Learn by Oneself: Exploiting Weight-Sharing Potential in Knowledge Distillation Guided Ensemble Network

Qi Zhao, *Member, IEEE*, Shuchang Lyu, *Graduate Student Member, IEEE*, Lijiang Chen, *Member, IEEE*, Binghao Liu, Ting-Bing Xu, Guangliang Cheng, and Wenquan Feng

***Abstract*— Recent CNNs (convolutional neural networks) have become more and more compact. The elegant structure design highly improves the performance of CNNs. With the development of knowledge distillation technique, the performance of CNNs gets further improved. However, existing knowledge distillation guided methods either rely on offline pretrained high-quality large teacher models or online heavy training burden. To solve the above problems, we propose a feature-sharing and weight-sharing based ensemble network (training framework) guided by knowledge distillation (EKD-FWSNet) to make baseline models stronger in terms of representation ability with less training computation and memory cost involved. Specifically, motivated by getting rid of the dependence of offline pretrained teacher model, we design an end-to-end online training scheme to optimize EKD-FWSNet. Motivated by decreasing the online training burden, we only introduce one auxiliary classmate branch to construct multiple forward branches, which will then be integrated as ensemble teacher to guide baseline model. Compared to previous online ensemble training frameworks, EKD-FWSNet can provide diverse output predictions without relying on increasing auxiliary classmate branches. Motivated by maximizing the optimization power of EKD-FWSNet, we exploit the representation potential of weight-sharing blocks and design efficient knowledge distillation mechanism in EKD-FWSNet. Extensive comparison experiments and visualization analysis on benchmark datasets (CIFAR-10/100, tiny-ImageNet, CUB-200 and ImageNet) show that self-learned EKD-FWSNet can boost the performance of baseline models by large margin, which has obvious superiority compared to previous related methods. Extensive analysis also proves the interpretability of EKD-FWSNet. Our code is available at https://github.com/cv516Buaa/EKD-FWSNet.**

***Index Terms*— Knowledge distillation, ensemble learning, weight-sharing blocks, high-efficiency network.**

Manuscript received 26 December 2022; revised 13 March 2023; accepted 11 April 2023. Date of publication 14 April 2023; date of current version 30 October 2023. This work was supported in part by the National Natural Science Foundation of China under Grant 62072021 and in part by the Beijing Natural Science Foundation under Grant 4214074. This article was recommended by Associate Editor H. Zhao. *(Corresponding authors: Lijiang Chen; Ting-Bing Xu.)*

Qi Zhao, Shuchang Lyu, Lijiang Chen, Binghao Liu, and Wenquan Feng are with the Department of Electronics and Information Engineering, Beihang University, Beijing 100191, China (e-mail: zhaoqi@buaa.edu.cn; lyushuchang@buaa.edu.cn; chenlijiang@buaa.edu.cn; liubinghao@buaa.edu.cn; buaafwq@buaa.edu.cn).

Ting-Bing Xu is with the Department of Instrumentation and Optoelectronic Engineering, Beihang University, Beijing 100191, China (e-mail: tingbing_xu@buaa.edu.cn).

Guangliang Cheng is with the Department of Computer Science, University of Liverpool, L69 3BX Liverpool, U.K. (e-mail: guangliangcheng2014@gmail.com).

Color versions of one or more figures in this article are available at https://doi.org/10.1109/TCSVT.2023.3267115.

Digital Object Identifier 10.1109/TCSVT.2023.3267115

I. Introduction

RECENTLY, deep CNNs have shown strong power in computer vision tasks, such as image recognition [1], [2], [3], [4], [5], [6], [7], [8], object detection [9], [10], [11], [12] and semantic segmentation [13], [14], [15], [16], [17], [18]. However, many high-performance networks always accompany with enormous trainable parameters, computation cost and complex modules. In real applications, especially applications using embedded devices, these networks are hard to apply. To meet the demand of resource-limited devices, researchers engage in designing networks with high performance and compact structure.

As a strong method for model generalization enhancement, knowledge distillation is widely applied in resource-constraint application. With the guidance of large-scale well-optimized “teacher” models, smaller “student” models can gain considerable performance improvement without using more trainable parameters and complex modules or operations. Fig.1 shows two main paradigms of current knowledge distillation guided training frameworks. The **top-left** diagram indicates offline “teacher-student” training framework, where student model is optimized to approximate large-scale and mature teacher model. The **top-right** diagram indicates online “ensemble teacher-student” training framework [19], [20], [21], [22], [23], [24], where student model is part of a multi-branch ensemble network and gains knowledge on-the-fly from “ensemble teacher”. This diagram focuses on utilizing the “ensemble teacher” to gain sub-optimal generalized knowledge due to the diversity and redundancy representation of multi-branch structure.

Although previous knowledge distillation guided training frameworks are notable and efficient, there are still some weaknesses: (1) Offline “teacher-student” training framework (**top-left** diagram of Fig.1) depends highly on cumbersome teacher models, which complicates the training process. Intuitively, when we expect to train a stronger student baseline model, designing and training a much stronger teacher model will become harder. (2) Online “ensemble teacher-student” training framework (Fig.1 **top-right**) brings in more training burden. When classmate branches increase, the training computation will rapidly increase. Additionally, distillation loss terms will also rapidly increase when more branches get involved, which also makes the training process more and more complicated.

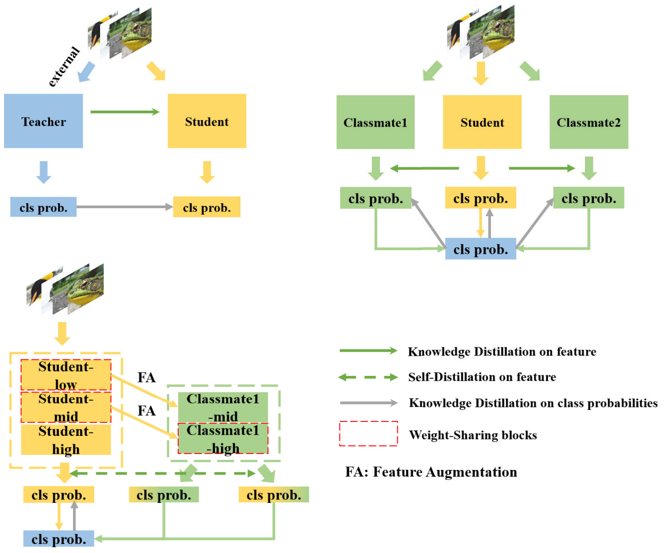

Fig. 1. The diagram of recent notable knowledge distillation guided training frameworks and EKD-FWSNet. Diagrams at **top-left**, **top-right** and **bottom** respectively denote offline teacher-student training framework, student-classmate ensemble training framework and EKD-FWSNet.

The weaknesses mentioned above motivate us to propose an easy-optimized and high-efficiency training framework guided by knowledge distillation. Specifically, motivated by not depending on offline "teacher-student" training framework, we will design an online training scheme. In this scheme, All baseline models will be optimized in an end-to-end manner without training teacher models in advance. Motivated by easing the online training burden, we only use one auxiliary classmate branch to simplify our proposed architecture and design less distillation loss functions to simplify the training process. Based on an efficient architecture, maximizing the optimization power by exploring the representation potential of weight-sharing blocks is another motivation of our method.

Based on above-mentioned motivations, we propose a novel feature-sharing and weight-sharing based ensemble network guided by knowledge distillation strategy (EKD-FWSNet), which can make baseline models learn by themselves and get stronger. As shown in **bottom** diagram of Fig.1, we construct a student-classmate training framework with multiple forward branches using weight-sharing blocks. With one auxiliary classmate branch, we can construct several forward branches and utilize the representation diversity of different predictions to obtain a high-performance ensemble teacher. Specifically, we first split the main student (baseline model) and classmate branch with several branch points respectively. Then, we create shortcut-connections between two neighboring branch points respectively from classmate and main student branch to construct multiple forward branches. During training, we employ knowledge distillation mechanism to guide the optimization of baseline model. Predicted posterior probabilities from different forward branches are integrated as ensemble teacher to transfer more mature global information to main student. Intermediate feature maps are also integrated as ensemble teacher to guide the main student in semantic level. Particularly, we only distill integrated knowledge to main student branch. Designing like this, EKD-FWSNet has efficient and concise knowledge distillation loss functions where these loss function number will not increase when more forward branches are involved. During inference, the auxiliary classmate branch will be removed, so no extra computation cost will be involved in main student.

In EKD-FWSNet, we use weight-sharing blocks to simplify ensemble network, which can ease the training burden with less training memory and computation cost. Previous ensemble training framework (Fig.1 **top-right**) [19], [20], [21], [22], [23], [24] constructs multiple forward branches by introducing multiple independent auxiliary classmate branches. Compared to these networks, EKD-FWSNet utilizes weight-sharing blocks to reduce the number and size of auxiliary classmate branches. Moreover, when more forward branches are required, previous ensemble training frameworks have to employ more classmate branches while EKD-FWSNet only needs to set more branch points and connect more neighboring branch points. Obviously, EKD-FWSNet is more efficient and flexible.

In EKD-FWSNet, we explore the representation potential of weight-sharing blocks. As shown in Fig.1 **bottom**, weight-sharing blocks in EKD-FWSNet are components existing in more than one forward branches. During forward pass, weight-sharing blocks provide more than one output feature maps. During back propagation, different gradient terms from different backward paths are integrated together to update the trainable parameters of weight-sharing blocks. Basically, trainable parameters of weight-sharing blocks in EKD-FWSNet have the representation potential to fit in with different gradient orientations. This proves that multiple forward branches can provide diverse predictions to form a more generalized ensemble teacher. However, the diversity decrease resulted from deeper weight-sharing blocks may harm the performance of ensemble teacher. To compensate this decrease, we implement online feature augmentation by inserting feature alignment (FA) modules including squeeze-and-excitation (SE) block [25], channel attention module (CAM) [17] and Dropout [26] block after weight-sharing blocks.

To verify the performance of EKD-FWSNet, we conduct extensive experiments on benchmark datasets (CIFAR-10/100 [27], tiny-ImageNet[1]), CUB-200 [28] and ImageNet [81]. Here, we select notable baseline models (E.g., ResNet [4], [30] and EfficientNet [31]) to construct EKD-FWSNet. Comparison experiments clearly show that baseline models training in EKD-FWSNet gain huge improvement. In some lightweight baseline models such as ResNet-20 and ResNet32, the classification accuracy can surprisingly improve by more than 4%, which surpasses the previous state-of-the-art methods. Some high-efficiency baseline models can improve by more than 8% on CUB-200, which is also encouraging.

This paper is extended from our conference paper [32]. The extensions are summarized as follows: (1) We make detailed mathematical analysis and explanation on the self-learned mechanism of EKD-FWSNet in terms of weight-sharing blocks. (2) To further make the performance of EKD-FWSNet

[1]https://tiny-imagenet.herokuapp.com

more convincing, we add more comparison experiments on two large-scale benchmark datasets, which are CUB-200 [28] and ImageNet [81]. (3) Besides constructing EKD-FWSNet on lightweight and high-efficiency models, we supplement some large-scale baseline models (E.g., ResNet-50, ResNet-101 [4]) based EKD-FWSNet to prove the generalization and robustness. (4) We further add comparison with dynamic-routing based dynamic neural networks including more related works and experiments (Appendix). (5) We add more comparison experiments with some most recent methods.

The main contributions of our paper are listed as follows.

- We maximally exploit the potential of weight-sharing blocks to ease the training burden and provide an insight about designing easy-optimized knowledge distillation guided ensemble training framework.
- We add more abundant experiments on two large-scale benchmark datasets, CUB-200 and ImageNet. We also add more large-scale models besides lightweight and high-efficiency models to construct EKD-FWSNet. All this extended experiments make our proposed method more convincing.
- We add detailed theoretical explanation and mathematical analysis on weight-sharing blocks, which proves the interpretability of EKD-FWSNet.
- Baseline models optimizing in EKD-FWSNet make huge progress than individually optimizing. On notable benchmark classification datasets, EKD-FWSNet performs better than previous SOTA methods.

II. Related Work

A. High-Efficiency Networks

High-efficiency networks can work well on resource-limited embedded devices with compact structure. Recently, various networks such as SqueezeNet [33], MobileNet [34], [35], [36], ShuffleNet [37] and EfficientNet [31] utilize elegant structure to make the model perform well with less cost. From another perspective, some researchers focus on applying low-bit technique on baseline model [38], [39], [40] to directly exponentially accelerate the inference. In addition, network pruning [41], [42], [43] strategy is also widely studied aiming at constructing high-efficiency models by removing redundant trainable parameters or complex operations. In this paper, we also aim to design high-efficiency network. Differently, we do not focus on designing compact network components or using pruning and quantization. We mainly focus on exploring optimizing and learning power.

B. Knowledge Distillation Guided Training Framework

Knowledge distillation (KD) is a classical technique [44] to transfer more mature information from a large-scale teacher model to a smaller-scale student model. The **top-left** diagram shows offline knowledge distillation guided training framework. Reference [45] integrate knowledge distillation and logits-mimicking to train a compact model by approximating another larger, more complex model. References [46], [47], [48], [49], [50], [51], [52], and [53] apply knowledge distillation both on posterior logits and intermediate feature maps for feature consistency in each stage of smaller compact model. Besides directly distilling knowledge on logits or intermediate feature maps, many methods exploit knowledge distillation on second-order feature. Reference [54] propose AFD (attention-based feature distillation), which utilizes relative similarities between features of teacher and students to control distillation intensities of feature pairs. References [48] and [55] propose pixel-wise distillation and holistic distillation respectively on similarity feature and holistic embedding.

Recently, many notable works investigate online knowledge distillation training framework integrating ensemble learning strategy to optimize models in an end-to-end manner without depending on pretrained teacher models (Fig.1 **top-right**) [19], [56], [57]. Specifically, [58] and [59] propose an elegant knowledge distillation guided training framework to improve high-efficiency models by exploring the representation invariance within baseline model itself. Similarly, FRSKD [60] designs a self-distillation network to provide refined feature maps to distill itself on intermediate feature maps. MixSKD [61] integrates Mixup augmentation [62] and self-distillation mechanism. With this novel idea, MixSKD improves baseline models by large margin and achieves competitive results on image classification tasks. References [63] and [64] construct ensemble teacher from iterative snapshots and apply knowledge distillation from ensemble teacher to baseline models. This design can improve the predictive power of compact models. References [20], [21], [22], [23], and [24] all design student-classmate ensemble training framework to obtain knowledge of ensemble teacher, which can guide both student and classmate efficiently in an end-to-end manner. AFID [65] directly employs one more complete sub-net to construct a two-branch ensemble training network. Besides distilling knowledge from ensemble teacher, it further proposes a feature interaction module to employ mutual learning between attentive feature maps of two sub-nets. PCL [66] is another outstanding method. Besides generating ensemble teacher, it adds a peer mean teacher, which is a temporal mean model updated by EMA (Exponential Moving Average) technique. This peer mean teacher can further boost the performance of baseline model through transferring knowledge collaboratively. In this paper, our proposed self-learned training framework also belongs to student-classmate ensemble training framework. Therefore, we mainly compare our EKD-FWSNet with the above-mentioned online ensemble training frameworks to show the superiority of our method.

C. Unsupervised Representation Learning Methods

Our proposed training framework is also related to some notable unsupervised representation learning methods [67], [68], [69], [70] in terms of knowledge distillation and self-supervised learning strategy among branches. The differences can be listed in the following. First, unsupervised representation learning methods utilize contrastive learning to bidirectionally transfer knowledge among feature prototypes of original image and transformed images. This design can enhance the generalization ability through maximally exploiting the inter-class variation and intra-class similarity in

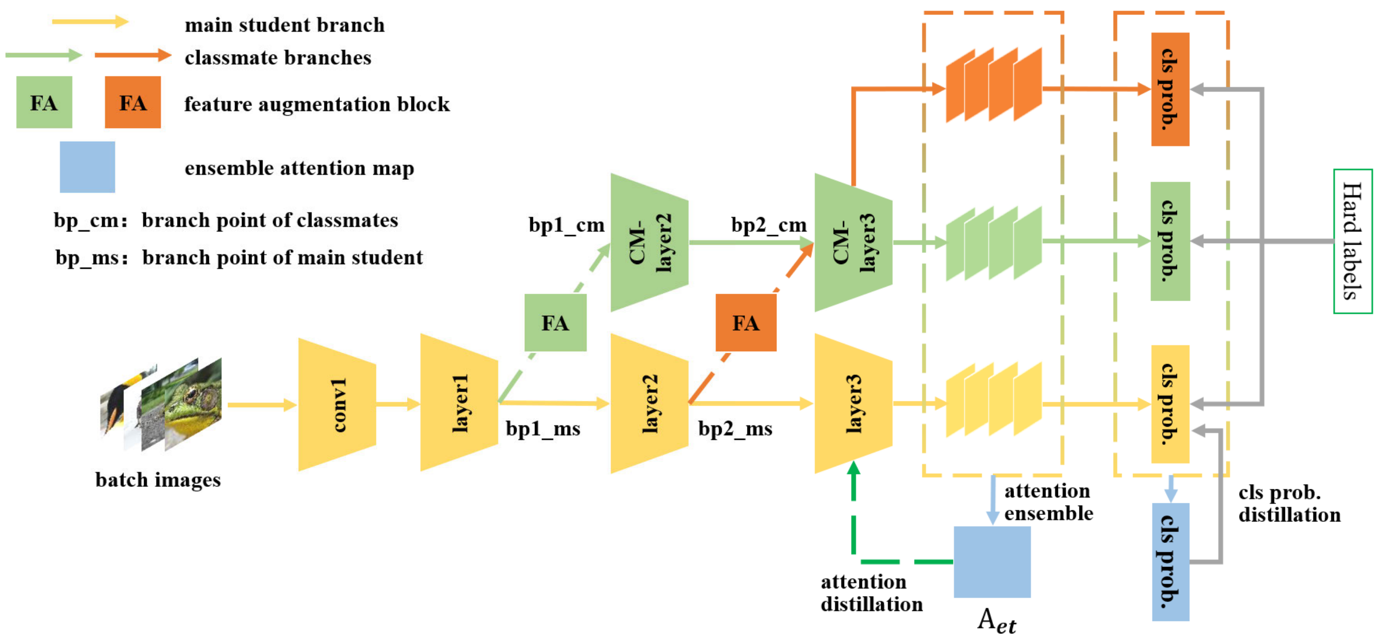

Fig. 2. **Overview of EKD-FWSNet.** In EKD-FWSNet, "bp1_ms", "bp2_ms", "bp1_cm" and "bp2_cm" are branch points on main student and classmate branch. We construct three forward branches by creating shortcuts between two branches ("bp1_ms" and "bp1_cm"; "bp2_ms" and "bp2_cm"). On classmate branch, CM_layer2 and CM_layer3 respectively have same type of convolutional blocks as layer1 and layer2. To optimize EKD-FWSNet, we apply hard-label learning on posterior class probabilities and soft-label distillation on both posterior class probabilities distillation and intermediate feature maps.

embedded space. In EKD-FWSNet, we use knowledge distillation mechanism mainly aiming at transferring knowledge from sub-optimal ensemble teacher to naive baseline model. Second, we actually have different motivations of constructing multiple forward branches with unsupervised representation learning methods. Specifically, multiple branches in their architectures is to provide multiple transformations of images. Multiple branches in our architecture mainly aim to provide diverse predicted class probabilities to obtain a sub-optimal ensemble teacher.

D. Dynamic-Routing Based Dynamic Neural Network

Dynamic neural network (DNN) [71] can adapt their structures during inference according to the configuration of computation and memory budgets of embedded devices. Dynamic-routing based DNN focuses on dynamically extracting the different-depth/width sub-nets inside a deep "Super-Net". SkipNet [72] utilizes the gating policy to selectively skip some redundant blocks and further combines reinforcement learning with supervised-learning task for non-differentiable skipping decisions. Similarly, ConvNet-AIG and Batch-shaping [73], [74] can adaptively decide the network topology conditioned on the input image. Recently, combining knowledge distillation and dynamic-routing based DNN becomes more and more popular. BYOT and slimmable neural network [76], [80] dynamically generate several small student models from a large-size teacher model in depth-level or width-level and distill knowledge from teacher to each student, which can boost the performance of smaller student model without harming the large teacher model. In this paper, the proposed EKD-FWSNet can be regraded as dynamic-routing architecture, because each forward branch can be regarded as a sub-net inside the whole ensemble training framework. To some degree, we are inspired by the architecture of dynamic-routing based DNN. Therefore, we conduct experiments in appendix to compare EKD-FWSNet with BYOT [80], which is a notable dynamic-routing based DNN related to our work.

III. Proposed Method

A. Architecture of EKD-FWSNet

To improve the representation power of baseline model without adding extra trainable parameters and designing complex modules, we propose EKD-FWSNet which is a self-learned efficient training framework guided by knowledge distillation. To construct EKD-FWSNet (Fig.2), we respectively set two branch points on main student branch (bp1_ms, bp2_ms) and classmate branch (bp1_cm, bp2_cm). We then create connections between the two neighboring branch points from main student and classmate branch (connections between bp1_ms and bp1_cm; connections between bp2_ms and bp2_cm). In this way, we construct a three-forward-branch ensemble training framework, where the three forward branches include one forward branch of main student (baseline model) ("conv1→layer1→layer2→layer3") and two forward branches of classmate ("conv1→layer1→CM_layer2→CM_layer3"; "conv1→ layer1→layer2→CM_layer3"). To flexibly construct a training framework containing more forward branches, we can create more connections between the main student and classmate branch by setting more branch points. In addition, weight-sharing blocks exist in more than one forward branches, so they have to adapt with different branches. In EKD-FWSNet, we maximally exploit the adaptive representation potential of weight-sharing blocks.

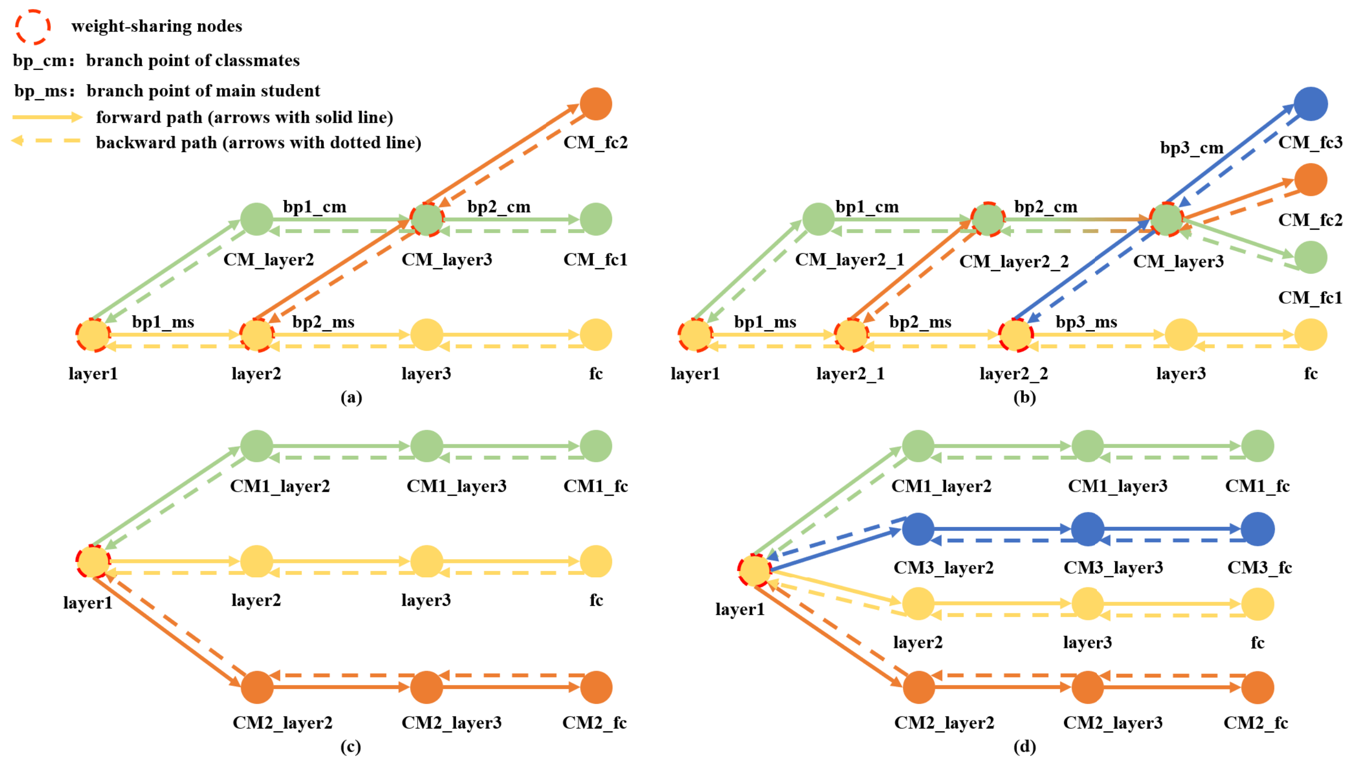

Fig. 3. **The comparison node diagram between EKD-FWSNet and previous ensemble training framework.** In this diagram, the nodes denote convolutional blocks (layers) of main student and classmate branches. Specifically, diagram (a) and (b) respectively denote three-forward-branch and four-forward-branch EKD-FWSNet. Diagram (c) and (d) respectively denote previous ensemble training framework. Through connecting neighboring branch points from main student and classmate branch, we can create more forward branches without using any more trainable parameters.

During training, the batch images first pass through "conv1 layer1". Then, the feature maps will be fed into every forward branches to generate different predictions. During inference, we will remove classmate branch and only leave the blocks on main student branch.

B. Weight-Sharing Blocks of EKD-FWSNet

Previous online "ensemble teacher-student" training frameworks (Fig.1 **top-right**) depend on auxiliary classmate branches to obtain multiple predictions for ensemble teacher. However, when more predictions are required, more auxiliary classmate branches will be involved, which lead to exponential cost on training memory and computation. Additionally, distillation loss functions also increase rapidly with more classmate branches involved, which will confuse the optimization. To ease the training burden and design a high-efficiency knowledge distillation guided ensemble training framework, we exploit the potential of weight-sharing blocks. In this paper, weight-sharing blocks indicate convolutional blocks containing several convolution units, batch normalization units and activation functions, which exists in more than one forward branches. As shown in Fig.2, layer1 and layer2 are two weight-sharing blocks in main student branch, while CM_layer3 is weight-sharing block in classmate branch. To show the superiority of weight-sharing blocks in a concise way, we use node diagram (Fig.3). From Fig.3, we find that: (1) Previous ensemble training frameworks (diagram (c) and (d)) construct multiple forward branches completely depending on multiple auxiliary classmate branches; (2) Our training framework (diagram (a) and (b)) only depends on one auxiliary classmate branch. For more forward branches, we simply split the main student and classmate branch with more branch points. (3) From diagram (a) to (b), it is clear that even when more predictions are required, no extra training parameters will be involved into EKD-FWSNet and the computation cost increase in a slow way. From diagram (c) to (d), it shows that when more forward branches are constructed, previous framework has to increase a complete classmate branch with huge memory and computation cost. As a whole, utilizing weight-sharing blocks is efficient on obtaining more generalized knowledge by constructing more forward branches with less trainable parameters and computation cost.

Convolutional blocks in EKD-FWSNet have four types. As shown in Fig.4, common blocks (Fig.4 diagram (a)) have one input and one output feature map existing only in one forward branch while weight-sharing blocks (Fig.4 diagram (b), (c), (d)) have more than one input or output existing in more than one forward branches. Specifically, more out-degrees of weight-sharing blocks (diagram (b)) bring in more gradients terms from more backward paths. It explores the representation potential of weight-sharing blocks to fit in with different gradient orientations. More in-degrees of weight-sharing blocks (diagram (c) and (d)) bring in different input feature maps from more forward branches. It explores the potential of weight-sharing blocks to represent feature maps with different distributions. All in all, we exploit the representation potential of weight-sharing blocks to make

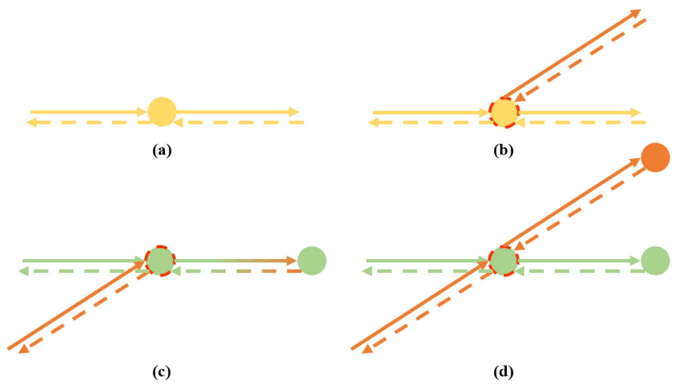

Fig. 4. **Node diagrams of convolutional blocks in EKD-FWSNet.** Diagram (a) denotes common blocks with one in-degree and one out-degree. Diagram (b) denotes weight-sharing blocks with one in-degree and two out-degrees. Diagram (c) and (d) denote weight-sharing blocks with two in-degrees and two out-degrees. Differently, the two output feature maps of diagram (c) will pass through same blocks while the two output feature maps of diagram (d) will pass through different blocks. Here, in-degree and out-degree respectively indicate input feature map and output feature map of convolutional blocks.

ensemble training framework become strong with concise structure.

As shown in Eq.1, common blocks (diagram (a)) uses one input feature map to provide one output feature map. As shown in Eq.2, weight-sharing block in diagram (b) also uses one input feature maps. Different from common blocks, the one output feature map will be served as input feature map in more than one forward branches at next stage. Eq.3 denotes the forward process of weight-sharing blocks in diagram (c) and (d). In diagram (c), output feature maps will be fed into same next-stage block. Differently in diagram (d), output feature maps will be fed into different next-stage blocks.

$$\boldsymbol{F(i)} = f(\boldsymbol{w_i}; \boldsymbol{F(i-1)}) \tag{1}$$

$$\boldsymbol{F^k(i)} = f(\boldsymbol{w_i}; \boldsymbol{F(i-1)}) \tag{2}$$

$$\boldsymbol{F^k(i)} = f(\boldsymbol{w_i}; \boldsymbol{F^k(i-1)}), \quad k = 1, 2, \cdots, N_{out}^i \tag{3}$$

where $f(\cdot)$ denotes convolutional blocks. $F(i), F(i-1)$ denotes feature maps in i^{th} and $(i-1)^{th}$ stage. $F^k(i)$ represents the i^{th}-stage feature map existing in k^{th} forward branch. N_{out}^i indicates the number of out-degrees at i^{th} stage.

C. Online Feature Augmentation

In EKD-FWSNet, we design weight-sharing blocks to provide diverse predictions with less memory and computation increment. However, when the number of weight-sharing blocks increase, representation among forward branches will gradually get close, which harms the performance of ensemble teacher. On this issue, we propose online feature augmentation blocks (FA) to enrich the patterns of feature maps. In EKD-FWSNet, we mainly introduce squeeze-and-excitation (SE) block [25], channel-attention-module (CAM) [17] and Dropout block [26] as online FA blocks. Among them, SE block and CAM utilize channel-wise attention for feature augmentation. Dropout block provides feature augmentation by occupation with random rate.

D. Knowledge Distillation in EKD-FWSNet

Designing well-optimized knowledge distillation mechanism is crucial for optimizing baseline model. Distillation loss functions of previous "student-classmate" ensemble training frameworks [20], [21], [22], [23] will increase rapidly when more classmate branches get involvement. More distillation loss functions will make the training complex and confuse the optimization. To make optimization easier, we only design two distillation loss functions keep the loss function number unchanged when involving more forward branches.

1) Distillation on Class Probabilities: Followed Hinton et al. [44], we apply temperature-scaled softmax operation to generate the posterior class probability which is denoted as $p(\hat{y} = y_i|\boldsymbol{x})$ for input batch images $\boldsymbol{x}$. Eq.4 shows the formulation. We then formulate the final posterior class probability of main student forward branch as $\boldsymbol{p_{ms}(x)} = \{p_{ms}(\hat{y} = 1|\boldsymbol{x}), \cdots, p_{ms}(\hat{y} = M|\boldsymbol{x})\}$. Similarly, we define the class probability of k^{th} forward branch as $\boldsymbol{p^k_{cm}(x)}$.

$$p(\hat{y} = y_i|\boldsymbol{x}) = \frac{e^{(z_i/T)}}{\sum_{j=1}^{M} e^{(z_j/T)}} \tag{4}$$

where z denotes the logits, which is a vector output from fully-connected classification layer. M and T respectively denote the number of the categories and the temperature. Additionally, T is set to 3 in this paper.

If the total number of classmate forward branches is set as K, we first generate ensemble teacher posterior class probability $(\boldsymbol{p_{et}(x)})$ by averaging all posterior class probabilities of main student and classmate forwards branches. We formulate this process in Eq. 5 and Eq. 6. Then we adopt the common KL (Kullback-Leibler) divergence constraint as knowledge distillation loss functions to guide main student (Eq. 7). In addition. N in Eq. 7 indicates the number of mini-batch size.

$$p_{et}(\hat{y} = y_i|\boldsymbol{x}) = \frac{e^{(\frac{1}{K+1}\sum_{k=1}^{K+1} z_{ki})/T}}{\sum_{j=1}^{M} e^{(\frac{1}{K+1}\sum_{k=1}^{K+1} z_{kj})/T}} \tag{5}$$

$$\boldsymbol{p_{et}(x)} = \{p_{et}(\hat{y} = 1|\boldsymbol{x}), \ldots, p_{et}(\hat{y} = M|\boldsymbol{x})\} \tag{6}$$

$$\begin{aligned} KL_{ms} &= \frac{1}{N}\sum_{n=1}^{N} KL\big(\boldsymbol{p_{ms}(x_n)} \| \boldsymbol{p_{et}(x_n)}\big) \\ &= -\frac{1}{N}\sum_{n=1}^{N} \boldsymbol{p_{et}(x_n)} \log \frac{\boldsymbol{p_{ms}(x_n)}}{\boldsymbol{p_{et}(x_n)}} \end{aligned} \tag{7}$$

Trained through knowledge distillation on class probabilities, main student can "discuss" with classmates and "absorb" knowledge from each other. The ensemble teacher class probability is not used to guide classmates because we do not want classmates to learn in same learning schedule as main student, especially when more weight-sharing blocks get involved. Also, with the involvement of more branches, no extra loss functions will be involved. This concise design make the training process concise and efficient.

2) Ensemble Attention Distillation on Intermediate Feature Maps: To enhance the semantic-level guidance of main student, we adopt ensemble-attention knowledge distillation strategy. Fig. 5 shows that intermediate feature

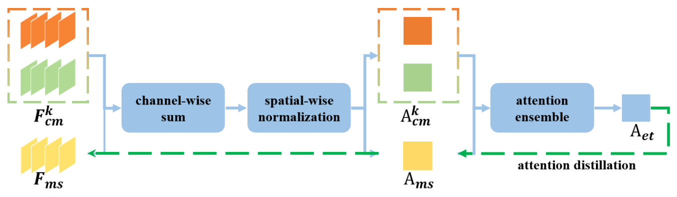

Fig. 5. The process of ensemble attention distillation. Druing forward process, attention maps of main student and classmates forward branches will be fused to form ensemble attention teacher. During backward, attention distillation loss function is used to transfer knowledge from ensemble teacher to main student.

map of main student will be guided by attention maps of ensemble teacher. Eq. 8 and Eq.9 represent the attention map generation process.

$$A_{ms} = \sum_{c=1}^{C} (F_{ms})_c, \quad A^k_{cm} = \sum_{c=1}^{C} (F^k_{cm})_c \tag{8}$$

$$A_{et} = \frac{1}{K+1}(norm(A_{ms}) + \sum_{k=1}^{K} norm(A^k_{cm})) \tag{9}$$

Here, we denote C as the number of feature map channels. we also denote spatial-wise normalization operation as $norm(\cdot)$. $F_{ms}, F^k_{cm} \in \mathbb{R}^{C\times H\times W}$ denote feature maps of main student and k^{th} classmate forward branch with same resolution. $A_{ms}, A^k_{cm} \in \mathbb{R}^{H\times W}$ denote attention maps, which are used to construct ensemble attention teacher (A_{et}). To transfer mature knowledge from A_{et} to A_{ms}, we apply MSE (mean-squared-error) loss shown in Eq. 10.

$$\begin{aligned} MSE_{ms} &= \frac{1}{N}\sum_{n=1}^{N} MSE\big(A_{ms}(x_n)\|A_{et}(x_n)\big) \\ &= \frac{1}{N}\sum_{n=1}^{N}\sum_{h=1}^{H}\sum_{w=1}^{W}\big((a_{ms}(x_n))_{hw} - (a_{et}(x_n))_{hw}\big)^2 \end{aligned} \tag{10}$$

where H and W are height and width of the attention map and $((a_{ms})(x_n))_{hw}$ denotes the pixel value at position (h, w) on attention map of n^{th} batch image.

E. Optimizing EKD-FWSNet

1) Combined Loss in EKD-FWSNet: To optimize baseline model in EKD-FWSNet, we first use cross-entropy loss (hard-label training) for each forward branch where L_{ms} and L^k_{cm} respectively make effect on main student and classmates forward branch. We then employ the two types of distillation loss functions (Eq. 7, Eq. 10). Finally, we combine all the loss functions for optimization. The combined loss (L) is defined in Eq. 11.

$$L = \underbrace{L_{ms} + \frac{1}{K}\sum_{k=1}^{K} L^k_{cm}}_{\text{cross-entropy}} + \underbrace{w \cdot KL_{ms}}_{\text{KL distillation}} + \underbrace{\alpha \cdot MSE_{ms}}_{\text{ensemble attention}} \tag{11}$$

In Eq. 11, w and α are two hyper-parameters for proportion adjustment of distillation and ensemble attention loss. In experiments of this paper, the values of w and α are respectively 60 and 1.0. Additionally, ensemble attention distillation is applied only at the neighboring layer before fully-connected layer. Forcing baseline model to approximate low-level features in early stage may obtain more redundant information or noise than useful knowledge.

2) Optimizing Weight-Sharing Blocks: Weight-sharing blocks exist in EKD-FWSNet in three types. As shown in diagram (b) of Fig.4, one input feature map is fed into weight-sharing block and same output feature map then pass through two forward branches. During backpropagation, the trainable parameters will be updated from two gradient terms, which is shown in Eq.12. As shown in diagram (c) and (d) of Fig.4, the weight-sharing blocks have two input feature maps and two output feature maps. Eq.13 shows the gradient terms.

$$\begin{cases} \dfrac{\partial L}{\partial w} = \displaystyle\sum_{k=1}^{N_{out}} \Big(\frac{\partial L}{\partial F^k_{out}} \cdot \frac{\partial F^k_{out}}{\partial w}\Big) \\ \dfrac{\partial L}{\partial F_{in}} = \displaystyle\sum_{k=1}^{N_{out}} \Big(\frac{\partial L}{\partial F^k_{out}} \cdot \frac{\partial F^k_{out}}{\partial F_{in}}\Big) \end{cases} \tag{12}$$

$$\begin{cases} \dfrac{\partial L}{\partial w} = \displaystyle\sum_{k=1}^{N_{out}} \Big(\frac{\partial L}{\partial F^k_{out}} \cdot \frac{\partial F^k_{out}}{\partial w}\Big) \\ \dfrac{\partial L}{\partial F^k_{in}} = \dfrac{\partial L}{\partial F^k_{out}} \cdot \dfrac{\partial F^k_{out}}{\partial F^k_{in}} \end{cases} \tag{13}$$

where N_{out} denotes the number of out-degrees. (E.g. In diagram (b) of Fig.4, $N_{out} = 2$). F^k_{out} represents the output feature map in k^{th} forward branch. F_{in} is the single input feature map. Weight-sharing blocks in diagram (c) and (d) have same gradient on trainable parameters (w) as weight-sharing blocks in diagram (b). The gradient on different input feature maps will be fed backward in different paths.

3) Optimizing Baseline Model in EKD-FWSNet: The proposed EKD-FWSNet is a self-learned efficient training framework which is designed to optimize baseline model in an end-to-end manner. During forward pass, we construct multiple predictions and generate ensemble teacher. This process is shown in Algorithm.1. During backpropogation, we use ensemble teacher to transfer knowledge to baseline model. This process is shown in Algorithm.2.

IV. Experiments and Analysis

A. Datasets and Implementation Details

In this paper, CIFAR [27] (CIFAR-10/100), tiny-ImageNet,[2] CUB-200 [28] and ImageNet [81] are four benchmark datasets for experiments. CIFAR consists of 50000 training images and 10000 testing images with small image scale (32 × 32). Tiny-ImageNet dataset has 200 categories. each category owns 500 training RGB images 50 testing RGB images. The resolution of all images are 64 × 64. CUB-200 contains 200 bird categories and total 11,788 images are split into 5994 images for training and 5794 images for testing. ImageNet consists 1000 categories, where 1.2 million 224 × 224 images are used for training and 50000 images for validation.

[2]https://tiny-imagenet.herokuapp.com

Algorithm 1 Forward Pass in EKD-FWSNet

```
Require: A minibatch of training samples $x = \{(x_1, y_1), \dots, (x_N, y_N)\}$. Define a EKD-FWSNet (Fig.2) where the number of classmate forward branches is set as $K$. Initialize trainable parameters $w = \{(w_1, \dots, w_l)\}$
Ensure: Posterior class probability of main student ($p_{ms}(x)$), classmate forward branches ($p^k_{cm}(x)$) and ensemble teacher ($p_{et}(x)$). Attention map of main student ($A_{ms}$), classmate forward branches ($A^k_{cm}$) and ensemble teacher ($A_{et}$).
  Forward Pass:
  $F(0) = x \in \mathbb{R}^{N \times C \times H \times W}$
  for $i = 1, \dots, l$ blocks do
    if Common blocks then
      Compute as diagram (a) of Fig.4: $F(i) = f(w_i; F(i-1))$
    else if Weight-sharing blocks then
      if Single input feature map then
        Compute as diagram (b) of Fig.4: $F(i) = f(w_i; F(i-1))$
      else
        Compute as diagram (c) and (d) of Fig.4: $F^k(i) = f(w_i; F^k(i-1)), k = 1, 2, \cdots, N^i_{out}$
      end if
    end if
  end for
  Get $p_{ms}(x)$, $p^k_{cm}(x)$ and $p_{et}(x)$ using Eq.4 $\sim$ Eq.6.
  Get $A_{ms}$, $A^k_{cm}$ and $A_{et}$ using Eq.8 and Eq.9.
```

Algorithm 2 Optimizing Baseline Model in EKD-FWSNet

```
Require: Define a EKD-FWSNet (Fig.2) where the number of classmate forward branches is set as $K$. Initialize trainable parameters $w = \{(w_1, \dots, w_l)\}$
Ensure: Updated trainable parameters $w' = \{(w'_1, \dots, w'_l)\}$
  Forward Pass:
  Get $p_{ms}(x)$, $p^k_{cm}(x)$, $p_{et}(x)$, $A_{ms}$, $A^k_{cm}$ and $A_{et}$ using Algorithm.1.
  Backpropogation:
  Clear gradients with zero-grad operation
  Get KL loss $KL_{ms}$ using Eq.7.
  Get ensemble attention loss $MSE_{ms}$ using Eq.10.
  Get combined loss $L$ using Eq.11.
  for $i = l, \dots, 1$ blocks do
    if Common blocks then
      Compute gradients with common backward operation
    else if Weight-sharing blocks then
      Compute gradients using Eq.12 and Eq.13
    end if
    Update trainable parameters $w_i$ with step operation
  end for
```

In this paper, ResNet [4] and EfficientNet [31] are selected as baseline models to construct EKD-FWSNet. The layer combination of baseline models are listed in Tab.I. Followed previous works, we slightly modify the "conv1" block of each baseline model to keep the resolution of output feature maps as same as input images. On CIFAR-10/100, we use Stochastic Gradient Descent (SGD) as optimizer in which the initial momentum value is 0.9 and weight decay value is 0.0001. In SGD optimizer, the learning rate is initialized as 0.1. During training, we set the batch size as 128. All baseline models in EKD-FWSNet will be trained with 300 epochs, where the learning rate will be divided 10 times at 150^{th} and 250^{th} epoch. On tiny-ImageNet, CUB-200 and ImageNet datasets, we adopt similar initial settings. Specifically, we follow the configuration on CIFAR and use same SGD optimizer. Differently, all models are trained with 100 epochs. The initial learning rate is 0.01 and will be decreased by 10 times at epoch 50 and 75 in training phase. In addition, For tiny-ImageNet, the input image size is 64 × 64, while for CUB-200 and ImageNet, the input images are resized to 224 × 224.

TABLE I

THE LAYER COMBINATIONS OF BASELINE MODELS [4], [31]. BASELINE MODELS ARE COMBINED WITH SEVERAL LAYERS IN DIFFERENT STAGES. E.G., RESNET-34 HAS FOUR LAYERS IN FOUR STAGES, WHERE EACH LAYER RESPECTIVELY HAS 3, 4, 6 AND 3 BASIC BLOCK UNITS. ADDITIONALLY, THE BLOCK UNIT INSIDE EACH LAYER OF RESNET AND EFFICIENTNET ARE RESPECTIVELY BASIC/BOTTLENECK BLOCK UNIT AND MBCONV BLOCK UNIT

Lightweight models	High-efficiency models
ResNet-20 [3, 3, 3]	ResNet-18/34 [2, 2, 2, 2]/[3, 4, 6, 3]
ResNet-32 [5, 5, 5]	EfficientNet-b0 [1, 2, 2, 3, 3, 4, 1]
ResNet-44 [7, 7, 7]	EfficientNet-b2 [2, 3, 3, 4, 4, 5, 2]
ResNet-56 [9, 9, 9]	EfficientNet-b4 [2, 4, 4, 6, 6, 8, 2]
Large-scale models	
ResNet-50 [3, 4, 6, 3]	ResNet-101 [3, 4, 23, 3]

B. Experimental Results

1) Classification on Lightweight Baseline Models: In this paper, lightweight models indicate models with simple structure (less complex operators) and less trainable parameters. Those models are mainly designed to satisfy devices with limited resource. Therefore, lightweight baseline models always have to sacrifice performance to lower down computation cost in inference phase. In lightweight baseline model based EKD-FWSNet, we construct a three-forward-branch structure by setting the first branch point after " layer1" (Fig.2). As for FA blocks, we apply Dropout blocks to compensate diversity loss.

To prove the effectiveness on improving lightweight models, we conduct experiments on notable lightweight model series, ResNet-20/32/44/56. It is worth noting that lightweight models are always designed for relatively simple classification tasks. Therefore, we only evaluate these models on small scale dataset, CIFAR-10/100. From the results on CIFAR-10/100 shown in Tab. II, we find that lightweight baseline models optimizing in EKD-FWSNet improve encouragingly. On the classification experiments of CIFAR-10, ResNet-20/32 respectively improve by 1.68% and 1.32% training in EKD-FWSNet. Larger lightweight baseline models (ResNet-44/56) also improve by large margin. On CIFAR-100, a more difficult dataset, EKD-FWSNet also performs excellent on classification experiments. ResNet-32 training in EKD-FWSNet

TABLE II

Top-1 Error Rate (%) of Lightweight Baseline Models on CIFAR-10/100. "BL" Means Individually Training Baseline Models. ⇂ Means the Margin of Error Rate Decrease. The Best Score Is Highlighted in Bold

Models	CIFAR-10		CIFAR-100	
	BL	EKD-FWSNet	BL	EKD-FWSNet
ResNet-20	8.37	**6.69** (⇂ 1.68%)	32.66	**28.54** (⇂ 4.12%)
ResNet-32	7.35	**6.03** (⇂ 1.32%)	30.73	**26.46** (⇂ 4.27%)
ResNet-44	6.78	**5.83** (⇂ 0.95%)	29.43	**25.74** (⇂ 3.69%)
ResNet-56	6.13	**5.49** (⇂ 0.64%)	28.91	**25.63** (⇂ 3.28%)

TABLE III

Top-1 Error Rate (%) of High-Efficiency Models on CIFAR-100 and Tiny-ImageNet. "Eb0/Eb2/Eb4" Are Three EfficientNets With Different Layer Combinations. The Best Score Is Highlighted in Bold

Models	CIFAR-100		tiny-ImageNet	
	BL	EKD-FWSNet	BL	EKD-FWSNet
ResNet-18	23.71	**20.49** (⇂ 3.22%)	30.91	**26.81** (⇂ 4.10%)
ResNet-34	22.16	**19.94** (⇂ 2.22%)	23.83	**21.74** (⇂ 2.09%)
Eb0	12.64	**11.33** (⇂ 1.31%)	18.91	**17.73** (⇂ 1.18%)
Eb2	11.34	**10.21** (⇂ 1.13%)	16.41	**15.02** (⇂ 1.39%)
Eb4	10.03	**8.90** (⇂ 1.13%)	13.78	**12.59** (⇂ 1.19%)

TABLE IV

Top-1 Error Rate (%) of High-Efficiency Models on Large-Scale Datasets, CUB-200 and ImageNet With Single-Crop Testing. The Best Score Is Highlighted in Bold

Models	CUB-200		ImageNet	
	BL	EKD-FWSNet	BL	EKD-FWSNet
ResNet-18	34.67	**26.24** (⇂ 8.43%)	30.32	**27.64** (⇂ 2.78%)
ResNet-34	29.37	**24.09** (⇂ 5.28%)	26.58	**23.91** (⇂ 2.67%)
Eb0	24.17	**21.58** (⇂ 2.59%)	23.31	**22.04** (⇂ 1.27%)
Eb2	21.38	**19.47** (⇂ 1.91%)	20.47	**19.45** (⇂ 1.02%)
Eb4	19.75	**18.12** (⇂ 1.63%)	18.21	**17.33** (⇂ 0.88%)

TABLE V

Top-1 Error Rate (%) of Large-Scale Models on Large-Scale Datasets, CUB-200 and ImageNet With Single-Crop Testing. The Best Score Is Highlighted in Bold

Models	CIFAR-100		tiny-ImageNet	
	BL	EKD-FWSNet	BL	EKD-FWSNet
ResNet-50	18.63	**16.16** (⇂ 2.47%)	20.98	**19.57** (⇂ 1.41%)
ResNet-101	15.76	**14.16** (⇂ 1.60%)	19.26	**17.99** (⇂ 1.27%)
Models	CUB-200		ImageNet	
	BL	EKD-FWSNet	BL	EKD-FWSNet
ResNet-50	25.42	**23.00** (⇂ 2.42%)	23.94	**22.79** (⇂ 1.15%)
ResNet-101	22.29	**20.73** (⇂ 1.56%)	22.40	**21.42** (⇂ 0.98%)

improve by 4.27% (error rate: 30.73% ∼ 26.46%) over individually training ResNet-32. Generally, compared to individually training baseline models, the average accuracy improvement of ResNet series optimizing in EKD-FWSNet is 3.89%.

2) Classification on High-Efficiency Baseline Models: In this paper, high-efficiency models indicate some recent compact models, which always utilize complex modules or operations to achieve high performance with fewer cost. Compared to lightweight models, high-efficiency models are more compact with high-efficiency design, so further improve their generalization capacity becomes harder. Here, we select to construct four-forward-branch structure. For ResNet-18/34, the first branch point is set after "layer1" and two more branch points are respectively set after "layer2" and "layer3". For EfficientNet series, the first branch point is set after "layer1" and two more branch points are respectively set after "layer3" and "layer5".

For high-efficiency baseline models, evaluating on simple classification tasks is not convincing, because they can all achieve very high results trained individually. Therefore, for high-efficiency baseline models, we conduct experiments on CIFAR-100, tiny-ImageNet, CUB-200 and ImageNet. The experiments in Tab.III and Tab.IV prove that optimizing with EKD-FWSNet, high-efficiency models can also get improved. From Tab.III, we find that ResNet-18 and ResNet-34 optimizing in EKD-FWSNet make huge progress. Especially on tiny-ImageNet, ResNet-18 in EKD-FWSNet surpasses individually training baseline model by 4.10% (30.91%∼26.81%). Experiments on EfficientNet further guarantee the efficiency of EKD-FWSNet. Even though Eb0/Eb2/Eb4 have already performed very competitive, they can still be improved by EKD-FWSNet. The average improvement is more than 1%. From Tab.IV, we observe that high-efficiency baseline models training in EKD-FWSNet can outperform individually training models both on CUB-200 and ImageNet. Particularly, ResNet-18 training in EKD-FWSNet achieves 8.43% improvement. Moreover, EKD-FWSNet shows much strong power on CUB-200 (Baseline models improve by large margin). The reason is that the multi-branch structure of EKD-FWSNet can possibly bring in diverse visual attentions [77], which can boost the model's discriminative ability on fine-grained classification task.

3) Classification on Large-Scale Baseline Models: In this paper, large-scale baseline models indicate wide or deep models with large number of trainable parameters and high computation. These models always gain high performance improvement depending on high computation and memory cost. In large-scale baseline model based EKD-FWSNet, we construct a four-forward-branch structure by setting the branch point after "layer1", "layer2" and "layer3", which follows the configurations of high-efficiency baseline model based EKD-FWSNet.

In this paper, we select ResNet-50 and ResNet-101 as large-scale baseline models (Tab.I) and conduct classification experiments on large-scale dataset CUB-200 and ImageNet. As shown in Tab.V, ResNet-50/101 training in EKD-FWSNet perform better than individually training. Particularly, the improved results on CUB-200 and ImageNet further show that EKD-FWSNet has strong generalization ability on large-scale benchmark dataset. Even though large-scale models have

TABLE VI

Top-1 Error Rate (%) of Lightweight Baseline Model Comparison on CIFAR-10/100. All Results Are "Average Value ± Standard Deviation" of Three Runs. In Addition, "SD" and "OEM" Respectively Indicate Method of [59], and [20]. The Best Score Is Highlighted in Bold

Models	CIFAR-10			CIFAR-100		
	SD	OEM	EKD-FWSNet	SD	OEM	EKD-FWSNet
ResNet-20	6.89	7.87	**6.69±0.10**	30.05	-	**28.54±0.08**
ResNet-32	6.10	7.05	**6.03±0.08**	28.22	29.03	**26.46±0.14**
ResNet-44	-	6.55	**5.83±0.05**	-	28.24	**25.74±0.13**
ResNet-56	6.02	-	**5.49±0.05**	26.78	27.84	**25.63±0.17**

already achieved high-performance, they still have potential to become better. Our EKD-FWSNet can explore this potential with self-learned mechanism.

4) Classification Comparison With Knowledge Distillation Guided Training Framework: Compared to individually training, baseline models training in EKD-FWSNet show encouraging improvement. To further prove the effectiveness of EKD-FWSNet, we conduct comparison experiments with recent notable knowledge distillation guided training frameworks [20], [22], [23], [24], [51], [59], [60], [61], [63]. Specifically, KD-ONE [22], OEM [20], DCCL [24], AFID [65] and PCL [66] are recent notable methods which combines knowledge distillation strategy with ensemble learning. All these three methods are student-classmate training framework (Fig.1 **top-right**).DML [23], FRSKD [60], MixSKD [61] and SD [59] are recent outstanding methods which enhance the optimization of baseline models by constructing multi-branch self-distillation guided training frameworks. These methods all exploits the data representation invariance by knowledge transferring between main student and classmate branches. KESI (Knowledge Distillation from Ensembles of Snapshots of Iterative Pruning) [63] constructs ensemble teacher from snapshots of iterative pruning and uses ensemble distillation mechanism to improve the representation power of compressed models.

a) Lightweight baseline models comparison: We compare EKD-FWSNet with previous excellent methods on lightweight models for CIFAR-10/100 classification task. In Tab.VI, we compare with SD [59] and OEM [20] on ResNet-20/32/44/56. Obviously, baseline models optimizing in EKD-FWSNet perform better. Especially on CIFAR-100, ResNet-20/32/44/56 can all surpass [59] and [20] by more than 1%. Since ResNet-32 is a typical lightweight baseline model, which is widely selected for many previous advanced methods, we further make comparison experiments on ResNet-32. Results shown in Tab.VII show that EKD-FWSNet has little inferiority compared to AFID [65] and PCL [66].

b) High-efficiency baseline models comparison: We compare EKD-FWSNet with previous excellent methods on high-efficiency baseline models for CIFAR100, tiny-ImageNet, CUB-200 and ImageNet classification. (1) On CIFAR-100, we compare EKD-FWSNet with OEM [20] on EfficientNet-b0/b2/b4 and compare EKD-FWSNet with SD [59] on ResNet-18/34. The results shown in Fig.6 clearly shows that high-efficiency baseline models achieve lower error rate optimizing with EKD-FWSNet. (2) On CIFAR-100 and CUB-200, ResNet-18 is a common selected typical high-efficiency baseline model. Here, we make more comparison experiments (Tab.VIII) on ResNet-18. For CIFAR-100 classification task, EKD-FWSNet achieves comparable results with previous SOTA method [61]. For CUB-200 classification task, EKD-FWSNet shows strong power and achieves new SOTA results, which further proves that EKD-FWSNet has strong power on fine-grained image classification task. Inspired by MixSKD [61], we also apply Mixup into EKD-FWSNet.

TABLE VII

Top-1 Error Rate (%) Comparison on ResNet-32 for CIFAR-10/100 Classification. All Results Are "Average Value ± Standard Deviation" of Three Runs. The Best Score Is Highlighted in Bold

Methods	CIFAR-10	CIFAR-100
OEM [20]	7.05	29.03
DML [23]	6.68	28.90
SD [59]	6.10	28.22
KD-ONE [22]	5.99	26.61
DCCL [24]	-	26.57
AFID [65]	-	26.00
PCL [66]	**5.67**	**25.86**
EKD-FWSNet	6.03±0.08	26.46±0.14

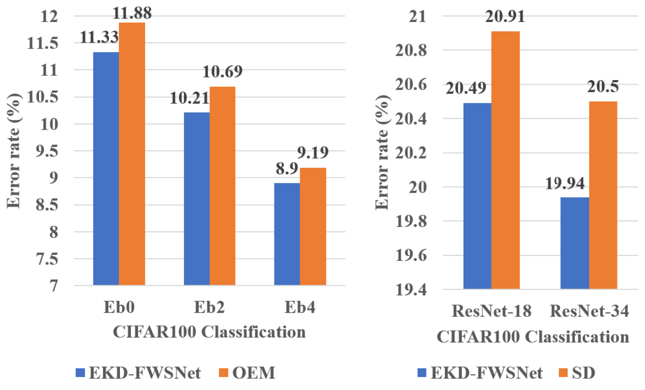

Fig. 6. Top-1 error rate (%) high-efficiency baseline model comparison on CIFAR-100. All results are average values of three runs. In addition, "SD", "OEM" and "KESI" respectively indicate method of [59], [20] and [63].

TABLE VIII

Top-1 Error Rate (%) Comparison on ResNet-18 for CIFAR-100 and CUB-200 Classification. All Results Are "Average Value ± Standard Deviation" of Three Runs. The Best Score Is Highlighted in Bold

Methods	CIFAR-100	CUB-200
CS-KD [51]	20.40±0.31	30.71±0.64
FRSKD [60]	21.26±0.19	32.02±0.58
MixSKD [61]	**19.68±0.13**	27.85±0.53
EKD-FWSNet	20.49±0.18	**26.24±0.41**
EKD-FWSNet+Mixup	19.81±0.37	**25.37±0.49**

TABLE IX

TOP-1 ERROR RATE (%) COMPARISON ON RESNET-18/34 FOR TINY-IMAGENET AND IMAGENET CLASSIFICATION. "†" MEANS LOADING IMAGENET-PRETRAINED WEIGHTS FOR TINY-IMAGENET CLASSIFICATION. THE BEST SCORE IS HIGHLIGHTED IN BOLD

Methods	tiny-ImageNet		ImageNet	
	ResNet-18	ResNet-34	ResNet-18	ResNet-34
DML [23]	32.68	30.86	30.15	25.99
EKD [49]	30.54	30.22	29.03	25.24
DCCL [24]	**30.41**	30.08	28.00	24.12
AFID [65]	-	-	28.97	24.07
PCL [24]	-	-	29.58	-
KESI† [63]	30.70	28.69	-	-
EKD-FWSNet	31.32	**28.43**	**27.64**	**23.91**
EKD-FWSNet†	26.81	21.74	**27.64**	**23.91**

Here, we simply serve Mixup as a data augmentation technique, so it can smoothly be applied in our architecture. The results show that Mixup contributes on improving EKD-FWSNet. (3) On tiny-ImageNet and ImageNet, previous methods widely make comparison using ResNet-18 and ResNet-34 as baseline models. Followed their experimental setting, we evaluate the performance of ResNet-18/34 optimizing in EKD-FWSNet. The results are shown in Tab.IX. For tiny-ImageNet classification task, EKD-FWSNet outperforms previous methods on optimizing ResNet-34 and achieves comparable performance on optimizing ResNet-18. For ImageNet classification task, EKD-FWSNet surpasses previous methods (including AFID [65] and PCL [66]) on both ResNet-18 and ResNet-34. Moreover, to fairly compare with KESI [63], we load ImageNet-pretrained weights on ResNet-18/34 for tiny-ImageNet classification and naturally obtain more competitive results.

c) Large-scale baseline models comparison: Most previous related methods focus on optimizing lightweight models of high-efficiency models, so not many methods select large-scale models as baseline models. In this paper, we mainly compare with OEM [20], SD [59] on ResNet-50/101 for ImageNet classification. The comparison is shown in Fig.7. On ImageNet dataset, ResNet-50 and ResNet-101 both have lowest error rate. We further compare the improvement margin (error rate decrease) with OEM and SD. The error rate decrease curve reveals that EKD-FWSNet can improve large-scale baseline models by largest margin.

d) Ensemble teacher comparison: Our proposed method integrates ensemble learning strategy into knowledge distillation training framework. Naturally, if the performance of ensemble teacher is better, the main student is easier to become better. In Tab.X, we compare the performance of ensemble teacher with other ensemble knowledge distillation guided methods [20], [22], [24], [65] to reveal the all-round superiority of EKD-FWSNet. On CIFAR-10, EKD-FWSNet achieves comparable results with KD-ONE, where the error rate is a little (0.04%) higher. On CIFAR-100, compared to AFID, EKD-FWSNet has little inferiority on classification result of

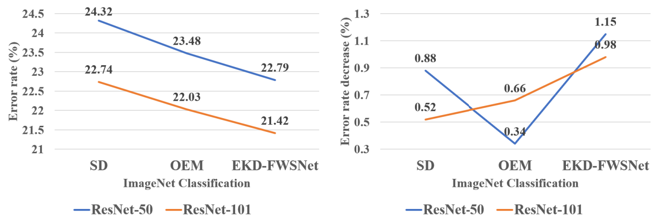

Fig. 7. Top-1 error rate and error rate decrease (%) of large-scale baseline model comparison on ImageNet. All results are average values of three runs. In addition, "SD" and "OEM" respectively indicate method of [59] and [20].

TABLE X

COMPARISON OF ENSEMBLE TEACHER PERFORMANCE USING ERROR RATE (%). WE SELECT RESNET-32 AS BASELINE MODEL AND RESPECTIVELY COMPARE THE "MAIN STUDENT"(RESNET-32: BASELINE MODEL AFTER TEACHING) AND "ENSEMBLE TEACHER"(RESNET-32-E: ENSEMBLE TEACHER) RESULTS WITH KD-ONE [22], DCCL [24], OEM [20] AND AFID [65]. THE BEST SCORE IS HIGHLIGHTED IN BOLD

Methods	CIFAR-10		CIFAR-100	
	ResNet-32	ResNet-32-E	ResNet-32	ResNet-32-E
OEM [20]	7.05	5.73	29.03	26.06
DCCL [24]	-	-	26.57	24.62
AFID [65]	-	-	**26.00**	**24.07**
KD-ONE [22]	**5.99**	-	26.61	24.63
EKD-FWSNet	6.03±0.08	**5.14±0.07**	26.46±0.14	24.15±0.09

ResNet-32. However, the performance of ensemble teacher between EKD-FWSNet and AFID is comparable. From Tab.X, we find that EKD-FWSNet has high-quality ensemble teacher, which guarantees the improvement of baseline model (Better teacher can teach better student).

e) Discussion and Analysis with AFID and PCL: AFID [65] and PCL [66] have close relation with our proposed EKD-FWSNet. These three methods both introduce ensemble teacher to facilitate the learning of baseline models in "logits-level" and "intermediate feature-level". The main difference between AFID and EKD-FWSNet is whether uses weight-sharing blocks to construct the ensemble training network. Specifically, AFID directly employs one more complete sub-network to construct a two-branch ensemble network. EKD-FWSNet only uses one auxiliary classmate branch formed with several layers to construct a multiple-branch (more than two-branch) ensemble network. Compared to AFID, EKD-FWSNet utilizes weight-sharing blocks and obviously has less training memory and computation cost.

The differences between PCL and EKD-FWSNet can be listed as follows. (1) PCL constructs multiple forward branches by serving low-level layers as weight-sharing blocks and directly increasing high-level layers. EKD-FWSNet is only aided by one auxiliary classmate branch with several layers. When more forward branches are required, the training memory of PCL will multiply while EKD-FWSNet will have no extra memory cost. (2) Compared to EKD-FWSNet, PCL adds peer collaborative distillation from a peer mean teacher (a temporal mean model updated by Exponential Moving

TABLE XI

TOP-1 ERROR RATE (%) COMPARISON ON CIFAR-100. ABLATION EXPERIMENTS TO SEPARATELY SHOW THE EFFECTIVENESS OF EACH KEY LOSS TERM ON EKD-FWSNET. "+" MEANS APPENDING. "BL" MEANS INDIVIDUALLY TRAINED BASELINE MODEL. THE BEST SCORE IS HIGHLIGHTED IN BOLD

Models	CIFAR-100			
	BL	+L_{ms}, L_{cm}^{k}	+KL_{ms}	+MSE_{ms} (EKD-FWSNet)
ResNet-20	32.66	32.57	29.39	**28.54**
ResNet-32	30.73	30.86	26.91	**26.46**
ResNet-44	29.43	29.62	26.03	**25.74**
ResNet-56	28.91	29.17	25.78	**25.63**
ResNet-18	23.71	23.64	21.09	**20.49**
ResNet-34	22.16	22.03	20.22	**19.94**

TABLE XII

EFFECTIVENESS OF DIFFERENT FEATURE AUGMENTATION BLOCKS ON EKD-FWSNET. WE SET FIRST BRANCH POINT AFTER "LAYER1". THEREFORE, THE BRANCH NUMBER OF LIGHTWEIGHT BASELINE MODELS AND HIGH-EFFICIENCY BASELINE MODELS ARE RESPECTIVELY 3 AND 4. ALL EXPERIMENTS ARE EVALUATED ON CIAFR-100 WITH ERROR-RATE (%). THE BEST SCORE IS HIGHLIGHTED IN BOLD

Models	Dropout	SE	CAM	w/o FA	Branch Number
ResNet-20	**28.54**	28.76	29.20	31.83	3
ResNet-32	**26.46**	26.69	27.07	28.65	3
ResNet-44	25.74	**25.69**	26.03	27.92	3
ResNet-56	**25.63**	25.93	26.21	27.44	3
ResNet-18	**20.49**	20.95	21.59	22.84	4
ResNet-34	**19.94**	20.55	20.92	21.37	4

Average) to student model. This design improves the performance of baseline model, but it further introduces training memory and computation cost. (3) Both these two methods use augmentation techniques to increase the diversity of multiple predictions from different forward branches. However, PCL and EKD-FWSNet respectively apply augmentation in data-level and feature-level.

From Tab.VII, Tab.IX and Tab.X, we intuitively observe that EKD-FWSNet performs much stronger on high-efficiency baseline models (E.g., ResNet-18/34) optimization while AFID and PCL show better performance on lightweight baseline models (E.g., ResNet-32) optimization. Generally speaking, EKD-FWSNet shows comparable optimization performance compared to AFID and PCL. As for training cost (training memory and computation cost), EKD-FWSNet shows obvious superiority over AFID and PCL.

C. Ablation Study

1) Effectiveness of Each Key Loss Term of EKD-FWSNet: To separately prove the effectiveness of each key loss terms of EKD-FWSNet, we conduct experiments shown in Tab. XI. When only applying cross-entropy loss (L_{ms}, L_{cm}^{k}), the performance baseline model will not improve. This is because optimizing with auxiliary classmate branch using only hard-labels cannot provide mature knowledge to baseline model. When further adding KL distillation loss on posterior class probabilities, the improvement is obvious. The reason is that the ensemble teacher can provide easy-achieved mature global knowledge for main student to mimic. When using ensemble attention distillation on intermediate feature maps, the accuracy of baseline models can get improved by average 0.41%. Although sometimes baseline models improve little, it still works in most cases. Theoretically, [59] and [20] adopt knowledge distillation strategy in intermediate feature maps by directly using distillation loss functions among feature maps, which are 3-dimension tensors. i.e., they construct tensor-level high-dimension approximation, which will introduce extra optimizing burden (Curse of Dimensionality). Moreover, distillation loss functions increase with the increase of forward branches. In this paper, EKD-FWSNet avoids the above problems. No matter how many forward branches are constructed in architecture, we only apply matrix-level ensemble attention distillation on 2-dimension attention maps with only one loss functions, which can ease the optimization of baseline models.

2) Effectiveness of Different Feature Augmentation Blocks: In this paper, we introduce online FA blocks to compensate the diversity diffusion of each forward branch's prediction caused by deeper weight-sharing blocks. Here, we select ResNet series as baseline models and conduct classification experiments on CIFAR-100 to analyze FA blocks. Results in Tab.XII shows that Dropout block performs better than SE and CAM in most cases. If we do not apply FA blocks, the performance of baseline models will largely decrease because of diversity loss.

3) Layer-Wise Versus Inner-Block-Wise Branch Point Setting: Recent notable baseline models always consist of several layers. In most cases, output feature maps from different layers have different resolution. In baseline models, layer indicates convolutional blocks containing several units, e.g. Layer1 of ResNet-34 has three basic block units and layer2 of EfficientNet-b0 has two MBConv block units (Tab.I). In experiments, we mainly adopt layer-wise branch point setting to construct EKD-FWSNet, which is shown in Fig.2 and diagram (a) of Fig.3. However, only setting branch points between layers will limit the growing of forward branches. To flexibly construct EKD-FWSNet, we apply inner-block-wise branch point setting to construct more forward branches by splitting layers on both main student branches and classmate branch and set more branch points (diagram (b) of Fig.3).

To compare the layer-wise and inner-block-wise setting, we first select ResNet-18/34 as baseline models and construct a six-forward-branch EKD-FWSNet (Fig.8). Then, we run the classification experiments on CIFAR-100. Through the comparison shown in Tab. XIII, we find the following three points. (1) Six-forward-branch EKD-FWSNet has little superiority compared to four-forward-branch EKD-FWSNet, which means the optimizing profit can not continuously grow. (2) Baseline models get improved mainly because they absorb the sub-optimal generalized knowledge of "ensemble teacher", which is not an infinite knowledge. (3) We can not provide an

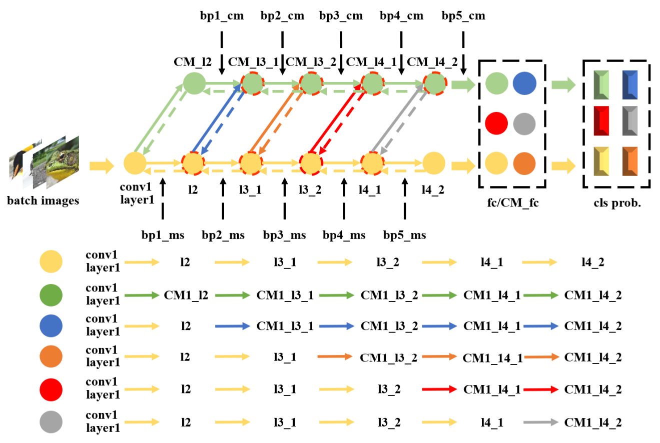

Fig. 8. An architecture of six-forward-branch EKD-FWSNet. On main branch and classmate branch of ResNet18/34, we both set five branch points, where "bp3_ms/bp3_cm" and "bp5_ms/bp5_cm" are respectively set inside "layer3/CM_layer3" and "layer4/CM_layer4" of main branch and classmate branch. The routes of six forward branches are shown in this figure.

TABLE XIII

Comparison Results Between Four-Forward-Branch Layer-Wise EKD-FWSNet and Six-Forward-Branch Inner-Block-Wise EKD-FWSNet. We Use ResNet-18/34 as Baseline Model and Evaluate on CIFAR-100

Models		Branch Number	Err-rate (%)
ResNet-18@layer1	EKD-FWSNet	4	20.49
ResNet-18@layer1	EKD-FWSNet	6	20.63
ResNet-34@layer1	EKD-FWSNet	4	19.94
ResNet-34@layer1	EKD-FWSNet	6	19.91

optimal and fix branch number for every classification tasks, so we need a flexible setting. The inner-block-wise branch point setting can make EKD-FWSNet more flexible when more forward branches are required. In summary, compared to previous works [19], [20], EKD-FWSNet can flexibly extend to an end-to-end ensemble network with lager number of forward branches without depending on more trainable parameters.

4) One Auxiliary Classmate Branch Versus More: As shown in Fig.8, EKD-FWSNet only has one auxiliary classmate branch and one main student, which means we can construct multi-forward-branch network without depending on multi-classmate branches. This design decreases the training computation and memory cost.

Previous methods [20], [22], [66] utilize multiple classmate branches to construct ensemble networks. Here, we also design a six-forward-branch EKD-FWSNet with two auxiliary classmate branches shown in Fig.9. We will compare this structure with "one-auxiliary-classmate" EKD-FWSNet (Fig.8) in the following points. (1) Compared to EKD-FWSNet shown in Fig.8, this architecture obviously has more memory cost. (2) With one more auxiliary classmate branch involved, less inner-block-wise branch points will be involved. Compared to "one-auxiliary-classmate" version, this architecture constructs same number of forward-branches without inserting any inner-block-wise branch points. (3) When adding one more auxiliary classmate branch, the sharing rate of deeper layers decrease. E.g., In this architecture, "CM1_l4_2" exists in 2 forward branches. In "one-auxiliary-classmate" EKD-FWSNet, "°CM1_l4_2" exists in 5 forward branches.

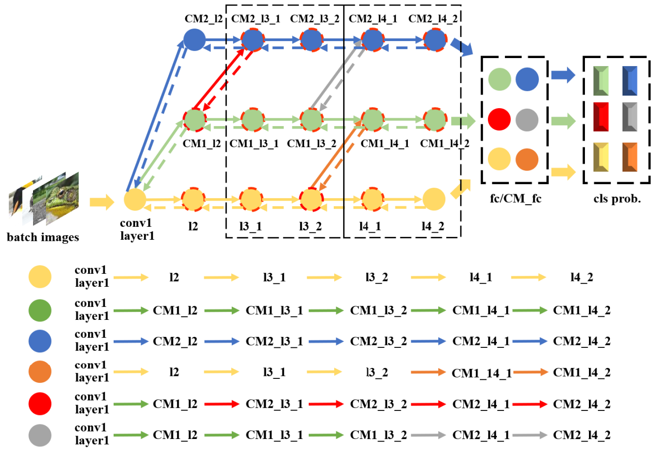

Fig. 9. An architecture of six-forward-branch EKD-FWSNet with two auxiliary branches.

TABLE XIV

Comparison Results Between Six-Forward-Branch "One-Auxiliary-Classmate" EKD-FWSNet and "Two-Auxiliary-Classmate" EKD-FWSNet. We Use ResNet-18/34 as Baseline Model and Evaluate on CIFAR-100

Models		Auxiliary Classmate Branch Number	Err-rate (%)
ResNet-18@layer1	EKD-FWSNet	1	20.63
ResNet-18@layer1	EKD-FWSNet	2	20.25
ResNet-34@layer1	EKD-FWSNet	1	19.91
ResNet-34@layer1	EKD-FWSNet	2	19.64

From the comparison results of Tab.XIV, we observe that "two-auxiliary-classmate" EKD-FWSNet gains minor improvement with more memory and computation cost. Essentially, high overlapping rate among forward branches will harm the diversity and weaken the ensemble teacher. Essentially, multiple auxiliary classmate branches make trade-off between diversity improvement and computation resource costing.

D. Visualization and Analysis

1) Hyper-Parameter Adjusting Principle: In this paper, we propose simple yet efficient training framework. Only two scalars are served as hyper-parameters, where w and α are respectively applied to adjust the proportions of *KL* distillation loss and *MSE* ensemble-attention loss. In different cases, we adopt different hyper-parameter adjusting strategies to make training framework optimize well. Tab. XV shows the configuration of these two scalars in different cases. The principle of adjusting the hyper-parameter is listed as follows. (1) We select large value of w, because we find the KL loss is far less than *MSE* and cross-entropy loss. This principle is also a experimental hyper-parameter adjusting experience. (2) We tend to use higher value of w for more complex task (CIFAR-100, tiny-ImageNet). Because the pattern of class

TABLE XV

The Configuration of Hyper-Parameter on Different Datasets. We Denote the Two Hyper-Parameter in Format of (w, α)

Datasets	Models		
	Lightweight	High-efficiency	Large-scale
CIFAR-10	(15, 1)	-	-
CIFAR-100	(60, 1)	(100, 1)	(100, 1)
tiny-ImageNet	-	(100, 1)	(100, 1)
CUB-200	-	(100, 1)	(100, 1)
ImageNet	-	(100, 1)	(100, 1)

TABLE XVI

The Evaluation on Hyper-Parameter (w, α) Sensitivity. Here, We Select ResNet-32 (Lightweight), ResNet-18 (High-Efficiency) and ResNet-50 (Large-Scale) and Evaluate on CIFAR-100 Dataset With Error Rate (%). "BL" Means Individually Trained Baseline Model

Models	BL	Combinations of (w, α)					
ResNet-32	30.73	(60, 1)	(120, 1)	(30, 1)	(10, 1)	(60, 2.5)	(60, 0.4)
		26.46	29.05	27.44	30.11	29.76	27.15
ResNet-18	23.71	(100, 1)	(200, 1)	(50, 1)	(20, 1)	(100, 2.5)	(100, 0.4)
		20.49	23.67	21.33	23.05	22.87	20.83
ResNet-50	18.63	(100, 1)	(200, 1)	(50, 1)	(20, 1)	(100, 2.5)	(100, 0.4)
		16.16	17.64	16.34	17.69	18.77	16.75

probabilities is more complex when the number of categories is large. (3) We set smaller α when applying *MSE* loss on large-scale feature maps to avoid abnormal values caused by large loss. (4) When more branches are added, we decrease the factor of cross-entropy loss to avoid exploding gradients. Fortunately, it rarely happens in our proposed models. (5) For ResNet [4] series and EfficientNet [31] series, we use a fix set of hyper-parameter shown in Tab.XV, which shows the robustness and less training sensitivity of EKD-FWSNet.

Followed the above-mentioned principles, we conduct an experiment to test the sensitivity of hyper-parameters (w, α). As shown in Tab.XVI, we will make the following analysis. (1) The ratio of KL loss term (w) cannot be too high or too low, because learning from hard-labels and soft-labels in a balanced manner can maximize the optimization effect. (2) Similarly, the ratio of *MSE* loss term (α) also needs to be set properly. Particularly when α is much larger, the harm will rapidly increase. This result also verifies our proposed third principle. (3) As shown in Tab.XI, KL loss term contributes more than *MSE* loss term. Therefore, when using much lower w, classification results will become close to baseline results.

2) Comparison and Analysis of Computation Cost in Training Phase: As Mentioned in main paper, some recent typical works [20], [22], [23], [24] show the power of "student-classmate" paradigm (Fig.1 **top-right**). However, with the increase of classmate branches, the training burden increase rapidly, which will complicate training process. Comparing with those methods, our proposed method is more compact and flexible. To intuitively show the computation cost in training phase, we compare the training FLOPs in Tab. XVII.

TABLE XVII

Comparison of Computation Cost in Training Phase. Parameters and FLOPs Are Calculated During Training Forward Process With Standard Input Size 224×224. We Use ResNet-32 as Baseline Model and Compare the Classification Performance on CIFAR-100. Additionally, We Reimplement the Inference Phase of OEM, DML, DCCL and KD-ONE to Calculate the Computation Cost. The Best Score Is Highlighted in Bold

Methods	Params	FLOPs	Branch Number	Err-rate (%)
ResNet-32	0.46M	3.40G	1	30.73
OEM [20]	0.95M	7.02G	5	29.03
DML [23]	0.93M	6.81G	2	28.90
DCCL [24]	0.92M	6.80G	2	26.57
KD-ONE [22]	1.17M	5.61G	3	26.61
KD-ONE [22]	2.22M	8.92G	6	-
EKD-FWSNet	**0.90M**	6.71G	3	**26.46**
EKD-FWSNet	**0.90M**	9.44G	6	**26.22**

To fairly compare the computation cost of EKD-FWSNet with previous methods, we first calculate the training parameters and FLOPs during training phase of individual baseline model. Then we carefully reimplement the inference phase of previous models [20], [22], [23] and calculate their training cost. As shown in Tab. XVII, DML [23] and DCCL [24] uses a two-branch (two ResNet-32) structure. Therefore, the training parameters and the training FLOPs are approximately the two times of baseline model. If more branches are involved, parameters and FLOPs will multiply. OEM [20] combines several branches with different size. However, each branch is fed with high-resolution feature maps, which also costs a lot both on parameters and FLOPs. KD-ONE [22] first utilizes low-level layers as weight-sharing blocks and then constructs multiple branches using separate high-level layers. Comparing with DML and OEM, KD-ONE largely lowers the training FLOPs.

Compared to DML, DCCL and OEM, EKD-FWSNet can train a baseline model better with less training parameters and FLOPs. Compared to three-branch KD-ONE, EKD-FWSNet has less parameters but more FLOPs. Then, we increase forward branches and make further comparison. When adding more branches, KD-ONE will use more high-level layers. Since high-level layers contain large group of parameters, the training parameters and FLOPs of six-branch KD-ONE rapidly increase, especially training parameters. To construct a six-branch EKD-FWSNet, we adopt block-wise branch point setting strategy (Fig. 8). From Tab.XVII, it is clear that EKD-FWSNet has no training parameters increase. From three-branch to six-branch structure, EKD-FWSNet has less training FLOPs increase (2.73G vs 3.31G).

In summary, one superiority of EKD-FWSNet is low training parameter cost. Another superiority of EKD-FWSNet is flexibility. when more branches are constructed, no extra

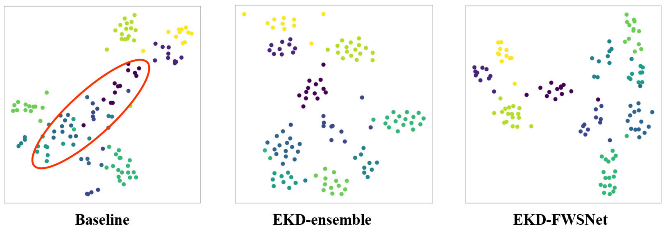

Fig. 10. Visualization results of logits on CIFAR-10 with T-SNE. We select ResNet-20 as baseline model. “EKD-ensemble” indicates the ensemble teacher logits of EKD-FWSNet. Different colors indicate different categories. here, we select CIFAR-10 dataset for visualization, so there is 10 different colors.

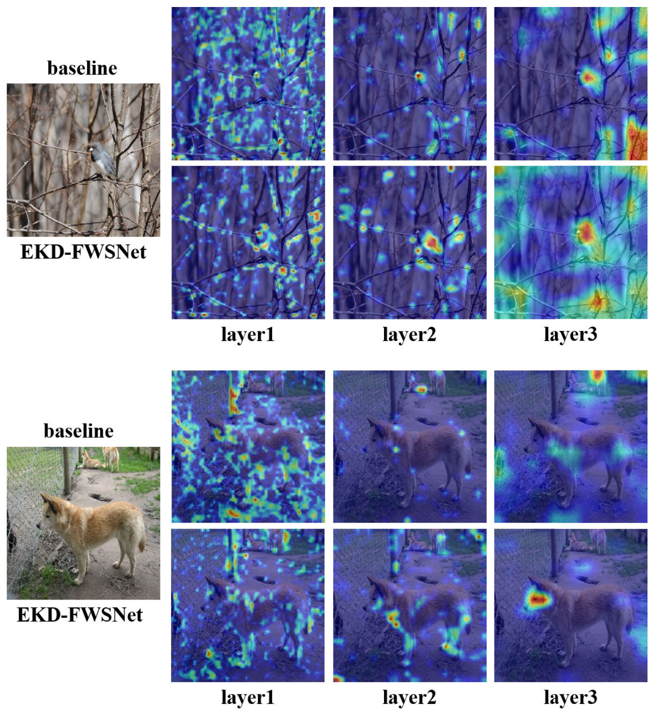

Fig. 11. The attention map visualization uses Grad-CAM method. We use ResNet-18 as baseline model and select images from ImageNet validation dataset.

training parameters are involved and the computation cost increases in a slow way.

3) Class Probability Visualization With T-SNE: To show the cluster pattern of predictions, we apply T-SNE [78] on the final logits of EKD-FWSNet. In Fig. 10, we compare the performance through the scatter plots generated by baseline model training individually and training in EKD-FWSNet. In scatter plots of EKD-FWSNet, the cluster of each class is tighter (smaller intra-class distance) and distance between clusters are larger (larger inter-class distance). Specifically, on the scatter plot of baseline model, it is clear that different points of some categories mix together (region with red mark). By contrast, the scatter plot of EKD-FWSNet and ensemble model are better. When comparing EKD-FWSNet with ensemble teacher, there is no significant difference in the T-SNE. The reason is that EKD-FWSNet directly gets knowledge from ensemble teacher, which can be regarded as a mimicking process. As a whole, Fig. 10 proves the interpretability of EKD-FWSNet.

4) Attention Region Visualization With Grad-CAM: To show the performance on feature representation of our proposed EKD-FWSNet, we adopt the Grad-CAM [79] to visualize attention region of each layer's feature map. As shown in Fig. 11, baseline models training with EKD-FWSNet can obtain better attention on different layers. Specifically, EKD-FWSNet training networks can generate less-noise low-level feature maps and richer-information high-level feature maps.

V. Conclusion

In this paper, we propose a novel training framework, EKD-FWSNet to explore the representation power of baseline models. Baseline models optimizing in EKD-FWSNet perform much better guided by knowledge distillation on posterior class probabilities and intermediate attention feature maps. To ease training burden, we mainly use weight-sharing blocks and concise distillation loss. We also make further mathematical analysis to show how we exploit the representation potential of weight-sharing blocks. Extensive experiments on lightweight, high-efficiency and large-scale models prove that EKD-FWSNet is more robust and competitive than previous knowledge distillation guided ensemble training framework. Visualization and analysis also prove that EKD-FWSNet is reasonable and interpretable. All in all, our proposed self-learned EKD-FWSNet can enhance baseline models in an easy-optimized manner. After optimizing, baseline models become better without using more trainable parameters and extra complex modules. In the future, we will consider designing a more efficient and flexible ensemble training framework, which will not introduce any extra auxiliary branches.

Appendix
Comparison With Dynamic-Routing Based Dynamic Neural Network

In real applications, different computation-resource devices need different models with different size with high-accuracy. Recent elegant dynamic-routing based DNN integrates different sub-models. Instead of training individual models with different configurations, dynamic-routing based DNN only trains a large “SuperNet” to dynamically switch different routes at runtime using one set of weight-sharing blocks. To improve the generalization of sub-nets, knowledge distillation training mechanism is adopted to transfer knowledge from the large sub-models to multiple small sub-models. Among recent notable dynamic-routing based DNN, BYOT [80] is highly related to our work. The relation between BYOT and EKD-FWSNet is that we both construct multiple forward branches for multiple predictions and adopt knowledge distillation strategy with no dependency on cumbersome teacher model. The difference can be summarized as follows. (1) Two methods have different goals. BYOT mainly aims to conduct adaptive accuracy-efficiency trade-offs on different resource-limited edge devices, i.e. BYOT is designed to meet the inference requirement of devices with different computation-constraints. Compared to BYOT, EKD-FWSNet is a training framework, which mainly aims to make baseline models

TABLE XVIII
COMPARISON AND ANALYSIS WITH BYOT [80]. WE FOLLOW BYOT AND SELECT RESNET-18 AS BASELINE MODEL TO COMPARE THE CLASSIFICATION PERFORMANCE ON CIFAR-100. HERE, "ERR-RATE-E" MEANS THE ERROR RATE OF ENSEMBLE PREDICTION. THE BEST SCORE IS HIGHLIGHTED IN BOLD

Methods	Branch Number	Err-rate (%)	Err-rate-E (%)
BYOT (Baseline)	1	22.91	-
BYOT (ResNet-18)	4	21.36	20.33
EKD-FWSNet (Baseline)	1	23.71	-
EKD-FWSNet (ResNet-18)	4	**20.49**	**18.96**

TABLE XIX
COMPARISON AND ANALYSIS WITH BYOT [80]. WE FOLLOW BYOT AND SELECT RESNET-18 AS BASELINE MODEL TO COMPARE THE CLASSIFICATION PERFORMANCE ON IMAGENET [81]. HERE, "ERR-RATE-E" MEANS THE ERROR RATE OF ENSEMBLE PREDICTION. THE BEST SCORE IS HIGHLIGHTED IN BOLD

Methods	Branch Number	Err-rate (%)	Err-rate-E (%)
BYOT (Baseline)	1	31.88	-
BYOT (ResNet-18)	4	30.16	31.07
EKD-FWSNet (Baseline)	1	30.32	-
EKD-FWSNet (ResNet-18)	4	**27.64**	**27.03**

stronger. Different-scale baseline models can insert into EKD-FWSNet and get improved without depending on teacher models. (2) Two methods have different structures. BYOT proposes a single neural network which contains multiple sub-nets with different depths. During inference, it can switch different sub-nets according to different computation demands. EKD-FWSNet proposes one auxiliary classmate branch to generate multiple forward branches. During inference, classmate branch will be removed and only leave main student branch. (3) Two methods adopt different distillation mechanisms. In BYOT, full-net are served as "teacher" to "teach" each sub-nets. In EKD-FWSNet, multiple predictions are integrated as "ensemble teacher" to only "teach" main student (baseline model).

In this appendix, we conduct classification experiments to compare our method with BYOT. On ResNet-18, BYOT uses early-exiting strategy to construct four forward branches, where the large sub-net (full-net: "classifier4/4") contains the complete baseline model. In Tab.XVIII, we compare the baseline model training in EKD-FWSNet with the full-net of BYOT. We also compare the ensemble teacher of four-forward-branch EKD-FWSNet with the ensemble predictions of four sub-nets of BYOT. The results show that baseline model (ResNet-18) optimizing in EKD-FWSNet achieves lower error rate (21.36% vs 20.49%) and larger gain (3.22% vs 1.55%). Furthermore, we conduct comparison experiments on large-scale dataset, ImageNet [81]. From Tab.XIX, we analyze in the following two points. (1) In terms of baseline model optimization, EKD-FWSNet performs better than BYOT. Obviously, EKD-FWSNet achieves lower error rate on final prediction and ensemble prediction compared to BYOT. In addition, EKD-FWSNet can achieve larger gain. (2) On ImageNet, some sub-nets (E.g. "classifier1/4", "classifier2/4") of BYOT have poor performance with small group of trainable parameters because the classification task is too complex. Therefore, the ensemble teacher performs worse than baseline model. Compared to BYOT, forward branches of EKD-FWSNet all have same amount of trainable parameters, so each forward branch can generate "good and diverse" prediction leading to a "better ensemble teacher". Naturally, in knowledge distillation ensemble training framework, a better ensemble teacher really means a lot.

As a typical dynamic neural network, the huge superiority of BYOT is that it can flexibly provide multiple sub-nets with different inference computation costs. Moreover, BYOT does not introduce any auxiliary modules. Even though EKD-FWSNet utilizes weight-sharing blocks to decrease the amount of auxiliary modules, it still increases the training cost compared to BYOT. All in all, on optimizing baseline model, EKD-FWSNet is better. On flexibly optimizing different-scale sub-nets at the same time with one training framework, BYOT is better.

REFERENCES

[1] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet classification with deep convolutional neural networks," in *Proc. Adv. Neural Inf. Process. Syst.*, 2012, pp. 1106–1114.
[2] M. D. Zeiler and R. Fergus, "Visualizing and understanding convolutional networks," in *Proc. Eur. Conf. Comput. Vis. (ECCV)*, vol. 8689, 2014, pp. 818–833.
[3] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," in *Proc. Int. Conf. Learn. Represent.*, 2015, pp. 1–14.
[4] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2016, pp. 770–778.
[5] C. Szegedy et al., "Going deeper with convolutions," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2015, pp. 1–9.
[6] G. Huang, Z. Liu, L. Van Der Maaten, and K. Q. Weinberger, "Densely connected convolutional networks," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jul. 2017, pp. 2261–2269.
[7] P. Yuan et al., "HS-ResNet: Hierarchical-split block on convolutional neural network," 2020, *arXiv:2010.07621*.
[8] D. Avola, L. Cinque, A. Fagioli, S. Filetti, G. Grani, and E. Rodola, "Multimodal feature fusion and knowledge-driven learning via experts consult for thyroid nodule classification," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 32, no. 5, pp. 2527–2534, May 2022.
[9] R. Girshick, J. Donahue, T. Darrell, and J. Malik, "Rich feature hierarchies for accurate object detection and semantic segmentation," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2014, pp. 580–587.
[10] S. Ren, K. He, R. Girshick, and J. Sun, "Faster R-CNN: Towards real-time object detection with region proposal networks," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 39, no. 6, pp. 1137–1149, Jun. 2017.
[11] G. Jocher et al., "ultralytics/YOLOv5," V5.0, Zenodo, Apr. 2021, doi: 10.5281/zenodo.4679653.
[12] X. Zhu, S. Lyu, X. Wang, and Q. Zhao, "TPH-YOLOv5: Improved YOLOv5 based on transformer prediction head for object detection on drone-captured scenarios," in *Proc. IEEE/CVF Int. Conf. Comput. Vis. Workshops (ICCVW)*, Oct. 2021, pp. 2778–2788.
[13] E. Shelhamer, J. Long, and T. Darrell, "Fully convolutional networks for semantic segmentation," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 39, no. 4, pp. 640–651, Apr. 2017.
[14] H. Zhao, J. Shi, X. Qi, X. Wang, and J. Jia, "Pyramid scene parsing network," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jul. 2017, pp. 6230–6239.
[15] L. C. Chen, G. Papandreou, and I. Kokkinos, "DeepLab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected CRFs," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 40, no. 4, pp. 834–848, Jun. 2016.

[16] Z. Zhou et al., "UNet++: Redesigning skip connections to exploit multiscale features in image segmentation," *IEEE Trans. Med. Imag.*, vol. 39, no. 6, pp. 1856–1867, Dec. 2019.

[17] J. Fu et al., "Dual attention network for scene segmentation," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2019, pp. 3146–3154.

[18] C. Yang, H. Zhou, Z. An, X. Jiang, Y. Xu, and Q. Zhang, "Cross-image relational knowledge distillation for semantic segmentation," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2022, pp. 12309–12318.

[19] Z. Allen-Zhu and Y. Li, "Towards understanding ensemble, knowledge distillation and self-distillation in deep learning," 2020, *arXiv:2012.09816*.

[20] D. Walawalkar, Z. Shen, and M. Savvides, "Online ensemble model compression using knowledge distillation," in *Proc. Eur. Conf. Comput. Vis.*, vol. 12364, 2020, pp. 18–35.

[21] U. Asif, J. Tang, and S. Harrer, "Ensemble knowledge distillation for learning improved and efficient networks," in *Proc. Eur. Conf. Artif. Intell.*, vol. 325, 2020, pp. 953–960.

[22] X. Lan, X. Zhu, and S. Gong, "Knowledge distillation by on-the-fly native ensemble," in *Proc. Adv. Neural Inf. Process. Syst.*, 2018, pp. 7528–7538.

[23] Y. Zhang, T. Xiang, T. M. Hospedales, and H. Lu, "Deep mutual learning," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 4320–4328.

[24] T. Su et al., "Deep cross-layer collaborative learning network for online knowledge distillation," *IEEE Trans. Circuits Syst. Video Technol.*, early access, Nov. 14, 2022, doi: 10.1109/TCSVT.2022.3222013.

[25] J. Hu, L. Shen, and G. Sun, "Squeeze-and-excitation networks," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 7132–7141.

[26] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: A simple way to prevent neural networks from overfitting," *J. Mach. Learn. Res.*, vol. 15, no. 1, pp. 1929–1958, Jan. 2014.

[27] A. Krizhevsky et al., "Learning multiple layers of features from tiny images," Citeseer, Princeton, NJ, USA, Tech. Rep. 7, 2009.

[28] C. Wah et al., "The Caltech-UCSD birds-200–2011 dataset," California Inst. Technol., Pasadena, CA, USA, Tech. Rep., CNS-TR-2011-001, 2011.

[29] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "ImageNet: A large-scale hierarchical image database," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2009, pp. 248–255.

[30] K. He et al., "Identity mappings in deep residual networks," in *Proc. Eur. Conf. Comput. Vis.*, 2016, pp. 630–645.

[31] M. Tan and Q. V. Le, "EfficientNet: Rethinking model scaling for convolutional neural networks," in *Proc. Int. Conf. Mach. Learn.*, vol. 97, 2019, pp. 6105–6114.

[32] S. Lyu et al., "Make baseline model stronger: Embedded knowledge distillation in weight-sharing based ensemble network," in *Proc. Brit. Mach. Vis. Conf.*, 2021, pp. 1–16. [Online]. Available: https://www.bmvc2021-virtualconference.com/assets/papers/0212.pdf

[33] F. N. Iandola, S. Han, M. W. Moskewicz, K. Ashraf, W. J. Dally, and K. Keutzer, "SqueezeNet: AlexNet-level accuracy with 50x fewer parameters and < 0.5 MB model size," 2016, *arXiv:1602.07360*.

[34] A. G. Howard et al., "MobileNets: Efficient convolutional neural networks for mobile vision applications," 2017, *arXiv:1704.04861*.

[35] M. Sandler, A. Howard, M. Zhu, A. Zhmoginov, and L.-C. Chen, "MobileNetV2: Inverted residuals and linear bottlenecks," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 4510–4520.

[36] A. Howard et al., "Searching for MobileNetV3," in *Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV)*, Oct. 2019, pp. 1314–1324.

[37] X. Zhang, X. Zhou, M. Lin, and J. Sun, "ShuffleNet: An extremely efficient convolutional neural network for mobile devices," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 6848–6856.

[38] I. Hubara et al., "Binarized neural networks," in *Proc. Adv. Neural Inf. Process. Syst.*, 2016, pp. 4107–4115.

[39] M. Rastegari, V. Ordonez, J. Redmon, and A. Farhadi, "XNOR-Net: ImageNet classification using binary convolutional neural networks," in *Proc. Eur. Conf. Comput. Vis. (ECCV)*, 2016, pp. 525–542.

[40] S. Han, H. Mao, and W. J. Dally, "Deep compression: Compressing deep neural network with pruning, trained quantization and Huffman coding," in *Proc. Int. Conf. Learn. Represent.*, 2016, pp. 1–14.

[41] S. Anwar, K. Hwang, and W. Sung, "Structured pruning of deep convolutional neural networks," *ACM J. Emerg. Technol. Comput. Syst.*, vol. 13, no. 3, p. 32, 2017.

[42] P. Molchanov et al., "Pruning convolutional neural networks for resource efficient inference," in *Proc. Int. Conf. Learn. Represent.*, 2017, pp. 1–17.

[43] H. Li et al., "Pruning filters for efficient ConvNets," in *Proc. Int. Conf. Learn. Represent.*, 2017, pp. 1–13.

[44] G. Hinton, O. Vinyals, and J. Dean, "Distilling the knowledge in a neural network," 2015, *arXiv:1503.02531*.

[45] G. Urban et al., "Do deep convolutional nets really need to be deep and convolutional?" in *Proc. Int. Conf. Learn. Represent.*, 2017, pp. 1–13.

[46] S. Zagoruyko and N. Komodakis, "Paying more attention to attention: Improving the performance of convolutional neural networks via attention transfer," in *Proc. Int. Conf. Learn. Represent.*, 2017, pp. 1–13.

[47] A. Romero et al., "FitNets: Hints for thin deep nets," in *Proc. Int. Conf. Learn. Represent.*, 2015, pp. 1–13.

[48] J. Xie et al., "Improving fast segmentation with teacher–student learning," in *Proc. Brit. Mach. Vis. Conf.*, 2018, p. 205.

[49] K. Zhang, C. Zhanga, S. Li, D. Zeng, and S. Ge, "Student network learning via evolutionary knowledge distillation," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 32, no. 4, pp. 2251–2263, Apr. 2022.

[50] B. Zhao, Q. Cui, R. Song, Y. Qiu, and J. Liang, "Decoupled knowledge distillation," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2022, pp. 11943–11952.

[51] S. Yun, J. Park, K. Lee, and J. Shin, "Regularizing class-wise predictions via self-knowledge distillation," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2020, pp. 13873–13882.

[52] T. Liu, K.-M. Lam, R. Zhao, and G. Qiu, "Deep cross-modal representation learning and distillation for illumination-invariant pedestrian detection," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 32, no. 1, pp. 315–329, Jan. 2022.

[53] J. Hou, H. Ding, W. Lin, W. Liu, and Y. Fang, "Distilling knowledge from object classification to aesthetics assessment," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 32, no. 11, pp. 7386–7402, Nov. 2022.

[54] M. Ji, B. Heo, and S. Park, "Show, attend and distill: Knowledge distillation via attention-based feature matching," in *Proc. AAAI Conf. Artif. Intell.*, 2021, pp. 7945–7952.

[55] Y. Liu, K. Chen, C. Liu, Z. Qin, Z. Luo, and J. Wang, "Structured knowledge distillation for semantic segmentation," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2019, pp. 2604–2613.

[56] K. Zhao, T. Matsukawa, and E. Suzuki, "Retraining: A simple way to improve the ensemble accuracy of deep neural networks for image classification," in *Proc. 24th Int. Conf. Pattern Recognit. (ICPR)*, Aug. 2018, pp. 860–867.

[57] Y. Tian, S. Sun, and J. Tang, "Multi-view teacher–student network," *Neural Netw.*, vol. 146, pp. 69–84, Feb. 2022.

[58] T.-B. Xu and C. Liu, "Data-distortion guided self-distillation for deep neural networks," in *Proc. AAAI Conf. Artif. Intell.*, 2019, pp. 5565–5572.

[59] T.-B. Xu and C.-L. Liu, "Deep neural network self-distillation exploiting data representation invariance," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 33, no. 1, pp. 257–269, Jan. 2022.

[60] M. Ji, S. Shin, S. Hwang, G. Park, and I.-C. Moon, "Refine myself by teaching myself: Feature refinement via self-knowledge distillation," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2021, pp. 10664–10673.

[61] C. Yang et al., "MixSKD: Self-knowledge distillation from mixup for image recognition," in *Proc. Eur. Conf. Comput. Vis.*, vol. 13684, 2022, pp. 534–551.

[62] H. Zhang et al., "mixup: Beyond empirical risk minimization," in *Proc. Int. Conf. Learn. Represent.*, 2018, pp. 1–13.

[63] D. H. Le, V. T. Nhan, and N. Thoai, "Paying more attention to snapshots of iterative pruning: Improving model compression via ensemble distillation," in *Proc. Brit. Mach. Vis. Conf.*, 2020, pp. 1–16.

[64] G. Huang et al., "Snapshot ensembles: Train 1, get *M* for free," in *Proc. Int. Conf. Learn. Represent.*, 2017, pp. 1–14.

[65] T. Su, Q. Liang, J. Zhang, Z. Yu, G. Wang, and X. Liu, "Attention-based feature interaction for efficient online knowledge distillation," in *Proc. IEEE Int. Conf. Data Mining (ICDM)*, Dec. 2021, pp. 579–588.

[66] G. Wu and S. Gong, "Peer collaborative learning for online knowledge distillation," in *Proc. AAAI Conf. Artif. Intell.*, 2021, pp. 10302–10310.

[67] T. Chen, S. Kornblith, M. Norouzi, and G. Hinton, "A simple framework for contrastive learning of visual representations," in *Proc. 37th Int. Conf. Mach. Learn. (PMLR)*, vol. 119, Jul. 2020, pp. 1597–1607.

[68] T. Chen et al., "Big self-supervised models are strong semi-supervised learners," in *Proc. Adv. Neural Inf. Process. Syst.*, 2020, pp. 1–13.

[69] K. He, H. Fan, Y. Wu, S. Xie, and R. Girshick, "Momentum contrast for unsupervised visual representation learning," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2020, pp. 9726–9735.

[70] J. Grill et al., "Bootstrap your own latent—A new approach to self-supervised learning," in *Proc. Adv. Neural Inf. Process. Syst.*, 2020, pp. 21271–21284.

[71] Y. Han et al., "Dynamic neural networks: A survey," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 44, no. 11, pp. 7436–7456, Oct. 2021.

[72] X. Wang, F. Yu, Z. Y. Dou, T. Darrell, and J. E. Gonzalez, "SkipNet: Learning dynamic routing in convolutional networks," in *Proc. Eur. Conf. Comput. Vis. (ECCV)*, vol. 11217, 2018, pp. 420–436.

[73] A. Veit and S. Belongie, "Convolutional networks with adaptive inference graphs," *Int. J. Comput. Vis.*, vol. 128, no. 3, pp. 730–741, Mar. 2020.

[74] B. E. Bejnordi, T. Blankevoort, and M. Welling, "Batch-shaping for learning conditional channel gated networks," in *Proc. Int. Conf. Learn. Represent.*, 2020, pp. 1–14.

[75] L. Zhang, J. Song, A. Gao, J. Chen, C. Bao, and K. Ma, "Be your own teacher: Improve the performance of convolutional neural networks via self distillation," in *Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV)*, Oct. 2019, pp. 3712–3721.

[76] J. Yu et al., "Slimmable neural networks," in *Proc. Int. Conf. Learn. Represent.*, 2019, pp. 1–16.

[77] B. Zhao, X. Wu, J. Feng, Q. Peng, and S. Yan, "Diversified visual attention networks for fine-grained object classification," *IEEE Trans. Multimedia*, vol. 19, no. 6, pp. 1245–1256, Jun. 2017.

[78] L. van der Maaten and G. Hinton, "Visualizing data using t-SNE," *J. Mach. Learn. Res.*, vol. 9, pp. 2579–2605, Nov. 2008.

[79] R. R. Selvaraju, M. Cogswell, A. Das, R. Vedantam, D. Parikh, and D. Batra, "Grad-CAM: Visual explanations from deep networks via gradient-based localization," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Oct. 2017, pp. 618–626.

[80] L. Zhang, J. Song, A. Gao, J. Chen, C. Bao, and K. Ma, "Be your own teacher: Improve the performance of convolutional neural networks via self distillation," in *Proc. IEEE Int. Conf. Comput. Vis.*, Oct. 2019, pp. 3712–3721.

[81] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "ImageNet: A large-scale hierarchical image database," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2009, pp. 248–255.

Shuchang Lyu (Graduate Student Member, IEEE) received the B.S. degree from the Department of Communication and Information Engineering, Shanghai University, Shanghai, China, in 2016, and the M.E. degree from the Department of Electronic and Information Engineering, Beihang University, Beijing, China, in 2019, where he is currently pursuing the Ph.D. degree. His current research interests include knowledge distillation method, high-efficiency network design, few-shot semantic segmentation, and object detection.

Lijiang Chen (Member, IEEE) received the B.S. and Ph.D. degrees from the Department of Electronic and Information Engineering, Beihang University, in 2007 and 2012, respectively. He is currently an Assistant Professor with the Department of Electronic and Information Engineering, Beihang University. His current research interests include natural language processing, medical image understanding, and human–computer interaction.

Binghao Liu received the B.E. degree in electronics and information engineering from Beihang University, Beijing, China, in 2019, where he is currently pursuing the Ph.D. degree with the School of Electronic and Information Engineering. His research interests include weakly supervised learning, semantic segmentation, and few-shot semantic segmentation.

Ting-Bing Xu received the Ph.D. degree from the National Laboratory of Pattern Recognition (NLPR), Institute of Automation, Chinese Academy of Sciences, Beijing, China, in 2020. He is currently a Post-Doctoral Fellow with the Department of Instrumentation and Optoelectronic Engineering, Beihang University, Beijing. His research interests include knowledge distillation, high-efficiency network design, autonomous driving, and handwriting recognition.

Guangliang Cheng received the Ph.D. degree from the National Laboratory of Pattern Recognition, Institute of Automation, Chinese Academy of Sciences, China, in 2017. From 2017 to 2019, he was a Post-Doctoral Researcher with the Institute of Remote Sensing and Digital Earth, Chinese Academy of Sciences. From 2019 to 2023, he was the Associate Research Director with SenseTime. He is currently a Reader (Associate Professor) with the Department of Computer Science, University of Liverpool, U.K. His research interests include computer vision, autonomous driving, and remote sensing image processing.

Qi Zhao (Member, IEEE) received the Ph.D. degree in communication and information systems from Beihang University, Beijing, China. From 2014 to 2015, she was a Visiting Scholar with the Department of Electrical and Computer Engineering, University of Pittsburgh. Since 2016, she has been working on wearable device based on first-view image processing and deep learning based on image recognition. She is currently a Professor with Beihang University. Her current research interests include one-shot semantic segmentation, communication signal processing, and target tracking.

Wenquan Feng received the Ph.D. degree in communication and information systems from Beihang University, Beijing, China. He is currently a Professor with Beihang University, where he is also the President of the Qingdao Research Institute. He has been working on using artificial intelligence techniques to handle image data from satellite. His research interests include remote sensing scene understanding, multi-modality image processing, and satellite data mining.