Adaptive Hierarchy-Branch Fusion for Online Knowledge Distillation

Linrui Gong[1], Shaohui Lin[1]*, Baochang Zhang[2], Yunhang Shen[3], Ke Li[3], Ruizhi Qiao[3], Bo Ren[3], Muqing Li[3], Zhou Yu[1,4], Lizhuang Ma[1]

[1]East China Normal University, Shanghai, China
[2]Beihang University, China
[3]Tencent Youtu Lab, China
[4]Key Laboratory of Advanced Theory and Application in Statistics and Data Science - MOE, China
linruigong965@gmail.com, bczhang@buaa.edu.cn, {odysseyshen,tristanli,ruizhiqiao,timren,ericqli}@tencent.com, {shlin, lzma}@cs.ecnu.edu.cn, zyu@stat.ecnu.edu.cn

Abstract

Online Knowledge Distillation (OKD) is designed to alleviate the dilemma that the high-capacity pre-trained teacher model is not available. However, the existing methods mostly focus on improving the ensemble prediction accuracy from multiple students (*a.k.a.* branches), which often overlook the homogenization problem that makes student models saturate quickly and hurts performance. We assume that the intrinsic bottleneck of the homogenization problem comes from the identical branch architecture and coarse ensemble strategy. We propose a novel Adaptive Hierarchy-Branch Fusion framework for Online Knowledge Distillation, termed AHBF-OKD, which designs *hierarchical branches* and *adaptive hierarchy-branch fusion* module to boost the model diversity and learn complementary knowledge. Specifically, we first introduce hierarchical branch architectures to construct diverse peers by increasing the depth of branches monotonously on the basis of the target branch. To effectively transfer knowledge from the most complex branch to the simplest target branch, we propose an adaptive hierarchy-branch fusion module to create hierarchical teacher assistants recursively, which regards the target branch as the smallest teacher assistant. During the training, the teacher assistant from the previous hierarchy is explicitly distilled by the teacher assistant and the branch from the current hierarchy. Thus, the important scores to different branches are effectively and adaptively allocated to reduce branch homogenization. Extensive experiments demonstrate the effectiveness of AHBF-OKD on different datasets, including CIFAR-10/100 and ImageNet 2012. For example, the distilled ResNet18 achieves the Top-1 error of 29.28% on ImageNet 2012, which significantly outperforms the state-of-the-art methods. The source code is available at https://github.com/linruigong965/AHBF.

Introduction

Deep neural networks have achieved remarkable success in various scenarios (He et al. 2016; Ren et al. 2015; He et al. 2017). However, the over-parameterized models are difficult to deploy on resource-limited devices, such as mobile and embedded devices. To make a trade-off between the model simplicity and efficiency, Knowledge Distillation (KD) techniques (Hinton, Vinyals, and Dean 2015; Romero et al. 2015) have been widely used to transfer knowledge from a high-capacity teacher to a high-efficiency student.

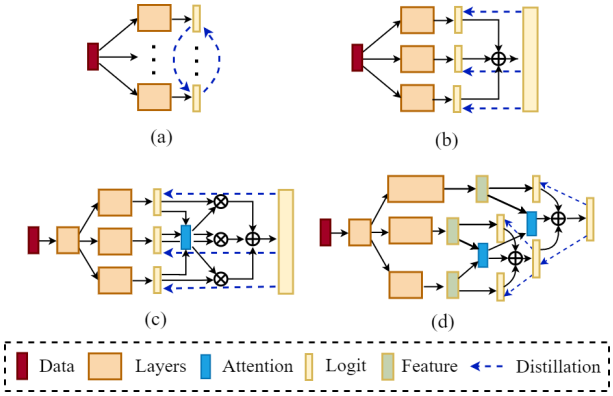


Figure 1: Different schematic frameworks of online knowledge distillation. (a) Mutual learning, (b) Ensemble the independent models by averaging outputs. (c) Ensemble the identical branches with the attention mechanism. (d) The proposed AHBF-OKD with gradual hierarchical distillation.

Traditional KD methods (Hinton, Vinyals, and Dean 2015; Zagoruyko and Komodakis 2017; Yim et al. 2017) employ a two-stage training procedure, *i.e.*, first pre-training a powerful teacher and then distilling its knowledge to a compact student. Despite the significant improvements in student discriminative ability, the following two problems still exist, prohibiting their usage in real applications. (1) The high-capacity pre-trained teachers are not always available; (2) The two-stage training process requires high computation costs. To address these issues, Online Knowledge Distillation (OKD) employs an end-to-end teacher-free training paradigm, which has received much research focus.

The existing OKD methods construct multiple students with identical architecture, one of which is considered as the target model and distilled from its peers. We compare different OKD schematics in Fig. 1. Mutual learning (Zhang et al. 2018) in Fig. 1(a) collaboratively learns the students by transferring the knowledge from their peers. Recently, consistency regularization (Berthelot et al. 2019; Sohn et al. 2020; Xie et al. 2020), a variant of mutual learning, aligns the peer outputs with different data augmentations of the

*Corresponding author

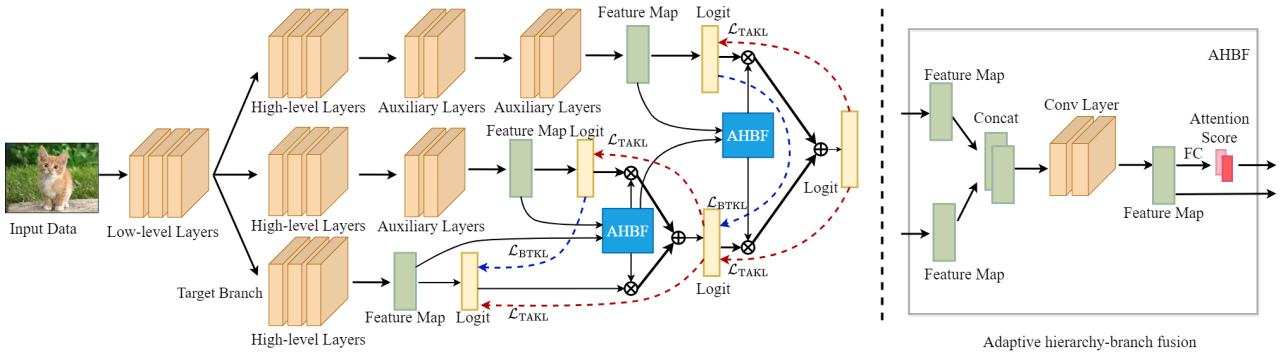


Figure 2: An overview of the proposed AHBF-OKD, which contains two components, hierarchical branch structure and adaptive hierarchy-branch fusion module (right panel).

same images. However, their performance improvement is marginal due to the low discriminability of each peer. Alternatively, as shown in Fig. 1(b), ensemble knowledge by averaging peer logits (Allen-Zhu and Li 2020) improves teacher discriminability and allows the target student to learn rich knowledge from the teacher. To further improve peer diversity, several methods introduce an attention mechanism to assign different scores to each branch for ensembling during training, as shown in Fig. 1(c). For example, the gate module (Lan, Zhu, and Gong 2018) ensembles all branch features to estimate the important score of the corresponding branch. The works in (Chen et al. 2020; Li et al. 2020) apply the self-attention mechanism to generate different attention scores to build an ensemble teacher. Despite the diversity attained by the attention mechanism, these methods still exist in the *homogenization problem* that different branches learn the same semantic features on the misclassified images.

To address the above problems, we propose a novel Online Knowledge Distillation via *Adaptive Hierarchy-Branch Fusion* (AHBF-OKD), which designs the hierarchical branch structures and adaptive hierarchy-branch fusion (AHBF) module to boost the model diversity and learn complementary knowledge. As shown in Fig. 1(d), hierarchical branches with diverse structures are first constructed by increasing the depth of peer monotonously based on the target branch. Then, the adaptive hierarchy-branch fusion module is proposed to create hierarchical teacher assistants recursively regarding the target branch as the smallest teacher assistant. Instead of dense connecting to multiple hierarchical teacher assistants, the teacher assistant and branch of the current hierarchy only explicitly distill the smaller teacher assistant of the previous hierarchy, which reduces the gap in the coarse distillation that roughly transfers knowledge from the ensemble logits of all branches. Thus, each branch learns the knowledge from more high-capacity branches and teacher assistants. Furthermore, we introduce simple yet effective attention learning to adaptively assign the important scores for different branches to build teacher assistants, which takes the features from the last AHBF and the current branch. We only keep the target branch during inference and remove the AHBF module and other branches.

We summarize our contributions as follows:

1. We propose a novel Adaptive Hierarchy-Branch Fusion framework for Online Knowledge Distillation (AHBF-OKD), which address the homogenization problem with hierarchical branch structure and hierarchy-branch fusion.
2. Recursive teacher assistants gradually reduce the gap between different hierarchy-branches and adaptively assign important scores for effective knowledge distillation.
3. Extensive experiments demonstrate the superior performance of the proposed AHBF-OKD on various datasets and network architectures. For example, on ImageNet 2012, AHBF-OKD outperforms baseline ResNet18 by 1.21% and also surpasses the SOTA OKD methods.

Related Work

Two-stage knowledge distillation. KD originates from the idea that the output of a pre-trained big model can be combined with the labeled data to train a small model (Bucilua, Caruana, and Niculescu-Mizil 2006). The work (Hinton, Vinyals, and Dean 2015) popularizes this idea by forcing the output softened logits of the student model to imitate the teacher model during training. Recently, many methods have been proposed to further exploit the teacher-student alignment of intermediate feature maps (Romero et al. 2015), instance relational graphs (Liu et al. 2019), similarity attention maps (Ji, Heo, and Park 2021) and generative adversarial predictions (Micaelli and Storkey 2019). However, these methods still follow a two-stage training paradigm (*i.e.*, first pre-training teacher and then distillation), which will significantly increase computational overhead. Moreover, the pre-trained teacher models are not always available in some scenarios, which leads to less commercial attractiveness.

Online knowledge distillation. To overcome the aforementioned drawbacks, OKD have been proposed by simultaneously optimizing both student and teacher in an end-to-end training manner, which simplifies the training process to save training computation overhead. For example, DML (Zhang et al. 2018) trains multiple models with the same capacity simultaneously and distills the knowledge from each other by mutual learning. Besides, feature-level adversarial training (Chung et al. 2020) is also leveraged into

mutual learning for OKD. KDCL (Guo et al. 2020) further averages the output logits of all students as the soft target, which is used as the knowledge to distill the target student. The works (Lan, Zhu, and Gong 2018; Chen et al. 2020) introduce the gate or attention mechanism to derive an individual target for each student and employ two-level distillation to transfer the strong ensemble teacher to the target student. Recently, the diversity in the logits of branches has been explored by feature-fusion module with classifier diversification (Li et al. 2020; Kim et al. 2021) and peer collaborative learning (Wu and Gong 2021). Differently, our method employs hierarchical branches to increase feature diversity during training and recursively constructs hierarchical teacher assistants to reduce the distillation gap.

Knowledge distillation by teacher assistant. Teacher assistant has been first proposed in (Mirzadeh et al. 2020) to reduce the large gap between student and teacher by a moderate model as transition (Mirzadeh et al. 2020). This strategy (Wang et al. 2020) is also effectively extended to distill assistants have been proposed for knowledge distillation, such as temporal mean teacher (Wu and Gong 2021), densely guided assistants (Son et al. 2021), attention-based meta network (Ji, Heo, and Park 2021) and cross-stage teacher (Chen et al. 2021). Unlike these methods, our hierarchical teacher assistants are recursively constructed by the adaptive hierarchy-branch fusion module, such that the knowledge from the highest-capacity branches is effectively transferred to the lowest-capacity target branch without dense connections.

Preliminaries

Notations. As illustrated in Fig. 2, suppose we have a network architecture with M branches, where low-level layers in each branch share the same parameter θ_{ll}, and the m-th branch has its own high-level layers with parameter θ_{hl}^m and fully-connected (FC) layers for classification with parameter $\theta_{fc}^m, m = 1, \cdots, M$. Additional auxiliary layers are added into each branch with parameter $\theta_a^m (m = 1, \cdots, M)$ in our OKD framework, where the first branch is denoted as the target one (*a.k.a.* a given student network) without θ_a^1 (*i.e.*, $\theta_a^1 \in \varnothing$). Without considering the AHBF module, we denote the parameter set of the m-th branch as $\Theta^m = \{\theta_{ll}, \theta_{hl}^m, \theta_a^m, \theta_{fc}^m\}$. The generated feature maps before the FC layer are denoted by F^m using the parameters θ_{ll}, θ_{hl}^m and θ_a^m.

Given a labeled training sample x with corresponding label $y \in \{0, 1\}^C$ from C classes, the output probability of class c given by the m-th branch f^m is computed as:

$$p_c^m = \frac{\exp(z_c^m)}{\sum_{i=1}^{C} \exp(z_i^m)}, \tag{1}$$

where z^m is the logit outputs of the nueral network f^m with parameter Θ^m. For multi-class classification, the objective function minimizes the cross entropy loss between the predicted vector and the ground-truth label y (a one-hot vector):

$$\mathcal{L}_{\mathrm{CE}}^m = \sum_{c=1}^{C} y_c \log p_c^m. \tag{2}$$

Traditional Knowledge Distillation

Generally, traditional knowledge distillation (KD) uses Kullback-Leibler divergence to minimize the distribution divergence between the softened logits of student f^1 and the teacher f^t model (Hinton, Vinyals, and Dean 2015). The softened output is formulated as:

$$\widetilde{p}_c^1 = \frac{\exp(z_c^1/\tau)}{\sum_{i=1}^{C} \exp(z_i^1/\tau)}, \widetilde{p}_c^t = \frac{\exp(z_c^t/\tau)}{\sum_{i=1}^{C} \exp(z_i^t/\tau)}, \tag{3}$$

where τ is the temperature parameter. Thus, the loss function of traditional knowledge distillation is denoted as:

$$\mathcal{L}_{\mathrm{KD}} = \tau^2 \sum_{c=1}^{C} \widetilde{p}_c^t \log \frac{\widetilde{p}_c^t}{\widetilde{p}_c^1}, \tag{4}$$

where $\widetilde{p}_c^t$ and $\widetilde{p}_c^s$ denote the softened outputs of teacher and student models, respectively. Thus, the overall loss of traditional KD with balancing parameter λ is formulated as:

$$\mathcal{L}_{\mathrm{TKD}} = \mathcal{L}_{\mathrm{CE}}^1 + \lambda \mathcal{L}_{\mathrm{KD}}. \tag{5}$$

Online Knowledge Distillation

Previous OKD methods consider M branches with identical architecture f^1 and all branches are optimized simultaneously during the training. OKD employs an appropriate fusion function $\mathcal{D}(\cdot)$ (*e.g.* self-attention or soft gate) to generate the important score $s \in \mathbf{R}^M$ using branch features F^m, and then the score is merged into the logit z_c^m of the corresponding branches to build a strong teacher. Thus, the logit of aggregated teacher f^{ta} is denoted as:

$$p_c^{\mathrm{at}} = \sum_{m=1}^{M} s_m \cdot p_c^m = \sum_{m=1}^{M} \mathcal{D}_m(F^1, \cdots, F^M) \cdot p_c^m. \tag{6}$$

Correspondingly, we obtain the softened logit of the aggregated teacher denoted by $\widetilde{p}_c^{\mathrm{at}}$. Thus, the overall loss function of OKD is constructed by aligning the softened output of the aggregated teacher and that of each branch using Eq. 4, as well as the cross entropy loss Eq. 2:

$$\mathcal{L}_{\mathrm{OKD}} = \sum_{m=1}^{M} \mathcal{L}_{\mathrm{CE}}^m + \mathcal{L}_{\mathrm{CE}}^{\mathrm{at}} + \lambda \cdot \tau^2 \sum_{k=1}^{M} \sum_{c=1}^{C} \widetilde{p}_c^{\mathrm{at}} \log \frac{\widetilde{p}_c^{\mathrm{at}}}{\widetilde{p}_c^m}. \tag{7}$$

However, minimizing the Eq. 7 may lead to the homogenization problem, as the same branch architecture and coarse ensemble strategy are used for distillation.

The Proposed AHBF-OKD

Hierarchical Branch Structure

As shown in Fig. 2, the hierarchical branch structure is constructed by adding auxiliary layers after high-level layers while keeping the same shared low-level layers. The first hierarchy-branch is our target network without auxiliary layers, and the following hierarchy-branches gradually increase the number of a blocks to build the diversity structure, where a is set to a multiple of 2. We call the target branch the first hierarchy-level branch.

For more clear presentation, we take ResNet32 architecture, for example, to elaborate on how to add the auxiliary layers. All hierarchy-branches share low-level layers in the first two stages and have their separated parameters after the second stage. We add different auxiliary layers after the third stage in the original ResNet32 containing the number of 10 layers (*i.e.*, 5 blocks) to build hierarchical branches according to the hierarchy level. Thus, the m-th level branch needs to add $2a(m-1)$ auxiliary layers, where 2 is the layer number in one block.

By constructing the hierarchical branch structure, we learn more diverse features for the following effective fusion, which alleviates the homogenization problem. Actually, we also compress the target network by setting it as the M-th branch and decreasing the layers to build other hierarchical branches in the opposite way. A more detailed discussion is presented in the experiments.

Adaptive Hierarchy-Branch Fusion

After constructing a hierarchical branch structure, we propose hierarchical teacher assistants recursively by the hierarchical attention score to gradually reduce the gap between the deepest M-th branch to the target one.

Hierarchical teacher assistants. Inspired by (Mirzadeh et al. 2020), teacher assistant (TA), as an intermediate teacher, is used to reduce the gap by transferring the knowledge from the huge capacity teacher to the compact student. In (Son et al. 2021), all branches regarded as TA (except the smallest one) guide each TA to learn every other smaller TA densely, which leads to large training computation overhead and the gap between them is still large. To this end, our hierarchical teacher assistants reduce this gap by recursively merging the last TA and the current hierarchy-branch. Moreover, the knowledge transfer comes from the adjacent TA without dense connections.

Fig. 2 illustrates the construction of hierarchical teacher assistants. For simplicity, we take the first hierarchy-level branch as the first TA, *i.e.*, its logit $p_c^{\text{ta}(1)}$ is equal to p_c^1. Then, the output logits of hierarchical teacher assistants are constructed by:

$$p_c^{\text{ta}(m)} = s_1^m p_c^m + s_2^m p_c^{\text{ta}(m-1)}, m = 2, \cdots, M, \quad (8)$$

where $s^m \in \mathbf{R}^2$ is the hierarchical attention score, which will be introduced in the following parts. Then, the softened output logits of hierarchical teacher assistants $\widetilde{p}_c^{\text{ta}(m)}$ are generated by Eq. 3.

With the help of hierarchical teacher assistants, we construct two kinds of distillation losses for knowledge transfer as: (1) $\mathcal{L}_{\text{TAKL}}^m$, knowledge distillation from the m-th TA to the $(m-1)$-th TA, and (2) $\mathcal{L}_{\text{BTKL}}^m$, knowledge distillation from the m-th hierarchy-branch to the $(m-1)$-th TA. These two losses are formulated using KL-divergence as:

$$\mathcal{L}_{\text{TAKL}}^m = \tau^2 \sum_{c=1}^{C} \widetilde{p}_c^{\text{ta}(m)} \log \frac{\widetilde{p}_c^{\text{ta}(m)}}{\widetilde{p}_c^{\text{ta}(m-1)}}. \quad (9)$$

$$\mathcal{L}_{\text{BTKL}}^m = \tau^2 \sum_{c=1}^{C} \widetilde{p}_c^{m} \log \frac{\widetilde{p}_c^{m}}{\widetilde{p}_c^{\text{ta}(m-1)}}. \quad (10)$$

Thus, we formulate the final knowledge distillation loss as:

$$\mathcal{L}_{\text{FKL}} = \sum_{m=2}^{M} (\lambda_1 \mathcal{L}_{\text{TAKL}}^m + \lambda_2 \mathcal{L}_{\text{BTKL}}^m), \quad (11)$$

where λ_1 and λ_2 are balanced parameters.

Hierarchical attention score. The attention mechanism is often used to allocate the importance score s for each branch and construct a more powerful teacher for OKD (Lan, Zhu, and Gong 2018; Chen et al. 2020; Li et al. 2021). Different from these methods by using all features of branches, we introduce hierarchical attention to construct the hierarchical teacher assistant by only fusing features from two branches. Inspired by the FPN (Lin et al. 2017), the input features for hierarchical attention in the m-th hierarchy are built by bottom-to-top, which are from the $(m-1)$-th teacher assistant $\widetilde{F}^{m-1}$ and the m-th hierarchy-branch F^m by the concatenation operation. As such, the hierarchical attention score s^m in the m-th hierarchy is obtained through a simple network $\mathcal{A}^m$ using the feature $\widetilde{F}^m$ as an input:

$$s^m = \mathcal{A}^m\big([F^m, \widetilde{F}^{m-1}], \theta_{ha}^m\big), \widetilde{F}^1 = F^1, m = 2, \cdots, M, \quad (12)$$

where $[\cdot]$ is feature concatenation operation, $\mathcal{A}^m$ is a sequence of layers including one convolution layer for reducing the channel number, one FC layer, and Softmax operation with parameter θ_{ha}^m. $\widetilde{F}^m, m = 2 \cdots, M-1$ is the feature from the convolutional output in $\mathcal{A}^{m-1}$. Therefore, we use Eq. 12 to generate the logit of each hierarchical TA.

Training and inference. Inspired by the (Laine and Aila 2017), simple initialization on all branches may lead to non-convergence during training. To this end, we introduce the ramp-up function to reduce the training sensitivity for knowledge distillation, which is formulated as:

$$\omega(i) = e^{-5(1-i/E)^2}, \quad (13)$$

where E is a hyper-parameter to decide the smoothness of the ramp-up function and i is the i-th epoch. Therefore, we leverage Eq. 11 into the cross-entropy loss of all hierarchy-branches and hierarchical teacher assistants to construct the overall loss function of the proposed AHBF-OKD, which is formulated as:

$$\mathcal{L} = \sum_{m=1}^{M} \mathcal{L}_{\text{CE}}^m + \sum_{m=2}^{M} \mathcal{L}_{\text{CE}}^{\text{ta(m)}} + \omega(i) \cdot \mathcal{L}_{\text{FKL}}, \quad (14)$$

where $\mathcal{L}_{\text{CE}}^{\text{ta(m)}}$ is the cross-entropy loss between the m-th TA and ground-truth label y. In Eq. 14, the $\omega(i)$ achieves a smaller value in the early epoch, which reduces the effect of knowledge distillation by using the ground-truth label to guide the training. With the increase of $\omega(i)$, the knowledge flow between hierarchy-branches becomes more important.

We directly minimize the overall loss function $\mathcal{L}$ of the proposed AHBF-OKD by stochastic gradient descent (SGD) in an end-to-end manner. After training, we use *the target branch for inference* by safely removing other structures, including other hierarchy-branches and adaptive hierarchy-branch fusion module.

Network	Baseline	CL	ONE	OKDDip	AFID	FFSD-C	AHBF-OKD
ResNet32	6.31 ± 0.07	6.01 ± 0.09	5.99 ± 0.05	5.58 ± 0.08	5.83 ± 0.07	5.49 ± 0.10	$\mathbf{5.32 \pm 0.11}$
ResNet110	5.58 ± 0.09	4.95 ± 0.17	5.17 ± 0.07	4.56 ± 0.11	5.02 ± 0.11	4.48 ± 0.10	$\mathbf{4.33 \pm 0.08}$
VGG16	6.22 ± 0.11	6.15 ± 0.12	6.16 ± 0.08	5.87 ± 0.03	5.98 ± 0.13	5.92 ± 0.16	$\mathbf{5.60 \pm 0.11}$
DenseNet40-12	7.02 ± 0.05	7.11 ± 0.06	6.85 ± 0.15	6.48 ± 0.12	6.62 ± 0.09	6.43 ± 0.07	$\mathbf{6.17 \pm 0.14}$
MobilenetV2(0.5)	14.35 ± 0.23*	14.12 ± 0.15*	14.01 ± 0.05*	13.85 ± 0.17*	14.05 ± 0.15*	13.77 ± 0.14*	$\mathbf{13.64 \pm 0.21}$

Table 1: Error (%) comparison with SOTA methods on CIFAR-10. $a \pm b$ means the mean value a with the standard variance b, and boldface represents the best performance in all tables. * denotes the re-implemented results using the official released codes. MobileNetV2(0.5) means to use the 50% channels of the original MobileNetV2.

Network	Baseline	CL	ONE	OKDDip	AFID	FFSD-C	AHBF-OKD
ResNet32	28.72 ± 0.19	27.67 ± 0.46	27.44 ± 0.05	25.63 ± 0.14	25.95 ± 0.05	25.50 ± 0.10	$\mathbf{25.19 \pm 0.16}$
ResNet110	23.79 ± 0.57	21.17 ± 0.58	21.56 ± 0.09	21.14 ± 0.14	21.43 ± 0.13	21.17 ± 0.12	$\mathbf{20.96 \pm 0.14}$
VGG16	25.68 ± 0.19	25.67 ± 0.08	25.62 ± 0.11	25.15 ± 0.19	25.23 ± 0.08	25.11 ± 0.14	$\mathbf{24.92 \pm 0.12}$
DenseNet40-12	28.97 ± 0.15	28.55 ± 0.34	28.61 ± 0.12	28.34 ± 0.02	28.47 ± 0.06	28.26 ± 0.08	$\mathbf{27.88 \pm 0.23}$
MobileNetV2(0.5)	40.21 ± 0.11	39.37 ± 0.17*	39.16 ± 0.16*	38.29 ± 0.12*	38.91 ± 0.17*	38.12 ± 0.11*	$\mathbf{37.77 \pm 0.14}$

Table 2: Error (%) comparison with SOTA methods on CIFAR-100.

Network	Baseline	ONE	OKDDip	FFSD-C	ABHF-OKD
ResNet18	30.49	29.45	29.37*	29.85	**29.28**
ResNet34	26.76	25.90*	25.60	25.80	**25.47**

Table 3: Error (%) comparison on ImageNet 2012.

Experiments

Experimental Setups

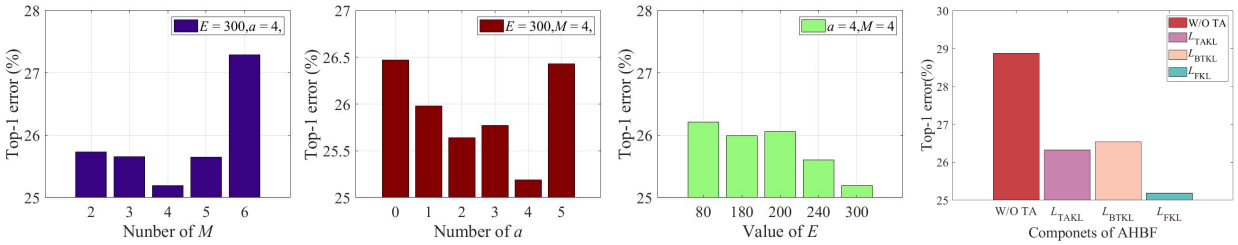
Datasets and model architectures. We evaluate the proposed AHBF-OKD approach on three widely-used datasets, CIFAR-10/100(Krizhevsky, Hinton et al. 2009) and ImageNet 2012(Russakovsky et al. 2015). All images are normalized by channel means and standard deviations, as well as the following standard data augmentation in (He et al. 2016). For model architectures, the baseline networks contain ResNets (He et al. 2016), VGG (Simonyan and Zisserman 2015), DenseNet (Huang et al. 2017) and MobileNetV2 (Sandler et al. 2018).

Implementations. Following ONE (Lan, Zhu, and Gong 2018), we separate the last stage of the baseline network to generate hierarchical multi-branch architecture, and other blocks are regarded as shared low-level layers. The branch number M and auxiliary block number a are both set to 4, unless otherwise specified. We use SGD with Nesterov momentum 0.9 as the optimizer and the temperature τ is set to 3. For CIFAR-10/100 datasets, we set the batch size to 128 and the initial learning rate to 0.1. The learning rate is decayed by 0.1 at the epochs 150 and 225 with 300 epochs in total and the weight decay is set to 5×10^{-4}. For ImageNet 2012, we set batch size to 96, and the learning rate is also initialized by 0.1, which is decayed by 0.1 at epochs 30 and 60 with a total of 90 epochs. The weight decay is set to 1×10^{-4}. In default, the hyper-parameter E is respectively set to 300 and 90 on CIFAR-10/100 and ImageNet 2012, and (λ_1, λ_1) is set to (4, 2). All results are generated by averaging the results over 3 runs. The proposed AHBF-OKD is implemented by PyTorch 1.10 and MindSpore 1.7.0 (Huawei 2020), and trained on two NVIDIA 3090 GPUs.

State-of-the-art OKD methods. We compare the proposed AHBF-OKD to SOTA OKD approaches, including CL (Song and Chai 2018), ONE (Lan, Zhu, and Gong 2018), OKDDip (Chen et al. 2020), AFID (Su et al. 2021) and FFSD (Li et al. 2022). We also train the target networks from scratch without knowledge distillation as *baselines*. For a fair comparison, we compare OKDDip and FFSD directly using the target branch as the leader for the second-level distillation, and the branch number of SOTA methods is set to 4. The top-1 error in the target is used to evaluate performance.

Comparison with SOTA Methods

CIFAR-10. We summarize the results in Tab. 1. Obviously, all OKD methods improve the performance of baselines. For example, the best previous SOTA method FFSD-C outperforms the baseline ResNet32 by 0.82%, and achieves the largest improvement of 1.1% on ResNet110, compared to other baselines. We also observe that our AHBF-OKD achieves the best performance, compared to all SOTA methods. For example, our method achieves a lower error of 4.33% on ResNet110, compared to the best previous SOTA error of 4.48% in FFSD-C. For the high-capacity model VGG16, our method achieves performance gains over FFSD-C by 0.32% . Compared to the previous SOTA methods on the compact MobileNetV2(0.5), AHBF-OKD also shows the best performance of our method.



(a) Effect of branch number (b) Effect of auxiliary block (c) Effect of E (d) Effect of AHBF elements

Figure 3: The Effect of components in ABHF-OKD.

CIFAR-100. In Tab. 2, our method also achieves the best performance to improve various network architectures on CIFAR-100, compared to other SOTA methods. For example, our AHBF-OKD outperforms the best SOTA method OKDDip by 0.18% on ResNet110. For DeseNet40-12, our AHBF-OKD achieves performance gains over FFSD-C by 0.38%. AHBF-OKD is also effective to improve the performance of compact vanilla MobileNetV2(0.5) by 2.44% and outperforms FFSD-C with an error of 38.12% by 0.35%. We also find that the target network architecture significantly affects the performance of online knowledge distillation, the highest capacity VGG16 is more difficult to improve performance using OKD methods, compared to other network architectures. To explain, the ensemble teacher constructed by such high-capacity models may lead to overfitting when training on the limited training data.

ImageNet 2012. We further evaluate the effectiveness of the proposed AHBF-OKD on the large-scale ImageNet dataset and set the auxiliary block a to 2. Tab. 3 summarizes the results of previous SOTA methods and our ABHF-OKD. Our method significantly decreases the Top-1 error of vanilla ResNet18 and ResNet34 by 1.21% and 1.29%, respectively. Moreover, our ABHF-OKD also achieves a new SOTA performance. For example, on ResNet34, our ABHF-OKD surpasses OKDDip by 0.13%. To explain, our ABHF-OKD employs the hierarchical branches to increase diversity and proposes the adaptive hierarchy-branch fusion module to construct hierarchical teacher assistants for effective KD.

Ablation Study

In this section, we select ResNet32 as the target network on CIFAR-100 for an ablation study.

Effect of branch number M. As shown in Fig. 3(a), the number of hierarchical branches varies from 2 to 6. Our method achieves the best performance when setting the branch number to 4. The increase in branch numbers does not consistently improve performance. This is due to the fact that overfitting occurs when significantly increasing the hierarchical branch number. Therefore, the excess branches hinder the effectiveness of knowledge distillation.

Effect of hyper-parameters λ_1, λ_2 and E. We first analyze the sensitivity of E in Fig. 3(c). Obviously, we gradually improve performance by increasing E. When E is set to 300 (*i.e.*, total epoch number), it helps to achieve the lowest error of 25.19%. This indicates that the hierarchical branches require several epochs to stabilize the training in the early epochs, such that the knowledge is well extracted for late knowledge distillation. Intensifying the degree of knowledge distillation in the early time significantly affects final performance. For example, the setting of 300 surpasses that of 80 by 1.02%. In Tab. 4, we further analyze the effect of balanced parameters λ_1 and λ_2 to control the losses of $\mathcal{L}_{\mathrm{TAKL}}$ and $\mathcal{L}_{\mathrm{BTKL}}$, respectively. We observe that (1) without these two losses (*i.e.*, $\lambda_1 = \lambda_2 = 0$), only using the CE-loss of hierarchy-branches and hierarchical teacher assistants *w.r.t.* ground-truth label can not work well; (2) The loss of $\mathcal{L}_{\mathrm{TAKL}}$ is more important to that of $\mathcal{L}_{\mathrm{BTKL}}$ (see 27.04% in the group of (1, 0) *vs.* 27.19% in the group of (0, 1)), which indicates the knowledge transferred from adjacent TAs is richer than that from the next hierarchy-branch. (3) More focus on the minimization of $\mathcal{L}_{\mathrm{TAKL}}$ achieves the better performance, compared to $\mathcal{L}_{\mathrm{BTKL}}$ under the same balanced situations. *E.g.*, the group of (4,2) achieves a lower error of 25.19% than 25.37% in the group of (2,4).

λ_1 \ λ_2	0	1	2	3	4
0	28.88	27.19	26.88	26.59	26.76
1	27.04	25.77	25.98	26.23	26.30
2	26.32	25.61	25.66	25.48	25.37
3	26.76	25.48	25.66	25.39	25.54
4	26.78	25.45	25.19	25.45	25.98

Table 4: Effect of hyper-parameter λ_1 and λ_2 on error (%).

Effect of hierarchical branch structure. In Fig. 3(b), we vary the auxiliary block a from 0 to 5 to explore the effect of hierarchical structure, where 0 presents each branch has the same structure without auxiliary layers. We find that the larger auxiliary block number does not consistently improve performance, which is set to 4 achieving the lowest error of 25.19%. Note that we achieve the error of 26.47% when dropping off all auxiliary layers (*i.e.*, $a = 0$).

Effect of AHBF module. As shown in Fig. 3(d), we evaluate the effectiveness of the AHBF module, which mainly consists of hierarchical teacher assistants and hierarchical attention scores. For hierarchical teacher assistants, we train the hierarchy-branch with the losses of $\mathcal{L}_{\mathrm{TAKL}}$ and

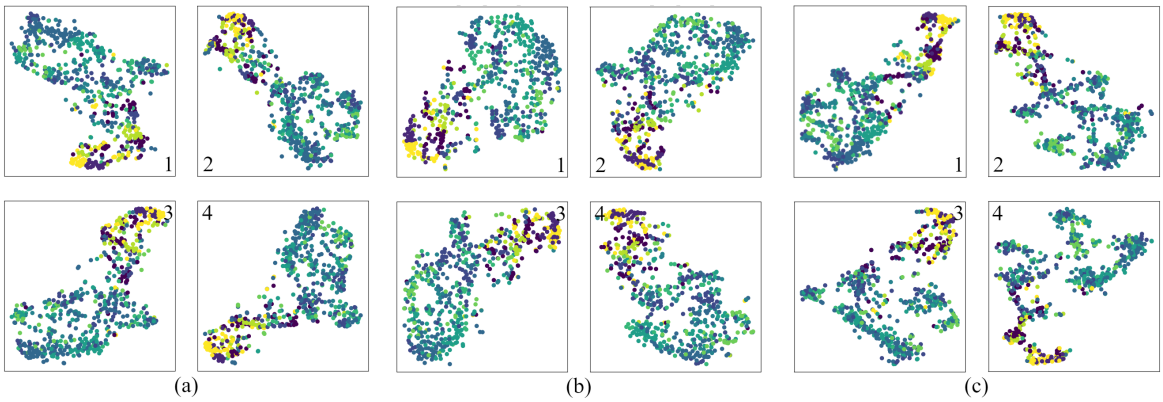


Figure 4: Feature visualization of different branches by t-SNE under the misclassified samples of the target branch (*i.e.*, branch 1). (a) ONE, (b) OKDDip, (c) ABHF-OKD. The numbers denote the branch hierarchy level and each color corresponds to a category in CIFAR-10.

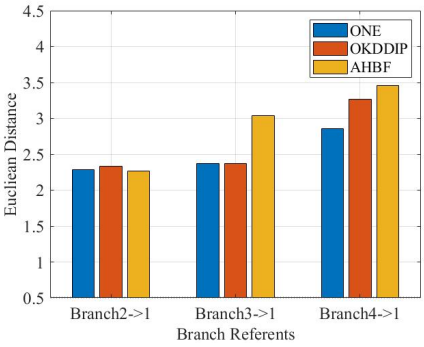


Figure 5: Euclidean distance between the target branch and other ones on the misclassified samples of the target branch using different OKD methods.

Network	a	Method	Error(%)	MFLOPs	Param(M)
ResNet32	-	Baseline	28.72	142.65	1.35
	-	ONE	27.44	142.65	1.35
	$2\downarrow$	Branch4	26.57	142.65	1.35
		Branch3	27.36	114.00	0.95
		Branch2	28.18	86.30	0.81
		Branch1	31.48	57.87	0.36
MobileNetV2(0.5)	-	Baseline	40.21	14.02	1.98
	-	ONE	39.16	14.02	1.98
	$1\downarrow$	Branch4	38.24	14.02	1.98
		Branch3	39.12	13.69	1.73
		Branch2	40.12	13.00	1.64
		Branch1	41.4	12.67	1.39

Table 5: Compression results of target models.

$\mathcal{L}_{\mathrm{BTKL}}$, which achieves the best performance. For example, $\mathcal{L}_{\mathrm{TAKL}}+\mathcal{L}_{\mathrm{BTKL}}$ (deonted by $\mathcal{L}_{\mathrm{FKL}}$) surpasses only $\mathcal{L}_{\mathrm{TAKL}}$ by 1.13%, only $\mathcal{L}_{\mathrm{BTKL}}$ by 1.40% and without the distillation knowledge of hierarchical teacher assistants by 3.79%. It indicates that knowledge is effectively transferred from the deepest branches and TA to the target branch.

Analysis and Discussion

Homogenization problem. We explore the homogenization problem by feature visualization of each branch based on t-SNE (Van der Maaten and Hinton 2008) and the Euclidean distance of each branch logit. As shown in Fig. 4, we find that each branch in the proposed AHBF-OKD has more discriminative than that of other methods for misclassified samples. Fig. 5 further shows each branch in ONE and OKDDip has a more similar logit distance than AHBF-OKD.

Target model compression. Our AHBF-OKD can be extended to compress the target branch by regarding the target branch as the top branch and gradually decreasing the blocks in each stage to construct the hierarchical branch structure. All training parameter remains the same except for a on CIFAR-100. In Tab. 5, our method (Branch3) reduces 28.65 MFLOPs and 0.4M parameters of ResNet32 with the lower error of 27.36%, compared to ONE. AHBF-OKD (Branch3) can easily compress MobileNetV2, which achieves a lower error of 39.12% with only 87% parameters and 92.7% FLOPs. In this case, AHBF-OKD requires less overall training computation compared to ONE. Note that the target network (Branch4) also outperforms baselines.

Conclusion

In this paper, we propose a novel Adaptive Hierarchy-Branch Fusion Framework for Online Knowledge Distillation (AHBF-OKD), which alleviates homogenization by designing hierarchical branches and an adaptive hierarchy-branch fusion module. We first employ the hierarchical branch structure by gradually adding the auxiliary layers into the target branch to generate diverse features. Then, hierarchical teacher assistants are constructed in the AHBF module by merging the previous TA and the current branch in a recursive manner, where the smaller TA from the previous hierarchy can be distilled from the knowledge of the current TA and hierarchy-branch. Experiments show the proposed AHBF-OKD achieves superior performance on a variety of CNN architectures over different datasets.

Acknowledgements

This work is supported by the National Natural Science Foundation of China (NO. 62102151, NO. 72192821, NO. 62076016), Shanghai Sailing Program (21YF1411200), Beijing Natural Science Foundation (L223024) and CAAI-Huawei MindSpore Open Fund (CAAIXSJLJJ-2021-031A).

References

Allen-Zhu, Z.; and Li, Y. 2020. Towards understanding ensemble, knowledge distillation and self-distillation in deep learning. *arXiv preprint arXiv:2012.09816*.

Berthelot, D.; Carlini, N.; Goodfellow, I.; Papernot, N.; Oliver, A.; and Raffel, C. A. 2019. Mixmatch: A holistic approach to semi-supervised learning. In *NIPS*.

Bucilua, C.; Caruana, R.; and Niculescu-Mizil, A. 2006. Model compression. In *SIGKDD*, 535–541.

Chen, D.; Mei, J.-P.; Wang, C.; Feng, Y.; and Chen, C. 2020. Online Knowledge Distillation with Diverse Peers. *AAAI*, 34(04): 3430–3437.

Chen, P.; Liu, S.; Zhao, H.; and Jia, J. 2021. Distilling Knowledge via Knowledge Review. In *CVPR*, 5008–5017.

Chung, I.; Park, S.; Kim, J.; and Kwak, N. 2020. Feature-map-level online adversarial knowledge distillation. In *ICML*, 2006–2015.

Guo, Q.; Wang, X.; Wu, Y.; Yu, Z.; Liang, D.; Hu, X.; and Luo, P. 2020. Online Knowledge Distillation via Collaborative Learning. In *CVPR*.

He, K.; Gkioxari, G.; Dollár, P.; and Girshick, R. 2017. Mask r-cnn. In *ICCV*, 2961–2969.

He, K.; Zhang, X.; Ren, S.; and Sun, J. 2016. Deep Residual Learning for Image Recognition. In *CVPR*.

Hinton, G. E.; Vinyals, O.; and Dean, J. 2015. Distilling the Knowledge in a Neural Network. *CoRR*, abs/1503.02531.

Huang, G.; Liu, Z.; Van Der Maaten, L.; and Weinberger, K. Q. 2017. Densely connected convolutional networks. In *CVPR*, 4700–4708.

Huawei. 2020. Mindspore. http://www.mindspore.cn/. Accessed: 2022-10-27.

Ji, M.; Heo, B.; and Park, S. 2021. Show, Attend and Distill: Knowledge Distillation via Attention-based Feature Matching. In *AAAI*, 7945–7952.

Kim, J.; Hyun, M.; Chung, I.; and Kwak, N. 2021. Feature fusion for online mutual knowledge distillation. In *ICPR*, 4619–4625.

Krizhevsky, A.; Hinton, G.; et al. 2009. Learning multiple layers of features from tiny images. Technical report, University of Toronto.

Laine, S.; and Aila, T. 2017. Temporal Ensembling for Semi-Supervised Learning. In *ICLR*.

Lan, X.; Zhu, X.; and Gong, S. 2018. Knowledge Distillation by On-the-Fly Native Ensemble. In *NIPS*, 7528–7538.

Li, S.; Lin, M.; Wang, Y.; Wu, Y.; Tian, Y.; Shao, L.; and Ji, R. 2022. Distilling a Powerful Student Model via Online Knowledge Distillation. *IEEE Transactions on Neural Networks and Learning Systems*, 1–10.

Li, Z.; Huang, Y.; Chen, D.; Luo, T.; Cai, N.; and Pan, Z. 2020. Online knowledge distillation via multi-branch diversity enhancement. In *ACCV*.

Li, Z.; Ye, J.; Song, M.; Huang, Y.; and Pan, Z. 2021. Online Knowledge Distillation for Efficient Pose Estimation. In *ICCV*, 11720–11730.

Lin, T.-Y.; Dollár, P.; Girshick, R.; He, K.; Hariharan, B.; and Belongie, S. 2017. Feature pyramid networks for object detection. In *CVPR*, 2117–2125.

Liu, Y.; Cao, J.; Li, B.; Yuan, C.; Hu, W.; Li, Y.; and Duan, Y. 2019. Knowledge Distillation via Instance Relationship Graph. In *CVPR*, 7096–7104.

Micaelli, P.; and Storkey, A. J. 2019. Zero-shot Knowledge Transfer via Adversarial Belief Matching. In *NeurIPS*, 9547–9557.

Mirzadeh, S.; Farajtabar, M.; Li, A.; Levine, N.; Matsukawa, A.; and Ghasemzadeh, H. 2020. Improved Knowledge Distillation via Teacher Assistant. In *AAAI*, 5191–5198.

Ren, S.; He, K.; Girshick, R.; and Sun, J. 2015. Faster r-cnn: Towards real-time object detection with region proposal networks. In *NeurIPS*, volume 28.

Romero, A.; Ballas, N.; Kahou, S. E.; Chassang, A.; Gatta, C.; and Bengio, Y. 2015. FitNets: Hints for Thin Deep Nets. In *ICLR*.

Russakovsky, O.; Deng, J.; Su, H.; Krause, J.; Satheesh, S.; Ma, S.; Huang, Z.; Karpathy, A.; Khosla, A.; Bernstein, M.; et al. 2015. Imagenet large scale visual recognition challenge. *International journal of computer vision*, 115(3): 211–252.

Sandler, M.; Howard, A.; Zhu, M.; Zhmoginov, A.; and Chen, L.-C. 2018. Mobilenetv2: Inverted residuals and linear bottlenecks. In *CVPR*, 4510–4520.

Simonyan, K.; and Zisserman, A. 2015. Very Deep Convolutional Networks for Large-Scale Image Recognition. In *ICLR*.

Sohn, K.; Berthelot, D.; Carlini, N.; Zhang, Z.; Zhang, H.; Raffel, C. A.; Cubuk, E. D.; Kurakin, A.; and Li, C.-L. 2020. Fixmatch: Simplifying semi-supervised learning with consistency and confidence. In *NeurIPS*, 596–608.

Son, W.; Na, J.; Choi, J.; and Hwang, W. 2021. Densely Guided Knowledge Distillation using Multiple Teacher Assistants. In *ICCV*, 9375–9384.

Song, G.; and Chai, W. 2018. Collaborative Learning for Deep Neural Networks. In *NeurIPS*, 1837–1846.

Su, T.; Liang, Q.; Zhang, J.; Yu, Z.; Wang, G.; and Liu, X. 2021. Attention-based Feature Interaction for Efficient Online Knowledge Distillation. In *ICDM*, 579–588.

Van der Maaten, L.; and Hinton, G. 2008. Visualizing data using t-SNE. *Journal of machine learning research*, 9(11).

Wang, W.; Wei, F.; Dong, L.; Bao, H.; Yang, N.; and Zhou, M. 2020. Minilm: Deep self-attention distillation for task-agnostic compression of pre-trained transformers. In *NIPS*, 5776–5788.

Wu, G.; and Gong, S. 2021. Peer Collaborative Learning for Online Knowledge Distillation. In *AAAI*, 10302–10310.

Xie, Q.; Luong, M.-T.; Hovy, E.; and Le, Q. V. 2020. Self-training with noisy student improves imagenet classification. In *CVPR*, 10687–10698.

Yim, J.; Joo, D.; Bae, J.; and Kim, J. 2017. A gift from knowledge distillation: Fast optimization, network minimization and transfer learning. In *CVPR*, 4133–4141.

Zagoruyko, S.; and Komodakis, N. 2017. Paying more attention to attention: Improving the performance of convolutional neural networks via attention transfer. In *ICLR*.

Zhang, Y.; Xiang, T.; Hospedales, T. M.; and Lu, H. 2018. Deep Mutual Learning. In *CVPR*.