Neural Processing Letters (2024) 56:172
https://doi.org/10.1007/s11063-024-11606-z

A Unified Asymmetric Knowledge Distillation Framework for Image Classification

Xin Ye[1] · Xiang Tian[1,3] · Bolun Zheng[2,3] · Fan Zhou[1,4] · Yaowu Chen[1,4]

Accepted: 21 March 2024 / Published online: 10 May 2024
© The Author(s) 2024

Abstract
Knowledge distillation is a model compression technique that transfers knowledge learned by teacher networks to student networks. Existing knowledge distillation methods greatly expand the forms of knowledge, but also make the distillation models complex and symmetric. However, few studies have explored the commonalities among these methods. In this study, we propose a concise distillation framework to unify these methods and a method to construct asymmetric knowledge distillation under the framework. Asymmetric distillation aims to enable differentiated knowledge transfers for different distillation objects. We designed a multi-stage shallow-wide branch bifurcation method to distill different knowledge representations and a grouping ensemble strategy to supervise the network to teach and learn selectively. Consequently, we conducted experiments using image classification benchmarks to verify the proposed method. Experimental results show that our implementation can achieve considerable improvements over existing methods, demonstrating the effectiveness of the method and the potential of the framework.

✉ Xiang Tian
tianx@zju.edu.cn

Xin Ye
11815023@zju.edu.cn

Bolun Zheng
zhengbolun1024@163.com

Fan Zhou
fanzhou@mail.bme.zju.edu.cn

Yaowu Chen
cyw@mail.bme.zju.edu.cn

1 Institute of Advanced Digital Technologies and Instrumentation, Zhejiang University, No. 38 Zheda Road, Hangzhou 310027, Zhejiang, China

2 School of Automatic, Hangzhou Dianzi University, No. 1158, 2nd Street, Baiyang Street, Hangzhou 310018, Zhejiang, China

3 Zhejiang Provincial Key Laboratory for Network Multimedia Technology, Hangzhou 310027, Zhejiang, China

4 Embedded System Engineering Research Center, Ministry of Education of China, Hangzhou 310027, Zhejiang, China

Keywords Model compression · Knowledge distillation · Deep neural networks · Image classification

1 Introduction

Knowledge Distillation (KD) [1] is a well-known model compression technique that has attracted considerable attention in recent years for its wide usage in deep learning applications. Compared to traditional single-model training, knowledge distillation is a procedure that uses a high-performance large model (teacher) to guide a relatively smaller model (student) for more appropriate training, thus transferring the advantage of the teacher to the student with acceptable performance loss. From an optimization perspective, it can be considered a special regularization method [2, 3] that achieves appropriate label smoothing through the predictions generated by real models with reliable performance and acceptable noise, similar to an experienced teacher imparting the knowledge he has mastered to the students.

Regardless of perspective, the key to knowledge distillation is determining what the knowledge is and how to distill it. To answer these two questions, numerous studies give their own model techniques based on knowledge definitions and distillation methods. Knowledge definitions include simple predictions [1, 4], midway feature maps [5–8], and to high-level correlations [9–11]. Besides, the distillation methods have significantly changed from the early two-stage offline type [12] to the present one-stage online type [13].

However, these methods have placed excessive emphasis on performance improvement while neglecting the simplicity of distillation modeling. This has resulted in overly complex distillation methods that are not only incomprehensible in theoretical understanding but challenging to practical deployment. It is difficult to discern the specific contributions of individual components or techniques, limiting our ability to gain insights into the inner workings of the distillation. Meanwhile, the intricate architectures and optimization techniques can introduce computational and memory overhead, making it harder to implement and deploy these methods efficiently on resource-constrained devices or in real-time applications. Therefore, it is crucial to strike a balance between performance gains and the simplicity of knowledge distillation methods.

Our motivation is to rebuild a concise yet effective distillation model that can promote better theoretical understanding, facilitate wider utilization, and perform well in various practical applications. To do so, this paper mainly studies the following two challenging problems. The first challenging problem is the growing model complexity. The distillation model in early studies, such as conventional knowledge distillation [1] was simple and intuitive, with only one teacher network, one student network, and a loss function constructed based on their predictions. However, this simplicity has been undermined in follow-up studies due to the introduction of new knowledge definitions and distillation methods that often call for additional structure and supervision items. Consequently, distillation models have become not as intuitive as before, resulting in difficulties in theoretical understanding and practical applications. In fact, conventional teacher-student knowledge distillation [1] is still widely adopted in applications because of its simplicity.

The other challenge is the implicit model symmetry, which is related to distillation performance. *Symmetry* refers to the interchangeability between the networks in the same model (analogous to the interchangeability between the unknowns that we call algebraic symmetry in a system of equations). Conventional knowledge distillation is not symmetric because the networks in such a model are supposed to be either teachers or students and are completely

different in terms of structure, role, and behavior during training and evaluation, thereby not interchangeable. Recent online and self-distillation studies have proposed to treat networks as learning partners. These networks are no longer restricted to old roles as teachers or students and can teach others while simultaneously learning from others, making them interchangeable, and their interchange does not fundamentally change the distillation model, thereby bringing implicit or potential symmetry to the model. However, this symmetry is not always conducive to knowledge distillation because it also establishes strong correlations between these interchangeable networks, thereby limiting their performance to a close level, whereas knowledge distillation is essentially focused on individual networks that greatly outperform themselves.

In this study, we first propose an abstract framework to unify these existing methods, which reduces the model complexity. Furthermore, we propose a method to construct asymmetric distillation from the deployed network under this framework to explicitly break the symmetry.

Based on existing methods, we first analyze the commonness of various knowledge distillation models, abstracting all types of knowledge providers as *instances* and knowledge transfers as *interactions* to describe distillation tasks in an instance-interaction framework. In Sect. 3.1, we provide specific definitions of the framework. Following these definitions, we can transform the framework into any existing distillation model.

With this framework, we re-analyze these typical models, pointing out that the symmetry of these models conflicts with the asymmetry of task training deployment. To overcome this, we deconstruct the distillation procedure into three steps and propose a method to generate a training instance group from the deployed instance to achieve asymmetric knowledge distillation. Specifically, we discuss the generation of instances and specification of interactions through theoretical derivations and experimental results. Regarding instances, we design more effective shallow-wide branches in conjunction with the multi-stage bifurcation method. For interactions, we ensure that they have a certain level of asymmetry while maintaining simplicity and constructing many-to-one supervision.

To verify the proposed method, we conduct experiments on two benchmarks for classification tasks with appropriate structural adjustments and hyperparameter settings, thereby demonstrating the effectiveness of our framework. For CIFAR-100, our ResNet-56 implementation outperforms the baseline by 4.93%, and ResNet-110 achieves 79.05% with a 5.45% boost, which significantly outperforms other existing methods. On ImageNet-1k, our ResNet-18 achieves a 1.74% improvement over the baseline.

The contributions of this study are as follows:

- We propose an abstract instance-interaction framework to unify the existing knowledge distillation methods to reduce the model complexity.
- We propose an asymmetric method under the framework to construct knowledge distillation and give its pipeline.
- We design a multi-stage shallow-wide bifurcation method to extend a group of training-only instances and a grouping strategy with many-to-one supervision to implement asymmetric interactions.
- We conduct experiments on two benchmark datasets to verify the performance and effectiveness of the proposed method.

2 Related Work

Teacher-Student Knowledge Distillation. Conventional knowledge distillation [1] established a base knowledge distillation model between a teacher network and a student network,

that is teacher-student distillation. This is considered offline learning because the training of the teacher and student networks occurs separately in two phases: the teacher network is trained in advance to ensure that it contains knowledge, and then the student network learns from the teacher's prediction. Compared with traditional single-network training, teacher-student distillation has achieved significant performance improvement, but it also largely increases training costs since more time and space are needed, especially for teacher preparation. Based on this, some studies [5, 6] have suggested that feature representation in intermediate layers also contains knowledge and proposed feature-based distillation, as opposed to previous prediction-based distillation. Unlike their predictions, the feature map outputs by teacher and student networks tend to have different shapes, thus requiring operations or structures for shape alignment before distillation. More studies [14–16] have focused on learning feature representations, where student networks attempt to better match the teacher's hidden knowledge in feature maps through more complex and well-designed structures. For example, an extra paraphrasing module was applied to transfer feature maps in [17]. Moreover, Jacobian matching [18] and singular value decomposition [19] have also been adopted to improve knowledge transfer. These feature-based methods do indeed enrich the types of knowledge, while the feature selection and matching heavily rely on empirical prior or experimental exploration. Additionally, it will introduce additional parameters and computational burden during feature matching. In fact, the differences in the structure complexity between the teachers and students lead to disparities in their feature representation capabilities, making it exceedingly challenging to compel the students to mimic the teachers' feature representations. Given the inherent challenges, recent studies have employed abstract correlations to convey high-level knowledge. In contrast to features, the selection and transmission of relationships offer relatively more flexibility. The correlation, encompassing structural information [9, 20–22], activation [23], attention [10, 24], mutual-information [25] and disturbance response consistency [11, 26, 27], are more easily acquired by student networks. However, their supervision is comparatively weaker than that of feature representation or predictions, necessitating careful handling during the distillation process and combination with other forms of supervision.

We notice that these teacher-student distillation methods can be broadly classified into three categories based on the knowledge types: prediction-based, feature-based, and relation-based methods, each exhibiting different levels of supervisory strength. Existing methods tend to overemphasize their type differences and ignore their intrinsic connections. In contrast, we adopt a unified framework to describe these three knowledge sources as the same entity, thereby attributing their observed differences in distillation to a more in-depth analysis of the underlying structural factors. This enables us to thoroughly investigate their inherent connections and construct better knowledge providers and transferring, thereby reducing the reliance on teachers and lowering training costs.

Online Mutual Learning. Recently, some studies have improved the basic two-stage offline distillation to one-stage online distillation by replacing teacher networks with learning partners, thus proposing a mutual learning model [13]. This improvement effectively reduces the training cost because, unlike teacher networks that call for an extra training phase of preparation, learning partners are trained together with student networks in one phase, which makes the training more efficient and compact. On the other hand, the knowledge provided by learning partners is not as reliable as that of pre-trained teachers, which limits the performance of the distillation. Meanwhile, as mentioned in [13], the benefits derived from increasing the number of learning partners are not as substantial as initially anticipated; instead, they exhibit a rapidly diminishing marginal return. To mitigate this problem, some studies have focused on using existing learning partners to synthesize stronger supervision

because distillation members no longer belong to the one-teacher-to-one-student type, but have derived many-to-one or even many-to-many types [28–30]. In [28] a simple ensemble method was adopted to map multiple weak networks onto a stronger network, and more effective ensemble methods have been further discussed in [31]. In [29], a two-level distillation was designed to use diverse peers to guide the group leader. Others try to introduce new supervision, in [32], knowledge hidden in filters is measured by information entropy and transferred. These methods strengthen the supervision capabilities of learning partners to a certain extent, but complicate the distillation model and increase the training burden as well.

It can be found that these methods have mainly focused on the utilization of partner networks, while consistently lacking in-depth discussions regarding their sources and construction. Besides, they have not adequately recognized and addressed the issue of modeling symmetry introduced by mutual learning. In our proposed method, the partner networks are no longer mere replicas of the student networks; instead, they undergo a more meticulous construction. Furthermore, we conducted a thorough analysis of modeling symmetry to construct asymmetric knowledge distillation that effectively mitigates performance limitations.

Self-supervised Distillation. Moreover, recent studies have adopted self-distillation to generate supervision signals from the network itself, rather than from teachers or learning partners. In [33], a self-distillation model was proposed and combined with deeply supervised networks [34] to better supervise a network using its intermediate discriminating information via extra auxiliary classifiers. In [35], label smoothing was achieved using this extra self-supervised information to improve performance. In [36], feature refinement was introduced to generate and utilize supervised information at the feature level. These self-distillation methods can avoid the use of more complex models and eliminate the need for generating pseudo-labels through clustering or meta-computing steps, but they also have some limitations. On the one hand, these self-distillation methods rely heavily on additional auxiliary structures or augmented inputs [37, 38] to ensure sufficient distinctive knowledge information for effective supervision. On the other hand, the use of auxiliary structures needs to be carefully considered, as larger auxiliary structures can make self-distillation equivalent to a mutual learning method with shared shallow network layers. Existing methods are often constrained by this contradiction and do not thoroughly explore the relationship between self-distillation and mutual learning, as well as the role of auxiliary structures in achieving effective self-supervision. In this study, we address these limitations by simultaneously incorporating both mutual learning and self-distillation within our framework. This approach allows us to effectively reconcile the contradiction between auxiliary structures and the student network and facilitates more comprehensive research and discussions in this area.

New methods have emerged continuously as network structures and supervision items become increasingly complex. Meanwhile, the high differentiation abilities of methods do not promote the core theory of knowledge distillation well, although the distillation performance is highly improved.

Our proposed framework is highly compatible with these methods and unifies them as knowledge extraction and transfer procedures from the framework perspective, and thus a unified platform for discussion and comparison is provided. The proposed asymmetric distillation method is re-modeled and re-derived under this framework and integrates the advantages of various method types. Compared with the teacher-student model, it utilizes single-stage training. Additionally, it uses asymmetric optimization objectives to solve the performance bottleneck caused by the online mutual learning model. Meanwhile, as an extension of self-distillation, the generation of various types of knowledge as supervision signals,

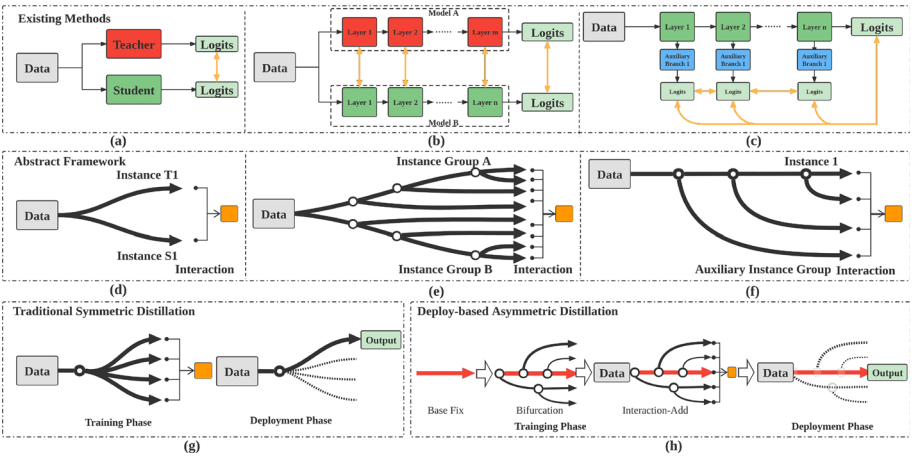

Fig. 1 An overview of our framework and method. Take some typical knowledge distillation methods as examples: **a** response-based teacher-student distillation; **b** feature-based mutual learning; and **c** self-distillation. Their corresponding models under our abstract framework are shown in (**d**, **e**, **f**). We abstract all network flows for extracting knowledge as instances and knowledge transfers as interactions. The traditional symmetric knowledge distillation construction procedure is shown in (**g**). In contrast, we propose an asymmetric knowledge distillation method based on a deployed network and discuss bifurcation and interactions as shown in (**h**)

corresponding auxiliary structure characteristics, and supervision grouping strategies are discussed in detail.

3 Methodology

In this section, we aim to view the existing knowledge distillation methods from a unified perspective, propose a simple but effective abstract framework, re-analyze the construction process of knowledge distillation using the proposed framework, and provide an asymmetric knowledge distillation construction method from the deployed network. Further details of the framework are discussed in subsequent sections.

An overview of the proposed framework and method is presented in Fig. 1.

3.1 Unified Knowledge Distillation Framework

First, we provide a general description of the knowledge distillation framework and extend its settings for further discussion.

Suppose that the basic task is a K-class image classification task with an annotated dataset $\boldsymbol{D} = \{\boldsymbol{x}_i, \boldsymbol{y}_i\}$, where $\boldsymbol{x}_i \in \mathbb{R}^{C\times H\times W}$ is the i-th image in the dataset, and $\boldsymbol{y}_i$ is the ground truth label. We employ a knowledge distillation method, whatever it is, with N ($N \geq 1$) networks, named $\boldsymbol{F}_1, \boldsymbol{F}_2, \cdots, \boldsymbol{F}_N$ [1]. The input to the network is an image $\boldsymbol{x}$, and the outputs are $\boldsymbol{f}_1, \boldsymbol{f}_2, \ldots, \boldsymbol{f}_N \in \mathbb{R}^K$, respectively, by the following formula representation:

$$\boldsymbol{f}_i = \boldsymbol{F}_i\left(\boldsymbol{x}; \boldsymbol{\theta}_i\right) \tag{1}$$

[1] Here, $\boldsymbol{F}$ is not only the symbol of the network but also the network-fitted nonlinear transformation.

, where $\boldsymbol{\theta}_i$ is the parameter of $\boldsymbol{F}_i$ and $\boldsymbol{\theta}_{total}$ is the union of all $\boldsymbol{\theta}_i$.

Since classification tasks prefer more discrete results, we can convert $\boldsymbol{f}_i$ into discrete probability distributions $\boldsymbol{p}_i \in \mathbb{R}^K$ using the $SoftMax$ function as follows:

$$\boldsymbol{p}_i = SoftMax(\boldsymbol{f}_i). \tag{2}$$

Specifically, the probability of the j-th class predicted by the i-th networks, that is, $\boldsymbol{p}_i^j$, can be calculated as follows:

$$\boldsymbol{p}_i^j = \frac{\exp(\boldsymbol{f}_i^j/T)}{\sum_{c=1}^{K}\exp(\boldsymbol{f}_i^c/T)}, \tag{3}$$

where T is a temperature parameter used to control the smoothness.

For training, every network $\boldsymbol{F}_i$ is given a corresponding loss function L_i, and the entire loss function L_{total} can be constructed by simply summing L_i. Once L_{total} is constructed, optimization algorithms, such as SGD [39] and its variants, are employed to approximately reach the optimization target by making L_{total} converge to the local minima. Then, during deployment, the network with the best performance is selected to represent the rest.

However, this network-based description encounters certain problems when promoted, the most serious of which is the ambiguity of the distillation subject. In general, networks are set to hold totally independent parameters as the case (a) in Fig. 2; however, they can also share their parameters in some cases. In fact, quite a few studies [28, 31, 40–42] have mentioned or adopted such shared cases, and some studies, such as self-distillation [33] and deeply supervised knowledge synergy (DSKS) [43], even take parameter sharing as the core design or special motivation and demonstrate the benefits of doing so, such as higher performance and lower training costs. Together with these benefits, a new problem arises: blurring the conceptual boundaries of networks. As shown in Fig. 2, the sharing case (b), in which two networks share parameters, can also be interpreted as the case in which a special network outputs two predictions simultaneously in one pass, and the nature of this structure, whether it represents a single network or two shared networks, remains ambiguous in the absence of explicit specification. In other words, the term "*network*" may no longer be adequate to accurately describe the subject in distillation models because we cannot determine which part it actually refers to, especially in self-distillation model.

Therefore, we propose the concept of *instances* instead of *networks*. An instance is an individual knowledge provider that is not only related to the network structure but also to the extraction and transfer of knowledge. It undertakes the forward inference task with a complete network flow path from receiving input $\boldsymbol{x}$ to outputting specific knowledge, such as a feature map or a prediction $\boldsymbol{f}_i$ in classification, and thus inherits the symbol $\boldsymbol{F}$. Put simply, one instance takes one input and transforms it into the expected output.

Then, we can effectively describe the sharing cases by instances; for example, the case (c) in Fig. 2 is an instance-based description that contains two instances whose network flow paths may overlap on some shared parts but eventually separate into two different parts. According to this overlapping property, we divide each instance into three components. We call the position at which two instances separate the *bifurcation point*, the shared part before the bifurcation point the *trunk*, and the independent part after the bifurcation point the *branch*, as shown in Fig. 2. In particular, two completely independent networks can be regarded as two instances whose bifurcation points are located at the beginning of their networks, with no shared trunk.

Then, we discuss the creation and removal of instances. Since the network model is translated into the instance model, N is the number of instances, which depends on the

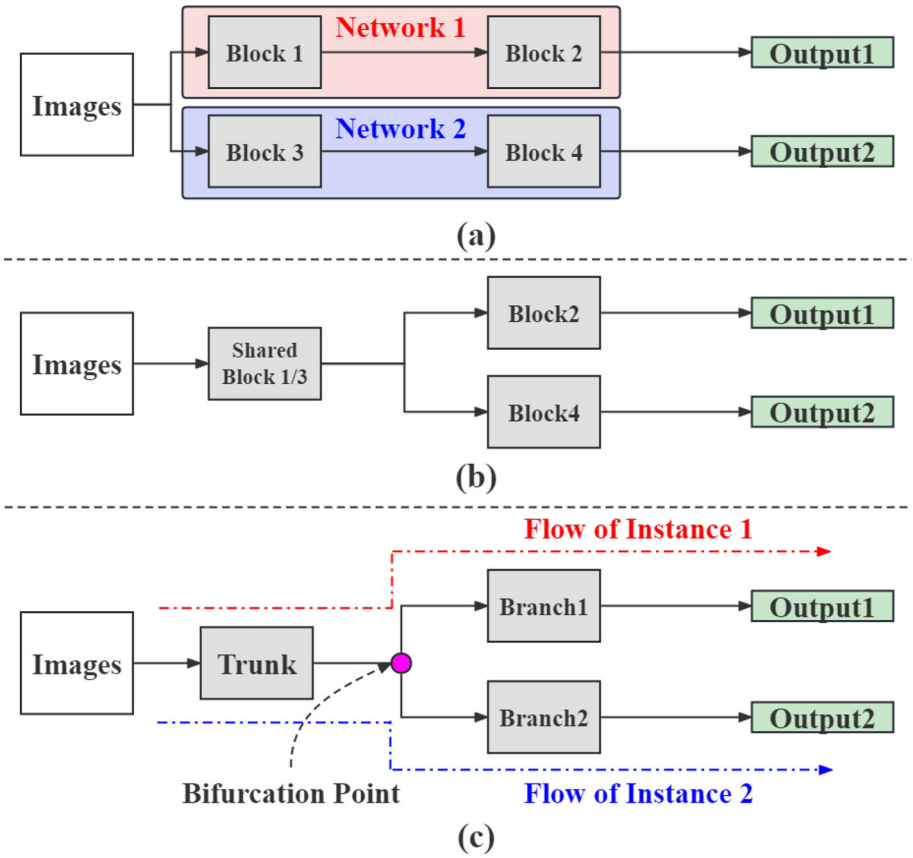

Fig. 2 From networks to instances. **a** two networks that hold totally independent parameters. **b** a sharing case of (**a**), but can also be regarded as a special network that outputs two predictions. **c** two instances with one *trunk*, one *bifurcation* point and two *branches*

number of knowledge extractions. When $N = 1$, the only instance is the network itself. As N increases, new instances are extended by identifying their bifurcation points and branches. Meanwhile, removing an existing instance erases the bifurcation point and its branches.

Furthermore, we discuss the optimization of the instances. Different instances can often extract different forms of knowledge. Therefore, existing methods usually treat them differently in the model, ignoring their unity under a bottom-up design. At the top level, we describe the possible mutual function between these instances through different indicator functions (e.g., distance, similarity, and correlation) and attempt to use optimization methods to make the functions converge in the expected direction. At the bottom level, what we call knowledge is actually the rich semantic information contained in the instance output, which is always a series of matrices. Knowledge transfer is also realized by a series of mathematical operations on these matrices. Since there is no difference among these instances in optimization from the perspective of calculation, for simplicity, we refer to the mutual function between instances as *interactions* to hide complex implementation and optimization details such as feature cosine similarity or prediction cross-entropy loss. Thereby, interactions are unified

operations, such as connecting lines and arranging a series of instances to build connections to promote the optimization of instances.

In summary, we provide a unified distillation framework described by instances and interactions that is compatible with a simple teacher-student model, online correlative learning with multiple partners, and a self-distillation model. Using this framework to translate these models into instance-interaction descriptions makes it possible to place them on the same ground, which is conducive to our further discussion.

3.2 Asymmetric Knowledge Distillation

As mentioned above, existing online distillation models often hold implicit symmetry, and this symmetry is more manifested in the interaction under the unified framework. A common interaction involves building a loss function L_i for instances in optimization. L_i is a metric of the instance's output $\boldsymbol{f}_i$ or $\boldsymbol{p}_i$ and varies in its form for different optimization targets and supervision intensities. For example, by following the principle of deep mutual learning (DML) [13], we can provide a possible form of L_i as

$$L_i = L_{CE}\left(\boldsymbol{p}_i, \boldsymbol{y}\right) + \sum_{j=1, j\neq i}^{N} L_{KL}\left(\boldsymbol{p}_i || \boldsymbol{p}_j\right), \tag{4}$$

where L_{CE} represents the cross-entropy loss between the SoftMax output $\boldsymbol{p}$ and ground-truth label $\boldsymbol{y}$ for input $\boldsymbol{x}$, and L_{KL} is the Kullback–Leibler (KL) divergence loss between two probability distributions $\boldsymbol{p}_i$ and $\boldsymbol{p}_j$ together with a temperature compensation item T^2, which is finally formulated as

$$L_{KL}\left(\boldsymbol{p}_i || \boldsymbol{p}_j\right) = T^2 \sum_{c=1}^{N} p_i^c \log \frac{p_i^c}{p_j^c}. \tag{5}$$

The final loss function for the entire model sums these L_i as

$$L_{total} = \sum_{i=1}^{N} L_i = \sum_{i=1}^{N} L_{CE}\left(\boldsymbol{p}_i, \boldsymbol{y}\right) + \sum_{i=1}^{N} \sum_{j=1, j\neq i}^{N} L_{KL}\left(\boldsymbol{p}_i || \boldsymbol{p}_j\right) \tag{6}$$

We find that such a model is symmetric because all instances are interchangeable in the loss function L_{total}, which means that all instances are equivalent, not only in the training procedure but also in the deployment selection. During training, the parameters are constructed in a target function $\boldsymbol{R}$ as a whole, which means that they are optimized together as follows:

$$\boldsymbol{\theta}_{total}^{*} = \arg\min_{\boldsymbol{\theta}_{total}} \{\boldsymbol{R}(\boldsymbol{\theta}_{total}, \boldsymbol{D})\}. \tag{7}$$

During deployment, one of them is selected randomly or deliberately by an evaluation function $\boldsymbol{E}$ as follows:

$$\boldsymbol{\theta}_{best} = \arg\min_{\boldsymbol{\theta}_i} \{\boldsymbol{E}(\boldsymbol{\theta}_i, \boldsymbol{D}) | i = 1..N\} \tag{8}$$

This strong constraint significantly impairs the diversity among instances, making their behavior and performance similar, and hence restricted. To illustrate this, we conducted a simple experiment on the distillation scale, which refers to the number of networks involved in the distillation. In the experiment, we selected ResNet-56 for the independent instances

Table 1 CIFAR-100 top-1 accuracy of instances under different distillation scales

Scale	Ave. Acc (%)	Ens. Acc (%)
2	74.85	76.53
3	75.00	77.38
4	75.33	77.68
5	75.21	77.69
6	75.04	77.24
7	74.77	76.99

The 'Scale' refers to the number of networks involved in distillation, and their average and ensemble accuracies are presented in the second and third columns, respectively

to construct the distillation implementation procedure according to the above model and gradually expanded the distillation scale from two to seven. For validation, we evaluated the average accuracy of the instances and their ensemble accuracy using CIFAR-100. The experimental results shown in Table 1 corroborate our viewpoint because adding more instances to this model did not achieve much improvement in either their individual performance or their ensemble accuracy. It is not difficult to imagine that when the constraints imposed by interactions are too strong, the instances become identical and have nothing to learn from each other, leading to distillation failure.

In contrast to this model symmetry, the distillation task itself has potential and natural asymmetry in terms of training deployment. In the training phase, all instances and their parameters participate, but only a subset of them is selected for deployment, which is also reflected in the early teacher-student model. Inspired by this, we find that explicitly emphasizing this asymmetry by determining the deployed instance in advance simplifies the task. This is because the deployment selection becomes redundant, and the optimization target in the training phase becomes clearer. Intuitively, rather than obtaining a series of slightly better instances, we prefer to obtain **ONE** instance that significantly exceeds its original performance. Once the deployed instance is determined, the optimization target is to make it learn more from others and outperform them. The construction of the interactions should also follow this principle.

To achieve this goal, we propose an asymmetric knowledge distillation procedure that explicitly determines the final deployed instance $\boldsymbol{F}_d$ and distinguishes it from others by constructing asymmetric interactions. From the perspective of parameters, all parameters are optimized during training, but only $\boldsymbol{\theta}_d \subseteq \boldsymbol{\theta}$, which represents the parameter of $\boldsymbol{F}_d$, presents in deployment. We slightly modify Eq. 8 as follows:

$$\boldsymbol{\theta}_d^* = \arg\min_{\boldsymbol{\theta}_d} \boldsymbol{R}'(\boldsymbol{\theta}_d, \boldsymbol{\theta}_e, \boldsymbol{D}), \tag{9}$$

where $\boldsymbol{R}'$ represents the loss function that explicitly treats the parameter groups differently, and $\boldsymbol{\theta}_e = \boldsymbol{\theta} - \boldsymbol{\theta}_d$.

Combined with the instance-interaction framework, we further divide the asymmetric knowledge distillation procedure into three steps, as shown in Fig. 3:

1. Initialize N to 1 by setting up an instance $\boldsymbol{F}_d$ that is set to be the deployed one;
2. Increase N properly by the instance extension and arrange interactions to perform distillation;
3. Decrease N back to 1 by the instance removal, leave only $\boldsymbol{F}_d$ for deployment.

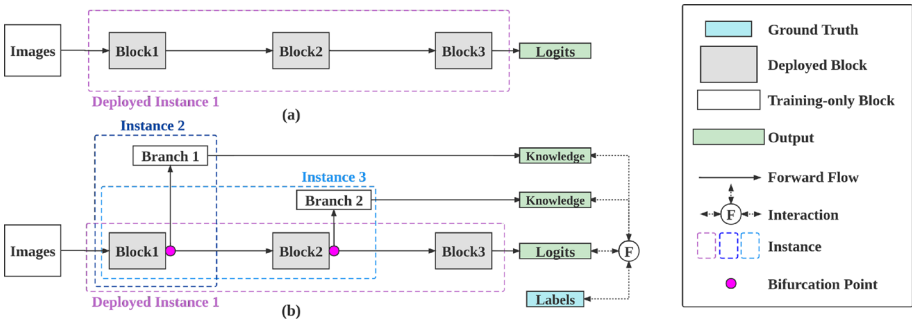

Fig. 3 Construction of asymmetric knowledge distillation. (1) Initialize N to 1 by setting up an instance $\boldsymbol{F}_d$ that is nominated to be the final deployed one as (**a**); (2) Increase N properly by the instance extension and add the interactions to construct and perform distillation, that is, from (**a**) to (**b**); (3) Decrease N back to 1 by the instance removal, leaving only $\boldsymbol{F}_d$ for deployment, that is, from (**b**) to (**a**)

Given that $\boldsymbol{F}_d$ is predetermined, steps 1 and 3 are considered standard and require no further elaboration on the design details. In this manner, we transform the original problem of knowledge distillation into two subproblems under the framework of instance extension and interaction arrangement, which are explained in detail in the following two sections.

3.3 Multi-stage Shallow-Wide Bifurcation

The first subproblem involves the principle of instance extension, specifically, determining the type of training-only instances to extend in the asymmetric knowledge distillation framework. This extension is performed by selecting their bifurcation points in existing structures (which also determines their trunks) and supplementing them with extra new branches. As mentioned previously, bifurcation points and branches are critical to instances because they constitute instances together with trunks and have a direct influence on the instances' performance and knowledge distillation. Therefore, this subproblem is also equal to appropriate bifurcation point selection and branch supplementation.

To solve the first subproblem, we propose a multi-stage shallow-wide bifurcation method on instances composed of multi-stage bifurcation point selection and a shallow-wide branch supplement.

Multi-stage bifurcation point selection means that we select these bifurcation points from different stages of the deployed instance $\boldsymbol{F}_d$ for two reasons. First, since the same bifurcation points and similar branches strengthen the model symmetry, the positions of the bifurcation points should be different to avoid this issue. Second, the selection should adapt to the mainstream network structure. At present, the mainstream network structures, such as ResNet [44], Wide ResNet (WRN) [45], MobileNet [46], and DenseNet [47], often have multi-stage designs, meaning that the network is divided into multiple stages in series. This multi-stage design is often accompanied by downsampling and channel widening, making the feature maps (i.e., the network midway outputs delivered between stages) lower in spatial resolution and broader in the channel, and the contained information gradually abstracted and complex [48]. Inspired by deep supervised nets (DSN) [34], we adopted a multi-stage bifurcation point selection, setting the bifurcation points at the connection between each of the two stages to obtain midway outputs in different stages. These midway features vary in resolution and

Table 2 CIFAR-100 top-1 accuracy of ResNet-56 with different mutual learning partners

Partner	Partner Acc (%)	ResNet-56 Acc (%)
–	–	72.20 (baseline)
ResNet-56	74.87	74.87
ResNet-110	76.47	74.76
WRN-16-2	75.43	75.65
WRN-22-8	81.61	74.53
MobileNetV2 [53]	75.20	74.39

The first row of results represents the baseline of ResNet-56 that is normally trained

channel, and are considered to contain different semantic information, which is naturally differentiated and beneficial for asymmetric distillation.

Shallow-wide branch supplementation involves supplementing multi-stage bifurcation points with shallow but wide structures as branches, which is slightly different from recent studies. Recent studies tend to employ deep structures because depth[2] is an important concept in current network architecture studies. According to widely recognized design principles, the network structure should be deep and sufficiently complex to ensure a good performance because it is no longer limited by the gradient problem caused by depth. Insufficient depth leads to poor nonlinear fitting ability, which has been proven to be a major reason why early networks with shallow structures (e.g., AlexNet [49] and VGG [50]) did not perform well in previous studies [34, 51].

Although the depth is important, it is not everything. In fact, increasing the depth does not always lead to ideal performance improvements, which is also empirically proved by the fact that ResNet-1202 is much deeper than ResNet-110 but the latter outperforms it on the CIFAR-100 dataset [44]. Moreover, based on an analysis of existing methods and further experiments, it is even harder to use deeper and more complex structures to obtain better distillation results. In the experiment shown in Table 2, we let Resnet-56 networks perform mutual learning with different partners following the settings of DML and evaluate their classification accuracy on the CIFAR-100 dataset. Consequently, although these partners vary in depth and performance, there is no significant difference between the distillation results, indicating that simply increasing the depth is not feasible in branch design.

This infeasibility is due to the fact that the original task performance of an instance is not necessarily synonymous with its capability as a teacher in knowledge distillation. Therefore, original task performance is not the design goal for training-only instances either. Instead, the real goal should be to strengthen training-only instances in terms of knowledge extraction and transfer ability and to further improve the performance in teaching $\boldsymbol{F}_d$. In extreme cases, the ground truth label can be regarded as the output of an instance whose accuracy is always 100%, which is considered too difficult for a student network to learn [52].

Inspired by the effectiveness of wide residual networks [45] and previous study on the efficacy of super-wide 1x1 convolutions in neural networks [54], we design a shallow-but-wide branch structure, as shown in Fig. 4. It is a two-level bottleneck-like structure consisting of intra- and inter-block bottlenecks. Within a block, we adopt a bottleneck structure similar to that proposed in [44] with two 1×1 convolutions to reduce/restore channels and 3×3 convolutions inserted in-between, while another 1×1 convolution is always applied to connect

[2] Depth is the number of layers in a structure, while width refers to the number of channels in each layer and its feature map.

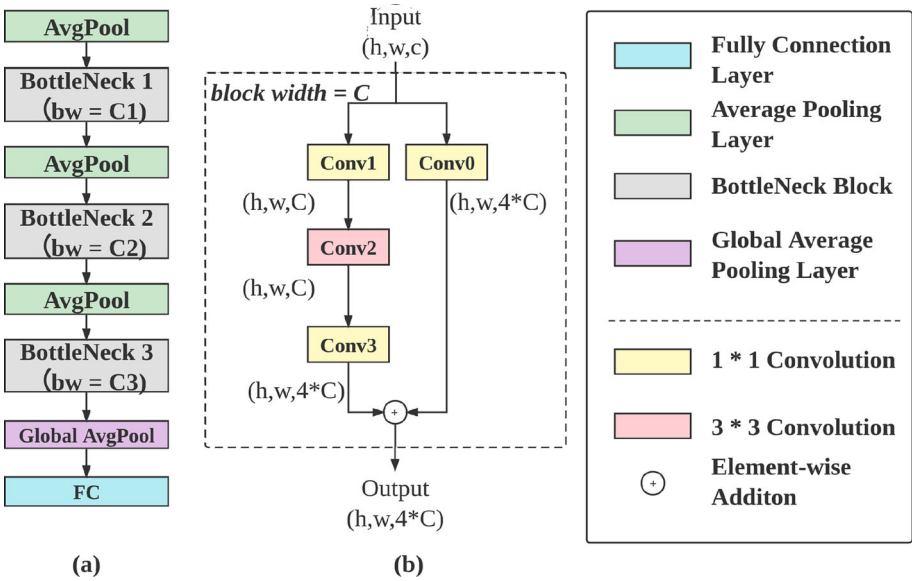

Fig. 4 Shallow-Wide branch structure. **a** a three-bottleneck shallow-wide branch marked as $\{C_1, C_2, C_3\}$; **b** the detailed structure of a bottleneck block with the block width (bw) C

the input and the output as a shortcut, and we define the block width as the output channel size of the 3×3 convolution layer. Among the blocks, we choose a narrowed-after-widened form in the channels instead of stacking the same or gradually wider blocks. The channels of the feature maps are significantly widened by the first two blocks and narrowed back to an appropriate level by the last block, which constitutes the inverse-bottleneck design between the blocks.

To perform downsampling spatially without stridden convolutions, average pooling is added in front of each block, which can be regarded as spatial feature fusion. With these blocks stacked, the spatial resolution of the feature maps gradually decreases, and the channels first increase and then decrease. The features that have undergone multiple aggregations become more robust and expressive and are translated into the output of the instance by a fully connected layer. In such a structure, the 1×1 convolutions play an important role in repeatedly aggregating and mapping features, thus enhancing the point-wise feature representation in the spatial dimension and providing a direct connection for the gradient between the input and output.

Based on the multi-stage bifurcation points and shallow-wide branches, we can obtain a multi-stage bifurcation instance group containing various instances with different trunks and branches to generate knowledge for more effective supervision.

3.4 Loss Function

The second subproblem involves the principle of interaction arrangement, that is, how to arrange interactions to make better use of these instances for knowledge distillation. As mentioned earlier, interactions are essentially mutual functions between instances whose final forms are loss functions based on matrix operations with related optimization methods. Therefore, this subproblem is equivalent to providing the appropriate loss function to present the optimization target and depict the interactions between instances.

To fit the framework, an appropriate loss function is definitely asymmetrical, and we attribute this asymmetry to two aspects of the loss function design. One aspect is the asymmetry between the deployed and training-only instances. For the deployed instance, the main task in training should be to learn as much as possible from the other instances. The rest, which are only present in the training phase, need to play the role of good knowledge providers. Through limited learning, they shall achieve appropriate performance and better expressiveness while preserving relative independence to better provide the various types of knowledge they have learned. In other words, the knowledge transfer flow among instances is in a many-to-one form, in which one learns more than it teaches, whereas others teach more than they learn.

The other aspect is the asymmetry among the training-only instances. Existing methods generally adopt two methods to treat teacher networks: completely independent [1, 13] or integrated into one [29, 31], both of which are symmetric. To avoid this issue, we need to make these instances function differently according to their capability, which means that some teach more, whereas others teach less.

Based on these two aspects, we can provide feasible two-level supervision and the corresponding loss function form, in which all instances are under the supervision of the ground truth label $\boldsymbol{y}$ by cross-entropy loss to ensure their basic validity, while the deployed instance $\boldsymbol{F}_d$ is further supervised by all training-only instances, which is depicted by the KL divergence loss.

In particular, for the training-only instances, instead of utilizing them directly, we group them and then apply the *ensemble* method to each group to obtain several supervision sources. The *ensemble* method is a technique widely used in machine learning to integrate multiple weak predictions into a stronger one. Although it has also been widely adopted in recent studies on knowledge distillation, the problem is that in knowledge distillation, the predictions obtained by integration are sometimes not reliable. The experiments in [13] have indicated that an ensemble product integrated from too many instances may grow too strong such that its probability of peaking at the true class is close to the ground truth label, thus hindering its expressiveness and effectiveness in supervision. Therefore, we use grouping to control the intensity and scale of the ensemble subtly.

Moreover, grouping renders the instances in different groups no longer interchangeable because the function of an instance in the ensemble depends on the performance and number of other members in the same group. For example, the function of an instance may differ when considered in isolation versus within a group of six instances. When an instance is alone, it can directly provide all the knowledge it has learned to the deployed instance, whereas in a group, its knowledge is comprehensively adjusted and optimized by other group members before being transferred. This can lead to better performance in knowledge distillation, as the group members can complement each other's knowledge and help eliminate any weaknesses or errors that may exist in individual instances. Through this ensemble-after-grouping strategy, we can make instances in different groups contribute to supervision differently, and thus asymmetric in the loss function (or abstractly speaking, interactions).

Regarding the grouping principle, we provide a simple yet effective grouping strategy in which instances are divided into two groups: those who share trunks with $\boldsymbol{F}_d$ as a group $\boldsymbol{S}_a$ and the rest as another group $\boldsymbol{S}_b$. We then adopt the mean ensemble method, which is formulated as follows:

$$\hat{\boldsymbol{p}} = \frac{1}{n} \sum_{\boldsymbol{F}_i \in \boldsymbol{S}} SoftMax(\boldsymbol{F}_i(\boldsymbol{x})), \tag{10}$$

where n is the number of instances in the set. In contrast to the common integration using $\boldsymbol{p}_i$, integrating $\boldsymbol{f}_i$ first and then calculating $\boldsymbol{p}_i$ can make it less sharp, thus alleviating the sharp peak problems. More details and discussions of the grouping strategies can be found in the extra study section.

After we obtain $\hat{\boldsymbol{p}}_a$ and $\hat{\boldsymbol{p}}_b$ calculated by formula (Eq. 10), we use them as soft labels to supervise $\boldsymbol{F}_d$ by adding their KL divergence with $\boldsymbol{p}_d$ to the loss function.

In summary, the final loss function is as follows:

$$L_{total} = \sum_{i=1}^{N} L_{CE}\left(\boldsymbol{p}_i, \boldsymbol{y}\right) + \alpha L_{KL}\left(\boldsymbol{p}_d || \hat{\boldsymbol{p}}_a\right) + \beta L_{KL}\left(\boldsymbol{p}_d || \hat{\boldsymbol{p}}_b\right) \tag{11}$$

where α and β are the two hyperparameters for the weight trade-off.

4 Experiments

We selected two common image classification benchmarks to evaluate the effectiveness of the proposed method: CIFAR-100 and ImageNet-1k. The details of the datasets, experimental settings, results, and supplementary studies are as follows.

4.1 Experiments on the CIFAR-100 dataset

CIFAR-100 [55] is a widely used dataset containing 60k images drawn from 100 classes with 50k for training and 10k for testing; the image size of each picture is 32×32 pixels.

To perform the experiments on this dataset, we selected ResNet, a widely used network architecture proposed in [44], as the deployed network $\boldsymbol{F}_d$. We employed ResNet for the CIFAR dataset, adopted a three-stage architecture, further declared two extra instances $\boldsymbol{F}_1$ and $\boldsymbol{F}_2$ using the multi-stage bifurcation method, and constructed a three-instance implementation called *Ours-S*. For the branches, we chose {32, 64, 16}[3] and {64, 128, 32} as the shallower and deeper ones, respectively. Then, we declared a new instance $\boldsymbol{F}_3$ that had the same structure but no shared trunk with $\boldsymbol{F}_d$ and further bifurcated it into two extra instances $\boldsymbol{F}_4$ and $\boldsymbol{F}_5$. Finally, we constructed a six-instance implementation *Ours-M*. Similarly, by repeating the above steps, we obtained a nine-instance implementation *Ours-L*.

Following the grouping strategy, $\boldsymbol{S}_a$ included two instances, and $\boldsymbol{S}_b$ included 0, 3, and 6 instances.

The models were trained for 200 epochs with a starting learning rate of 0.1, which was further divided by 10 at 100 and 150 epochs. We adopted the stochastic gradient descent (SGD) method as the optimizer and set the weight decay and momentum to 0.0005 and

[3] For convenience, we use a list of numbers enclosed by a pair of braces to represent this branch structure, where the length of the list is the number of bottleneck blocks, and each number represents the corresponding bottleneck block width.

Table 3 Top-1 accuracy (%) comparison on the CIFAR-100 dataset

Method	ResNet-20	ResNet-32	ResNet-56	ResNet-110	WRN-16-2	WRN-16-8
Baseline	69.02	70.86	72.20	73.60	72.53	79.49
KD	70.82	73.36	75.18	75.88	74.89	81.13
DML	70.69	72.97	74.87	75.89	74.91	81.32
KDCL-3	71.13	74.16	75.69	76.63	75.92	81.61
ONE	70.80	73.05	74.89	75.46	74.92	80.59
OKDDip	71.12	73.45	75.13	76.38	75.08	81.00
FKT	70.58	73.22	74.97	76.01	75.66	80.99
Ours-S	<u>71.90</u>	74.54	76.65	78.11	76.09	81.94
Ours-M	71.89	<u>74.85</u>	<u>77.00</u>	<u>78.78</u>	<u>76.66</u>	<u>82.43</u>
Ours-L	**72.06**	**74.97**	**77.13**	**79.05**	**76.80**	**82.47**

The compared results of ONE and OKDDip are reproduced by us using the author-provided codes, and others are produced by our implementation according to the relevant papers. All results represent the average of six runs. The best one of each column is in **bold**, and the second best is <u>underlined</u>

0.9, respectively; the batch size was 128. We adopted simple image preprocessing methods, namely random cropping with padding and random flipping, following [44]. For the hyperparameters, we set $\alpha = 2$, $\beta = 2$ and $T = 3$.

We compared our method with some classical and state-of-the-art online knowledge distillation methods, including deep mutual Learning (DML) [13], knowledge distillation via collaborative learning (KDCL) [31], on-the-fly native ensemble (ONE) [28], online knowledge distillation with diverse peers (OKDDip) [29] and filter knowledge transfer (FKT) [32]. Moreover, we provide the results of conventional knowledge distillation (KD) [1] with a normally pre-trained teacher network (with the same structure as the student network $\boldsymbol{F}_d$) at temperature $T = 3$. The implementations of ONE and OKDDip were based on the codes provided by their authors, while the other methods were reproduced by us according to previous studies. For persuasiveness, we selected well-matched settings for the hyperparameter and ran every model six times to report the average. KDCL, ONE, and OKDDip were implemented using three auxiliary networks. In particular, in the implementation of KDCL, we reproduced the MinLogit version proposed in [31]. We also conducted experiments on WRN [45] to verify the effectiveness of the wider basic network. Two basic networks WRN-16-2 and WRN-16-8 were selected. The results presented in Table 3 and Fig. 5 suggest that the traditional KD algorithm with only one teacher can achieve good performance after temperature adjustment, which indirectly confirms our analysis of the bottlenecks in online knowledge distillation methods. Furthermore, our proposed distillation method, despite being an online method, still outperforms existing online knowledge distillation methods and KD, demonstrating the effectiveness of asymmetric distillation.

Specifically, we conducted more experiments to compare our proposed method with more teacher-student methods such as FitNet [5], attention transfer (AT) [24], contrastive representation distillation (CRD) [26], strong teacher (DIST) [56], decoupled knowledge distillation (DKD) [57], knowledge Review (ReKD) [14], mimicking features (MF) [15], regularizing feature norm and direction (RFND) [16], self-supervised knowledge distillation (SSKD) [37] and the hierarchical self-supervised augmented knowledge distillation (HSAKD) [38].

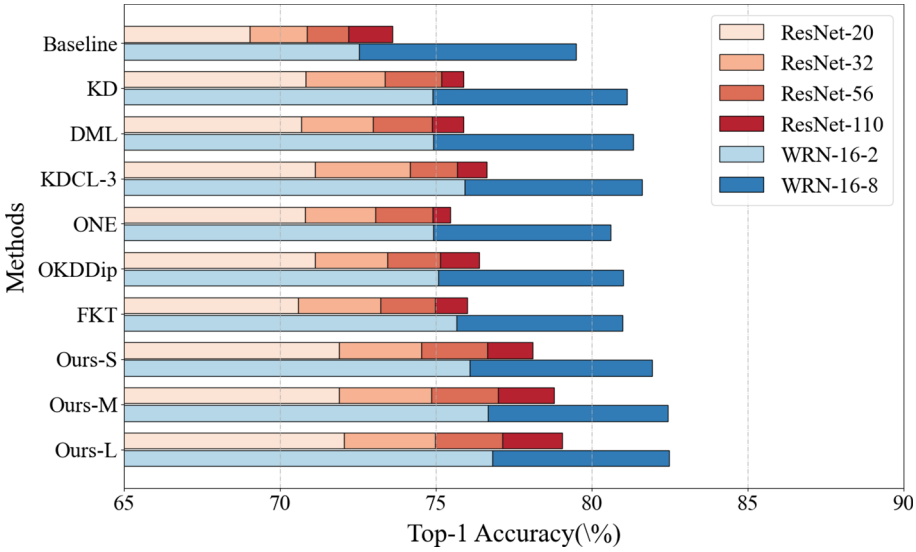

Fig. 5 Top-1 accuracy (%) comparison on the CIFAR-100 dataset with the baseline, KD, and several online mutual learning methods, corresponding to the Table 3

Since there is no explicit concept of teacher-student in our framework, we selected the six-instance implementation M as our base, taking the student network as the deployed instance $\boldsymbol{F}_d$ and the teacher network as the instance $\boldsymbol{F}_3$.

Additionally, the total number of training epochs was increased to 300, and the learning rate decay steps correspondingly changed from [100, 150] to [150, 225].

Compared with other methods, HSAKD uses four times the data samples in training (which can be approximated as a special rotating data augmentation in which 0°, 90°, the 180° and 270° rotations of the same image are regarded as four subclasses). Owing to the difficulty of using the 4× data directly, we introduced six new instances by multi-stage bifurcation, three on $\boldsymbol{F}_d$ (student) and three on $\boldsymbol{F}_3$ (teacher), and arranged the interactions between these instances and the 4× data samples. Thereby, based on the six-instance implementation M, we extended the 12-instance implementation $M+$. The results presented in Table 4 and Fig. 6 highlight the competitiveness of our proposed method compared to other offline methods, and demonstrate that the knowledge extracted through the shallow-wide branches and group ensemble strategy is more suitable for teaching and knowledge transfer.

4.2 Experiments on ImageNet-1k dataset

ImageNet [58] is a much larger visual task dataset for more detailed and rigorous academic use and is the official dataset used in the ImageNet Large Scale Visual Recognition Challenge (ILSVRC). We used its open-access sub-dataset, available on ILSVRC2012 and officially named ImageNet-1k. It contains about 1.28 million images of 1000 common objects for training and 50 thousand images for validation.

We selected ResNet-18 as the deployed network and applied a bifurcation at the end of the second and third stages with {256, 512, 128} and {512, 1024, 256}, respectively,

Table 4 Top-1 accuracy (%) comparison on the CIFAR-100 dataset with some representative teacher-student methods

Student Teacher	ResNet-20 ResNet-56	WRN-16-2 WRN-40-2	WRN-40-1 WRN-40-2
DIST	71.72	76.02	74.70
DKD	71.96	76.24	74.77
FitNet	71.44	75.35	74.25
AT	71.52	75.31	74.42
CRD	71.77	75.99	75.21
ReKD	70.81	76.16	74.91
SSKD	71.02	76.01	75.89
MF	71.44	76.41	74.64
RFND	72.52	76.13	75.00
HSAKD	72.58	77.20	77.00
HSAKD(*)	73.73	78.67	78.12
Ours-M	73.24	77.05	76.01
Ours-M+	**73.80**	**79.16**	**78.35**

All results are reproduced by us using the author-provided codes. 'HSAKD(*)' denotes using a more powerful teacher according to the paper. The best one of each column is in **bold**, and the second best is underlined

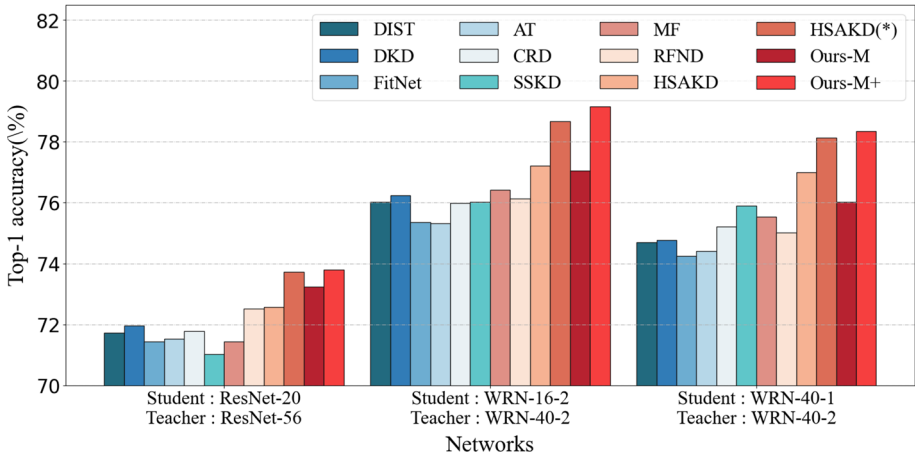

Fig. 6 Top-1 accuracy (%) comparison on the CIFAR-100 dataset with some representative teacher-student methods, corresponding to the Table 4

to form a three-instance version named *Ours-SX*. Additionally, we implemented two six-instance versions. One introduced another ResNet-18 instance and applied the bifurcation method while the other introduced a ResNet-34 instance as the bifurcation base. The former is marked as *Ours-MX* and the latter as *Ours-EX*.

We compared the proposed method with KD [1], DML [13], AT [24], CRD [26], DKD [57], ReKD [14], SSKD [37], and HSAKD [38]. The models were trained for 100 epochs with a starting learning rate of 0.1, which was further divided by 10 at 30, 60, and 90 epochs with a batch size of 128; the other training settings were the same as those used in [44].

Table 5 Top-1 accuracy (%) comparison on the ImageNet-1k dataset

Baseline	KD	AT	DML	CRD	DKD	ReKD	SSKD	HSAKD	Ours-SX/MX/EX
70.13	70.66	70.70	71.08	71.38	71.70	71.61	71.62	**72.16**	71.35/71.58/<u>71.87</u>

The best one is in **bold**, and the second best is <u>underlined</u>

Table 6 Study on instance number

Group	C_{S_a}	C_{S_b}	Acc (%)
Ours-M	2	3	77.00
(1)	0	3	76.28
(2) Ours-S	2	0	76.78
(3) Ours-L	2	6	77.13
(4)	0	6	76.86
(5)	2	9	77.32

The results shown in Table 5 indicate that our proposed methods are competitive with the other methods. Similar to the results for the CIFAR-100 dataset, HSAKD adopts a data augmentation method and has an advantage in the distillation results. Limited by the computing devices, we did not perform further experiments on ImageNet-1k for comparison under rotating data augmentation.

4.3 Ablation Study

We conducted more detailed studies and analyses on the sensitive parts and hyperparameters of the framework. The following experiments and results are based on the CIFAR-100 dataset, and the implementations involved are mainly the Ours-S/M/L versions in Section IV.A and their variants.

4.3.1 Instance Number

First, we examined the effect of the instance number. The control group was *Ours-M* of ResNet-56 owing to its performance and moderate number of instances in $\boldsymbol{S}_a$ and $\boldsymbol{S}_b$. We added the following five experimental groups:

(1) Removing all instances in $\boldsymbol{S}_a$;
(2) Removing all instances in$\boldsymbol{S}_b$ (i.e. *Ours-S*);
(3) Adding three more instances to $\boldsymbol{S}_b$ (i.e., *Ours-L*);
(4) Removing all instances in $\boldsymbol{S}_a$ and adding three more instances to $\boldsymbol{S}_b$.
(5) Adding six more instances to $\boldsymbol{S}_b$.

For simplicity, in the following, we use $C_{\boldsymbol{S}}$ to represent the number of instances in set $\boldsymbol{S}$. The results are shown in Table 6.

It can be found that the performance of $\boldsymbol{F}_d$ will decline after removing the instance of any group. Conversely, when the number of instances increases, the method can still benefit from the increase reasonably. This result also indicates that different groups of instances contribute differently to the distillation effect, with the instances in $\boldsymbol{S}_a$ having a greater impact on the performance of $\boldsymbol{F}_d$.

Table 7 Study on branch/individual

$\boldsymbol{F}_1$	$\boldsymbol{F}_2$	$\boldsymbol{F}_4$	$\boldsymbol{F}_5$	Acc (%)	Params (M)	Time (s)
				77.00	3.11	39.02
✓				76.89	3.32	43.97
✓	✓			76.68	3.53	55.31
✓	✓	✓		76.41	3.73	60.01
✓	✓	✓	✓	76.28	3.94	68.83

The experiments have a hierarchical relationship, with the symbol "✓" indicating that the corresponding instances ($\boldsymbol{F}_1$, $\boldsymbol{F}_2$, $\boldsymbol{F}_4$ and $\boldsymbol{F}_5$) were gradually detached and became individual instead of bifurcation. The models are implemented using PyTorch 1.10 with CUDA 11.3. The time period is measured on the Nvidia RTX 3090 GPU for an entire training epoch of 50000 images

4.3.2 Multi-stage Bifurcation

To verify the effectiveness of the multi-stage bifurcation, we gradually replaced these shared-path instances with independent networks. In this experiment, we used the six-instance implementation *Ours-M* of ResNet56 as the baseline. For convenience, we denote the instances in $\boldsymbol{S}_a$ by $\boldsymbol{F}_1$ and $\boldsymbol{F}_2$, while $\boldsymbol{F}_3$, $\boldsymbol{F}_4$ and $\boldsymbol{F}_5$ represent those in $\boldsymbol{S}_b$ where $\boldsymbol{F}_1$ and $\boldsymbol{F}_2$ are the bifurcation of $\boldsymbol{F}_d$ and $\boldsymbol{F}_4$ and $\boldsymbol{F}_5$ are of those of $\boldsymbol{F}_3$. Then, we gradually replaced instances $\boldsymbol{F}_1$, $\boldsymbol{F}_2$, $\boldsymbol{F}_4$ and $\boldsymbol{F}_5$ with their individual replicas to form a series of experimental groups in numerical order. The results are shown in Table 7. This time, we focus on not only the accuracy but also the parameter amount and the time required for one complete epoch in training.

As seen in the results, although individual instances introduced more parameters into the framework, they did not effectively improve the performance of $\boldsymbol{F}_d$, but significantly increased the training time, which deviates slightly from the common belief that more parameters shall lead to better performance. Our explanation for the aforementioned results is that bifurcation forces these shared trunks to be supervised by both sides and thus establish a special interaction to obtain better gradient feedback than individuals.

In order to provide additional evidence, we conducted more experiments to observe the performance of $\boldsymbol{F}_d$ and its two brother instances $\boldsymbol{F}_1$ and $\boldsymbol{F}_2$ under the only supervision of the ground truth label. $\boldsymbol{F}_d$ is always supervised, while $\boldsymbol{F}_1$ and $\boldsymbol{F}_2$ can be supervised, resulting in the following four experimental configurations:

(1) Only $\boldsymbol{F}_d$ is supervised;
(2) $\boldsymbol{F}_d$ and $\boldsymbol{F}_1$ are supervised;
(3) $\boldsymbol{F}_d$ and $\boldsymbol{F}_2$ are supervised;
(4) all of them are supervised.

The results shown in Table 8 indicate that the accuracy of $\boldsymbol{F}_d$ can be improved even by applying conventional cross-entropy to $\boldsymbol{F}_1$ and $\boldsymbol{F}_2$, which confirms the effectiveness of bifurcation for better gradient utilization. Additionally, the results demonstrate the effectiveness of the proposed shallow-wide branch design for smooth gradient propagation.

4.3.3 Grouping Strategy

As mentioned previously, there are many grouping strategies, and here we selected several representative ones for evaluation. Since we need sufficient instances for different group

Table 8 Study on multi-stage bifurcation

Cross-entropy supervision			Top 1 accuracy (%)
$\boldsymbol{F}_d$	$\boldsymbol{F}_1$	$\boldsymbol{F}_2$	of $\boldsymbol{F}_d$
✓			72.20
✓	✓		72.63
✓		✓	73.13
✓	✓	✓	73.37

Table 9 Study on grouping

Grouping strategy	Number of groups	Acc (%)
$\{\boldsymbol{F}_1,\boldsymbol{F}_2\}$ $\{\boldsymbol{F}_3,\boldsymbol{F}_4,\boldsymbol{F}_5,\boldsymbol{F}_6,\boldsymbol{F}_7,\boldsymbol{F}_8\}$ (Ours-L)	2	77.13±0.13
$\{\boldsymbol{F}_1\}$ $\{\boldsymbol{F}_2\}$ $\{\boldsymbol{F}_3\}$ $\{\boldsymbol{F}_4\}$ $\{\boldsymbol{F}_5\}$ $\{\boldsymbol{F}_6\}$ $\{\boldsymbol{F}_7\}$ $\{\boldsymbol{F}_8\}$	8	76.35±0.17
$\{\boldsymbol{F}_1,\boldsymbol{F}_2,\boldsymbol{F}_3,\boldsymbol{F}_4,\boldsymbol{F}_5,\boldsymbol{F}_6,\boldsymbol{F}_7,\boldsymbol{F}_8\}$	1	77.08±0.26
$\{\boldsymbol{F}_1\}$ $\{\boldsymbol{F}_2\}$ $\{\boldsymbol{F}_3,\boldsymbol{F}_4,\boldsymbol{F}_5,\boldsymbol{F}_6,\boldsymbol{F}_7,\boldsymbol{F}_8\}$	3	76.70±0.24
$\{\boldsymbol{F}_1,\boldsymbol{F}_2\}$ $\{\boldsymbol{F}_3\}$ $\{\boldsymbol{F}_4\}$ $\{\boldsymbol{F}_5\}$ $\{\boldsymbol{F}_6\}$ $\{\boldsymbol{F}_7\}$ $\{\boldsymbol{F}_8\}$	7	76.58±0.31
$\{\boldsymbol{F}_1,\boldsymbol{F}_2\}$ $\{\boldsymbol{F}_3,\boldsymbol{F}_4,\boldsymbol{F}_5\}$ $\{\boldsymbol{F}_6,\boldsymbol{F}_7,\boldsymbol{F}_8\}$	3	77.25±0.24

For clarity, instances within the same ensemble group are enclosed in "{}"

strategies, we selected the nine-instance implementation *Ours-L* of ResNet-56 with two instances ($\boldsymbol{F}_1$ and $\boldsymbol{F}_2$) in $\boldsymbol{S}_a$ and six ($\boldsymbol{F}_3$ to $\boldsymbol{F}_8$) in $\boldsymbol{S}_b$. We conducted an extra study using the following strategies:

(1) Each as a group (eight groups in total);
(2) All as a group (only one group);
(3) $\boldsymbol{S}_a$ adopts (1) and $\boldsymbol{S}_b$ adopts (2) (three groups in total);
(4) $\boldsymbol{S}_a$ adopts (2) and $\boldsymbol{S}_b$ adopts (1) (seven groups in total);
(5) split three instances out of $\boldsymbol{S}_b$ to form group $\boldsymbol{S}_c$ (three groups in total).

As shown in Table 9, Strategy (1) and (2) indicate that an ensemble that is too weak and too decentralized is unsuitable, while an ensemble that is too strong leads to the loss of implicit distribution information in the predictions, also rendering it unsuitable. Better results can be achieved by appropriately controlling the number and intensity of instance ensembles in each group as Strategy (5).

4.3.4 Feature Representation

Furthermore, we conducted additional comparisons at the feature level. Specifically, we adopt various methods to train ResNet-56, and then extract their feature maps before the final fully connected layer. For better comparison, we utilize the t-SNE method for dimension reduction, and the visualized results are shown in Fig. 7.

It can be found that, compared with the existing methods, our proposed method can effectively expand the inter-class distance while simultaneously tightening the intra-class samples, which results in better classification performance.

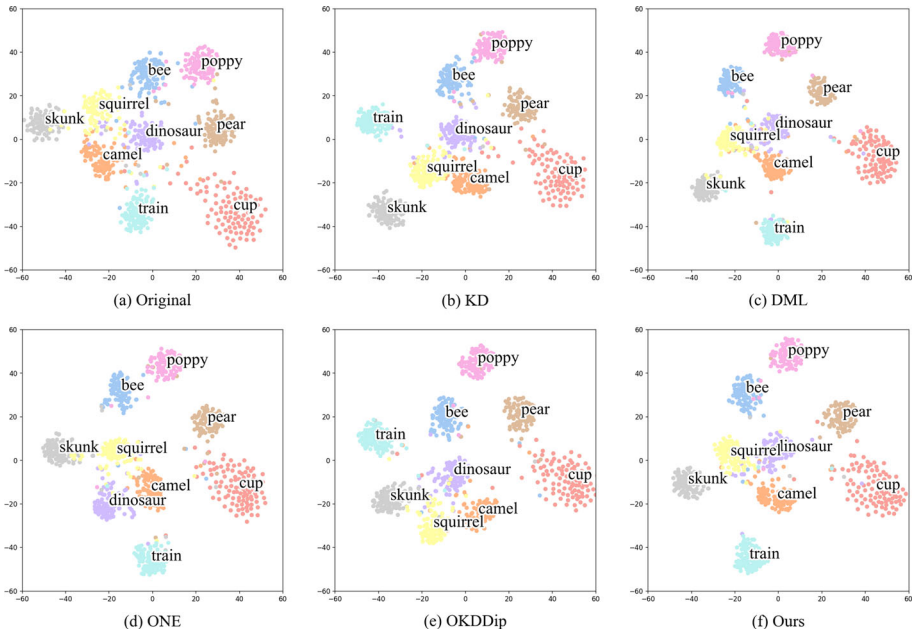

Fig. 7 The visualization of feature representation after t-SNE dimension reduction on the CIFAR-100 dataset. We randomly select 10 categories in the CIFAR-100 validation dataset for a total of 1000 samples. All results are presented on a two-dimensional plane of the same scale for comparison

5 Conclusion

In this study, we proposed an asymmetric knowledge distillation method based on a deployed network. To do so, we first proposed an extendable knowledge distillation framework and introduced the concepts of instances and interactions as key components of the framework. The framework aims to address the implicit symmetry of the distillation model while ensuring simplicity and unity. We then discussed the training-deployment asymmetry of the task under this framework and presented the pipeline of the asymmetric distillation method. Additionally, we designed a multi-stage shallow-wide bifurcation method to complete the pipeline, which consists of a multi-stage bifurcation point selection and a shallow-wide branch supplement. Experiments on the CIFAR-100 and ImageNet-1k datasets demonstrate that the implementation under this framework outperforms many existing methods in terms of validation, which proves the effectiveness of our asymmetric distillation method and further bolsters the rationality of the unified framework.

Notably, the proposed instance-interaction framework is not a specific model but a higher abstraction of existing methods and models. As the framework for all other distillation models, it translates different distillation methods into a standard instance-interaction description and maintains compatibility and simplicity, indicating that these methods, although seemingly different, are unified from a higher-level perspective. With the help of this framework, we decomposed the original distillation task into two subproblems, instance extension and interaction arrangement, and demonstrated the importance of the asymmetry model, which is the basis of the deployed instance-based asymmetric distillation method. In addition to the case investigated in this study, we can reproduce most of the existing distillation algorithms

under the proposed framework, precisely controlling for the difference between them within a limited range (instances and interactions), and thus providing more intuitive and detailed discussions and comparisons, which are of great significance to future studies in the field.

To achieve better outcomes with this framework, we need more theoretical and general guidelines on the instance-interaction design to explicitly evaluate instance and interaction performances, such as assessing the teaching ability of an instance. Moreover, the framework requires more careful refinements to adapt to more complex input–output structures in tasks beyond image classification, such as object detection or image reconstruction.

To overcome these limitations, more robust branches and reliable adaptive grouping strategies should be explored and discussed to develop additional theoretical discoveries and applicable implementations. Nonetheless, we hope that our proposed method can serve as a starting point to encourage further and deeper theoretical and applied studies in the field of knowledge distillation.

Acknowledgements This work was supported in part by the Zhejiang Provincial Natural Science Foundation of China under Grant No.LDT23F01013F01; and in part by the Fundamental Research Funds for the Central Universities.

Author Contributions Xin Ye wrote the main manuscript. All authors reviewed the manuscript critically and provided feedback.

Declarations

Conflict of interest The authors declare no competing interests.

Open Access This article is licensed under a Creative Commons Attribution 4.0 International License, which permits use, sharing, adaptation, distribution and reproduction in any medium or format, as long as you give appropriate credit to the original author(s) and the source, provide a link to the Creative Commons licence, and indicate if changes were made. The images or other third party material in this article are included in the article's Creative Commons licence, unless indicated otherwise in a credit line to the material. If material is not included in the article's Creative Commons licence and your intended use is not permitted by statutory regulation or exceeds the permitted use, you will need to obtain permission directly from the copyright holder. To view a copy of this licence, visit http://creativecommons.org/licenses/by/4.0/.

References

1. Hinton G, Vinyals O, Dean J (2015) Distilling the knowledge in a neural network. arXiv preprint arXiv:1503.02531
2. Yuan L, Tay FE, Li G, Wang T, Feng J (2020) Revisiting knowledge distillation via label smoothing regularization. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, pp 3903–3911
3. Li X, Xiong H, An H, Xu C, Dou D (2022) Colam: co-learning of deep neural networks and soft labels via alternating minimization. Neural Process Lett 54(6):4735–4749
4. Li Y, Yang J, Song Y, Cao L, Luo J, Li L-J (2017) Learning from noisy labels with distillation. In: Proceedings of the IEEE international conference on computer vision, pp 1910–1918
5. Romero A, Ballas N, Kahou SE, Chassang A, Gatta C, Bengio Y (2014) Fitnets: Hints for thin deep nets. arXiv preprint arXiv:1412.6550
6. Yim J, Joo D, Bae J, Kim J (2017) A gift from knowledge distillation: Fast optimization, network minimization and transfer learning. In: Proceedings of the IEEE conference on computer vision and pattern recognition, pp 4133–4141
7. Heo B, Kim J, Yun S, Park H, Kwak N, Choi JY (2019) A comprehensive overhaul of feature distillation. In: Proceedings of the IEEE/CVF international conference on computer vision, pp 1921–1930
8. Li L, Su W, Liu F, He M, Liang X (2023) Knowledge fusion distillation: improving distillation with multi-scale attention mechanisms. Neural Process Lett 55(5):6165–6180

9. Zhu J, Tang S, Chen D, Yu S, Liu Y, Rong M, Yang A, Wang X (2021) Complementary relation contrastive distillation. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, pp 9260–9269
10. Ji M, Heo B, Park S (2021) Show, attend and distill: Knowledge distillation via attention-based feature matching. In: Proceedings of the AAAI conference on artificial intelligence, vol. 35, pp 7945–7952
11. Liu W, Nie S, Yin J, Wang R, Gao D, Jin L (2021) Sskd: Self-supervised knowledge distillation for cross domain adaptive person re-identification. In: 2021 7th IEEE international conference on network intelligence and digital content, pp 81–85 . IEEE
12. Yang Z, Shou L, Gong M, Lin W, Jiang D (2020) Model compression with two-stage multi-teacher knowledge distillation for web question answering system. In: Proceedings of the 13th international conference on web search and data mining, pp 690–698
13. Zhang Y, Xiang T, Hospedales TM, Lu H (2018) Deep mutual learning. In: Proceedings of the IEEE conference on computer vision and pattern recognition, pp 4320–4328
14. Chen P, Liu S, Zhao H, Jia J (2021) Distilling knowledge via knowledge review. In: Proceedings of the IEEE/CVF Conference on computer vision and pattern recognition, pp 5008–5017
15. Wang G-H, Ge Y, Wu J (2021) Distilling knowledge by mimicking features. IEEE Trans Pattern Anal Mach Intell 44(11):8183–8195
16. Wang Y, Cheng L, Duan M, Wang Y, Feng Z, Kong S (2023) Improving knowledge distillation via regularizing feature norm and direction. arXiv preprint arXiv:2305.17007
17. Kim J, Park S, Kwak N (2018) Paraphrasing complex network: network compression via factor transfer. Adv Neural Inform Process Syst, 31
18. Lee SH, Kim DH, Song BC (2018) Self-supervised knowledge distillation using singular value decomposition. In: Proceedings of the European conference on computer vision, pp 335–350
19. Srinivas S, Fleuret F (2018) Knowledge transfer with jacobian matching. In: International conference on machine learning, pp 4723–4731. PMLR
20. Park W, Kim D, Lu Y, Cho M (2019) Relational knowledge distillation. In: Proceedings of the IEEE/CVF Conference on computer vision and pattern recognition, pp 3967–3976
21. Liu Y, Cao J, Li B, Yuan C, Hu W, Li Y, Duan Y (2019) Knowledge distillation via instance relationship graph. In: Proceedings of the IEEE/CVF Conference on computer vision and pattern recognition, pp 7096–7104
22. Chen Z, Zheng X, Shen H, Zeng Z, Zhou Y, Zhao R (2020) Improving knowledge distillation via category structure. In: Proceedings of the European conference on computer vision, pp 205–219. Springer
23. Tung F, Mori G (2019) Similarity-preserving knowledge distillation. In: Proceedings of the IEEE/CVF International conference on computer vision, pp 1365–1374
24. Komodakis N, Zagoruyko S (2017) Paying more attention to attention: improving the performance of convolutional neural networks via attention transfer. In: International conference on learning representations
25. Chariton A, Passalis N, Pleros N, Tefas A (2023) Mutual information-based neural network distillation for improving photonic neural network training. Neural Process Lett, 1–16
26. Tian Y, Krishnan D, Isola P (2019) Contrastive representation distillation. In: international conference on learning representations
27. Lee H, Hwang SJ, Shin J (2020) Self-supervised label augmentation via input transformations. In: International conference on machine learning, pp 5714–5724. PMLR
28. Zhu X, Gong S et al (2018) Knowledge distillation by on-the-fly native ensemble. Adv Neural Inf Process Syst 31:7517–7527
29. Chen D, Mei J-P, Wang C, Feng Y, Chen C (2020) Online knowledge distillation with diverse peers. In: Proceedings of the AAAI conference on artificial intelligence, 34, pp 3430–3437
30. Liu Y, Zhang W, Wang J (2020) Adaptive multi-teacher multi-level knowledge distillation. Neurocomputing 415:106–113
31. Guo Q, Wang X, Wu Y, Yu Z, Liang D, Hu X, Luo P (2020) Online knowledge distillation via collaborative learning. In: Proceedings of the IEEE/CVF Conference on computer vision and pattern recognition, pp 11020–11029 . IEEE
32. Gou J, Hu Y, Sun L, Wang Z, Ma H (2024) Collaborative knowledge distillation via filter knowledge transfer. Expert Syst Appl 238:121884. https://doi.org/10.1016/j.eswa.2023.121884
33. Zhang L, Song J, Gao A, Chen J, Bao C, Ma K (2019) Be your own teacher: Improve the performance of convolutional neural networks via self distillation. In: Proceedings of the IEEE/CVF international conference on computer vision, pp 3713–3722
34. Lee C-Y, Xie S, Gallagher P, Zhang Z, Tu Z (2015) Deeply-supervised nets. In: Artificial Intelligence and Statistics, pp 562–570. PMLR

35. Zhang Z, Sabuncu M (2020) Self-distillation as instance-specific label smoothing. Adv Neural Inform Process Syst, vol. 33
36. Ji M, Shin S, Hwang S, Park G, Moon I-C (2021) Refine myself by teaching myself: Feature refinement via self-knowledge distillation. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, pp. 10664–10673
37. Xu G, Liu Z, Li X, Loy CC (2020) Knowledge distillation meets self-supervision. In: Proceedings of the European conference on computer vision, Springer. pp 588–604
38. Yang C, An Z, Cai L, Xu Y (2021) Hierarchical self-supervised augmented knowledge distillation. In: Proceedings of the international joint conference on artificial intelligence, pp 1217–1223
39. Robbins H, Monro S (1951) A stochastic approximation method. The Annals Math Statist, 400–407
40. Li H, Zhang H, Qi X, Yang R, Huang G (2019) Improved techniques for training adaptive deep networks. In: Proceedings of the IEEE/CVF international conference on computer vision, pp 1891–1900. IEEE
41. Li Z, Huang Y, Chen D, Luo T, Cai N, Pan Z (2020) Online knowledge distillation via multi-branch diversity enhancement. In: Proceedings of the Asian conference on computer vision
42. Walawalkar D, Shen Z, Savvides M (2020) Online ensemble model compression using knowledge distillation. In: Proceedings of the European conference on computer vision, Springer, pp 18–35
43. Sun D, Yao A, Zhou A, Zhao H (2019) Deeply-supervised knowledge synergy. In: Proceedings of the IEEE/CVF Conference on computer vision and pattern recognition, IEEE, pp 6997–7006
44. He K, Zhang X, Ren S, Sun J (2016) Deep residual learning for image recognition. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, IEEE, pp 770–778
45. Zagoruyko S, Komodakis N (2016) Wide residual networks. In: Proceedings of the British machine vision conference . British Machine Vision Association
46. Howard AG, Zhu M, Chen B, Kalenichenko D, Wang W, Weyand T, Andreetto M, Adam H (2017) Mobilenets: Efficient convolutional neural networks for mobile vision applications. arXiv preprint arXiv:1704.04861
47. Huang G, Liu Z, Van Der Maaten L, Weinberger KQ (2017) Densely connected convolutional networks. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, IEEE, pp 4700–4708
48. Lin T-Y, Dollár P, Girshick R, He K, Hariharan B, Belongie S (2017) Feature pyramid networks for object detection. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, IEEE, pp 2117–2125
49. Krizhevsky A, Sutskever I, Hinton GE (2012) Imagenet classification with deep convolutional neural networks. Adv Neural Inform Process Syst 25:1097–1105
50. Simonyan K, Zisserman A (2014) Very deep convolutional networks for large-scale image recognition. arXiv preprint arXiv:1409.1556
51. Szegedy C, Liu W, Jia Y, Sermanet P, Reed S, Anguelov D, Erhan D, Vanhoucke V, Rabinovich A (2015) Going deeper with convolutions. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, IEEE, pp 1–9
52. Muller R, Kornblith S, Hinton GE (2019) When does label smoothing help? In: Wallach H, Larochelle H, Beygelzimer A, Alché-Buc F, Fox E, Garnett R (eds) Advances in neural information processing systems, vol 32. Curran Associates Inc, New York
53. Sandler M, Howard A, Zhu M, Zhmoginov A, Chen L-C (2018) Mobilenetv2: Inverted residuals and linear bottlenecks. In: Proceedings of the IEEE conference on computer vision and pattern recognition, pp 4510–4520
54. Lou Z, Ye X, Zhang L, Wu W, He Y, Zhou H (2023) Rethinking the value of local feature fusion in convolutional neural networks. Neural Process Lett, 1–16
55. Krizhevsky A, Hinton G, et al (2009) Learning multiple layers of features from tiny images. Handbook of Systemic Autoimmune Diseases
56. Huang T, You S, Wang F, Qian C, Xu C (2022) Knowledge distillation from a stronger teacher. Adv Neural Inf Process Syst 35:33716–33727
57. Zhao B, Cui Q, Song R, Qiu Y, Liang J (2022) Decoupled knowledge distillation. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, pp 11953–11962
58. Deng J, Dong W, Socher R, Li L-J, Li K, Fei-Fei L (2009) Imagenet: A large-scale hierarchical image database. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, IEEE, pp 248–255

Publisher's Note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.