network regularization knowledge where the objective function for the peer network $\theta_k$ is formulated as $aL_{\theta_k}^{col} + bL_{\theta_k}^{CE} + cL_{\theta_k}^{CE}$.
• Case E only includes the response knowledge where the objective function for the peer network $\theta_k$ is formulated as $aL_{\theta_k} + bL_{\theta_k}^{CE}$.
• Case F only includes the feature knowledge where the objective function for the peer network $\theta_k$ is formulated as $aL_{\theta_k}^{col} + bL_{\theta_k}^{CE} + cL_{\theta_k}^{CE}$.
• Case G only includes the network regularization knowledge where the objective function for the peer network $\theta_k$ is formulated as $aL_{\theta_k} + bL_{\theta_k}^{CE}$.

It should be noted that $a$ to $g$ above are parameters in the variant cases.

According to the ablation results in Table 5, the removal of any knowledge will cause performance degradation. More importantly, the FFCL $\theta_f$ via the fused feature knowledge transfer performs better than FFCL $\theta_k$ and FFCL $\theta_k$ with and without using fused feature. Thus, these ablation results verify the effectiveness of our proposed FFCL method via collaborative learning, network regularization, and feature fusion.

**Conclusion**

In this article, we have proposed a novel KD framework called FFCL. Through collaborative learning, the proposed FFCL method effectively concatenates the features of peer networks to generate a more expressive feature map for transferring feature knowledge between peer networks. Meanwhile, it also transfers the response knowledge in the output layers between the peer networks during the distillation process. We have also introduced a regularization process, which eliminates the trouble of training for a large teacher network and provides a positive knowledge for the student peer network, leading to a further improved performance. The insights of collaborative KD given by FFCL can potentially facilitate the future works.

**Declaration of conflicting interests**

The author(s) declared no potential conflicts of interest with respect to the research, authorship, and/or publication of this article.

**Funding**

The author(s) disclosed receipt of the following financial support for the research, authorship, and/or publication of this article: The work was, in part, supported by the National Natural Science Foundation of China (grant nos 61976107 and 61502208) and the Postgraduate Research and Practice Innovation Program of Jiangsu Province (grant no. KYCX20_3085).

**ORCID iD**

Jianping Gou https://orcid.org/0000-0002-8438-7286

**References**

1. Jung S, Lee D, Park T, et al. Fair feature distillation for visual recognition. In: Proceedings of the IEEE conference on computer vision and pattern recognition (CVPR), New Orleans, LA, 20–25 June 2021, pp.12115–12124. New York: IEEE.
2. Aljundi MF and Huchaiah MD. Multi-model deep learning approach for collaborative filtering recommendation system. CAAI Trans Intell Technol 2020; 5(4): 268–275.
3. Wu X, Liu J, Li X, et al. Improving stylized neural machine translation with iterative dual knowledge transfer. In: Proceedings of the 30th international joint conference on artificial intelligence (IJCAI), Montreal, QC, Canada, 19–27 August 2021, pp.3971–3977.
4. Han S, Pool J, Tran J, et al. Learning both weights and connections for efficient neural networks. In: Proceedings of the 28th international conference on neural information processing systems (NIPS), Montreal, QC, Canada, 7–12 December 2015, pp.1135–1143. Cambridge, MA: The MIT Press.
5. Li H, Kadav A, Durdanovic I, et al. Pruning filters for efficient ConvNets. In: Proceedings of the international conference on learning representations (ICLR), Toulon, 24–26 April 2017.
6. Wen W, Xu C, Wu C, et al. Coordinating filters for faster deep neural networks. In: Proceedings of the IEEE international conference on computer vision (ICCV), Venice, 22–29 October 2017, pp.658–666. New York: IEEE.
7. Grachev AM, Ignatov DI, Savchenko AV, et al. Neural networks compression for language modeling. In: Proceedings of the international conference on pattern recognition and machine intelligence (PReMI), Kolkata, India, 5–8 December 2017, pp.1135–1143. Cham: Springer.
8. Hinton G, Vinyals O and Dean J. Distilling the knowledge in a neural network, https://arxiv.org/abs/1503.02531
9. Romero A, Ballas N, Kahou SE, et al. FitNets: hints for thin deep nets. In: Proceedings of the international conference on learning representations (ICLR), San Diego, CA, 7–9 May 2015.
10. Chen D, Mei J-P, Zhang Y, et al. Cross-layer distillation with semantic calibration. In: Proceedings of the 35th AAAI conference on artificial intelligence: a virtual conference, Vancouver, BC, Canada, 2–9 February 2021, pp.7028–7036. Menlo Park, CA: Association for the Advancement of Artificial Intelligence (AAAI).
11. Shen C, Wang X, Yin Y, et al. Progressive network grafting for few-shot knowledge distillation. In: Proceedings of the 35th AAAI conference on artificial intelligence: a virtual conference, Vancouver, BC, Canada, 2–9 February 2021.
12. Jafari A, Rezagholizadeh M, Sharma P, et al. Annealing knowledge distillation. In: Proceedings of the European conference on artificial life (ECAL), Klagenfurt, 17–20 May 2021. IEEE.

13. Zhang Y, Xiang T, Hospedales TM, et al. Deep mutual learning. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition (CVPR), Salt Lake City, UT, 18–23 June 2018, pp.4320–4328. New York: IEEE.

14. Guo Q, Wang X, Wu Y, et al. Online knowledge distillation via collaborative learning. In: Proceedings of the IEEE conference on computer vision and pattern recognition (CVPR), Salt Lake City, UT, 18–23 June 2018, pp.4320–4328. New York: IEEE.

15. Hou S, Liu X and Wang Z. DualNet: learn complementary features for image recognition. In: Proceedings of the IEEE international conference on computer vision (ICCV), Venice, 22–29 October 2017, pp.502–510. New York: IEEE.

16. Kim J, Hyun M, Chung I, et al. Feature fusion for online mutual knowledge distillation. In: Proceedings of the international conference on pattern recognition (ICPR), Milan, 10–15 January 2021, pp.4619–4625. IEEE.

17. Ma N, Zhang X, Zheng H-T, et al. ShuffleNet V2: practical guidelines for efficient CNN architecture design. In: Proceedings of the European conference on computer vision (ECCV), Munich, 8–14 September 2018, pp.122–138. Cham: Springer.

18. Gou J, Yu B, Maybank SJ, et al. Knowledge distillation: a survey. Int J Comput Vision 2021; 129(6): 1789–1819.

19. Yuan L, Tay FE, Li G, et al. Revisiting knowledge distillation via label smoothing regularization. In: Proceedings of the IEEE conference on computer vision and pattern recognition (CVPR), Seattle, WA, 14–19 June 2020, pp.3909–3911. New York: IEEE.

20. Zhang L, Song J, Gao A, et al. Be your own teacher: improve the performance of convolutional neural networks via self distillation. In: Proceedings of the IEEE/CVF international conference on computer vision (ICCV), Seoul, South Korea, 27 October–2 November 2019, pp.3712–3721. New York: IEEE.

21. Phuong M and Lampert CH. Distillation-based training for multi-exit architectures. In: Proceedings of the IEEE/CVF international conference on computer vision (ICCV), Seoul, South Korea, 27 October–2 November 2019, pp.1355–1364. New York: IEEE.

22. Yun S, Park J, Lee K, et al. Regularizing class-wise predictions via self-knowledge distillation. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition (CVPR), Seattle, WA, 14–19 June 2020, pp.13873–13882. New York: IEEE.

23. Boo Y, Shin S, Choi J, et al. Stochastic precision ensemble: self-knowledge distillation for quantized deep neural networks. In: Proceedings of the 35th AAAI conference on artificial intelligence: a virtual conference, Vancouver, BC, Canada, 2–9 February 2021, pp.6794–6802. Menlo Park, CA: Association for the Advancement of Artificial Intelligence (AAAI).

24. Wu G and Gong S. Peer collaborative learning for online knowledge distillation. In: Proceedings of the 35th AAAI conference on artificial intelligence: a virtual conference, Vancouver, BC, Canada, 2–9 February 2021, pp.1–9. Menlo Park, CA: Association for the Advancement of Artificial Intelligence (AAAI).

25. Lan X, Zhu X and Gong S. Knowledge distillation by on-the-fly native ensemble. In: Proceedings of the 32nd international conference on neural information processing systems (NIPS), Montreal, QC, Canada, 3–8 December 2018, pp.7517–7527. Red Hook, NY: Curran Associates, Inc.

26. Chen D, Mei J, Wang C, et al. Online knowledge distillation with diverse peers. In: Proceedings of the 35th AAAI conference on artificial intelligence: a virtual conference, Vancouver, BC, Canada, 2–9 February 2021, pp.7028–7036. Menlo Park, CA: Association for the Advancement of Artificial Intelligence (AAAI).

27. Krizhevsky A and Hinton G. Learning multiple layers of features from tiny images. Master’s thesis, Department of Computer Science, University of Toronto, 2009, Citeseer.

28. He K, Zhang X, Ren S, et al. Deep residual learning for image recognition. In: Proceedings of the IEEE conference on computer vision and pattern recognition (CVPR), Las Vegas, NV, 27–30 June 2016, pp.770–778. New York: IEEE.

29. Zagoruyko S and Komodakis N. Wide residual networks. In: Proceedings of the British machine vision conference (BMVC), York, 19–22 September 2016. BMVA Press.