# Shadow Knowledge Distillation: Bridging Offline and Online Knowledge Transfer

**Anonymous Author(s)**
Affiliation
Address
email

## Abstract

Knowledge distillation can be generally divided into offline and online categories according to whether teacher model is pre-trained and persistent during the distillation process. Offline distillation can employ existing models yet always demonstrates inferior performance than online ones. In this paper, we first empirically show that the essential factor for their performance gap lies in the reversed distillation from student to teacher, rather than the training fashion. Offline distillation can achieve competitive performance gain by fine-tuning pre-trained teacher to adapt student with such reversed distillation. However, this fine-tuning process still costs lots of training budgets. To alleviate this dilemma, we propose SHAKE, a simple yet effective **SHA**dow **K**nowl**E**dge transfer framework to bridge offline and online distillation, which trades the accuracy with efficiency. Specifically, we build an extra shadow head on the student's backbone to mimic the predictions of pre-trained teacher as its shadow. Then, this shadow head is leveraged as a proxy teacher to perform bidirectional distillation with student on the fly. In this way, SHAKE not only updates this student-aware proxy teacher with the knowledge of pre-trained model, but also greatly optimizes costs of augmented reversed distillation. Extensive experiments on classification and object detection tasks demonstrate that our technique achieves state-of-the-art results with different CNNs and Vision Transformer models. Additionally, our method shows strong compatibility with multi-teacher and augmentation strategies by gaining additional performance improvement. The code will be made public.

## 1 Introduction

Deep Neural Networks (DNNs) have achieved great success in tackling a variety of tasks [21, 11]. Despite appealing performance, prevailing DNN models usually have large numbers of parameters, bringing heavy costs of computation. To alleviate this problem, many network compression methods [23, 32] have been proposed, among which Knowledge Distillation (KD) [16, 43, 12] has recently attracted increased attention.

The aim of KD is to transfer the learnt knowledge of a high-capacity teacher model to a low-capacity student model. Numerous offline methods [1, 16] use a two-stage training process that begins with training a teacher model and then keeping it fixed to distill student model (see KD in Figure 1 (a)). Besides offline fashion, recent online methods [22, 4] adopt a one-stage training process, jointly training the student and teacher/peer models using bidirectional distillation like DML [48] in Figure 1 (b). These online distillations always surpass the offline distillations under the same teacher model. However, some large teacher models trained from scratch would bring some difficulties (*e.g.*, high computational resources and unstable optimizations) for distillation, especially for tasks that rely on large transformers (*e.g.*, BERT [42], GPT-3 [2] and ViT-MoE [34]). Therefore, two questions

Submitted to 36th Conference on Neural Information Processing Systems (NeurIPS 2022). Do not distribute.

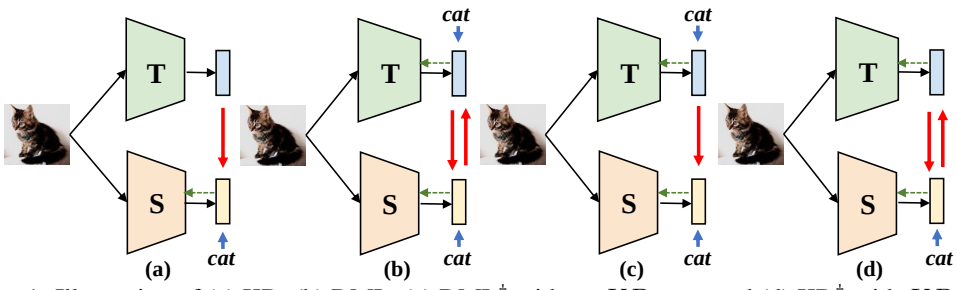


Figure 1: Illustration of (a) KD, (b) DML, (c) DML$^\dagger$ without $KD_{S\rightarrow T}$, and (d) KD$^\dagger$ with $KD_{S\rightarrow T}$. $KD_{S\rightarrow T}$ denotes reversed distillation from student to teacher.

Table 1: Left: comparison of training time, Top-1 accuracy (%), and teacher-student gap (T-S gap) among the (a) KD, (b) DML, (c) DML$^\dagger$ without $KD_{S\rightarrow T}$, (d) KD$^\dagger$ with $KD_{S\rightarrow T}$, and our SHAKE for ResNet-20 (69.09%) via pre-trained teacher (Pre-T) ResNet-110 (74.31%) on CIFAR-100. Training time is measured on a single 2080Ti GPU and $\times$ represents the improving ratios than KD. The teacher-student gap [6] is defined as KL divergence between their outputs (lower is better). Right: training time and accuracy of different settings for KD$^\dagger$ and SHAKE. KD$^\dagger$ (head) means only updating head, and KD$^\dagger$ (60) means only updating for 60 epochs. SHAKE (share) denotes proxy teacher shares student's backbone. SHAKE (multi) denotes SHAKE with two ResNet-110 as teachers.

| Method | Pre-T | $KD_{T\rightarrow S}$ | $KD_{S\rightarrow T}$ | Time | Top-1 | T-S gap |
|---|---|---|---|---|---|---|
| (A) KD | ✓ | ✓ | ✗ | $\times$1.00 | 70.66 | 1.12 |
| (B) DML | ✗ | ✓ | ✓ | $\times$4.32 | 71.52 | 0.28 |
| (C) DML$^\dagger$ | ✗ | ✓ | ✗ | $\times$4.41 | 70.55 | 0.62 |
| (D) KD$^\dagger$ | ✓ | ✓ | ✓ | $\times$4.29 | 71.76 | 0.76 |
| (G) SHAKE | ✓ | ✓ | ✓ | $\times$1.18 | 72.02 | 0.21 |

| Method | Time | Top-1 |
|---|---|---|
| (F) KD$^\dagger$ (head) | $\times$1.13 | 71.05 |
| (G) KD$^\dagger$ (60) | $\times$2.12 | 71.13 |
| (H) SHAKE (alone) | $\times$1.85 | 71.82 |
| (I) SHAKE (share) | $\times$1.18 | 72.02 |
| (J) SHAKE (Multi) | $\times$1.32 | 72.55 |

naturally arise: (1) Why is there a performance gap between offline and online distillation? (2) How the performance of offline distillation can be advanced with design techniques of online KD methods?

To clarify the first question, we compare KD and DML regarding the training fashion of teacher and distillation loss in Table 1. Contrary to the common belief, we empirically observe that training fashion may not affect the distillation performance since DML$^\dagger$ obtains similar performance as KD (70.55% vs. 70.66%). Instead, the reversed distillation from the student model yields significant accuracy gains for KD$^\dagger$ than KD (71.76% vs. 70.66%) and DML than DML$^\dagger$ (71.52% vs. 70.55%).

Then, we analyze the output discrepancy of the teacher-student for different methods, and the reversed distillation reduces the gap from 1.12 of KD to 0.76 of KD$^\dagger$. Therefore, the main reasons behind the performance gap lie in two aspects: (a) Teacher models in conventional offline pipelines are not optimized for the student model. Thus, they could only provide general knowledge, which may be suboptimal for the particular student. (b) The reversed distillation changes the universal teacher model into a student-aware one and bridges the teacher-student gap, which clearly justifies:

*Teachers should teach students in accordance with their aptitude and shouldn't follow the same pattern.*

For the second question, fine-tuning pre-trained teacher model with reversed distillation is a straightforward way to bridge the two training fashions. However, in most scenarios, fine-tuning the whole network still needs lots of training budges (more 4$\times$ costs than KD in Table 1). Some trade-offs in reducing the fine-tuning parameters or epochs involve performance loss. Thus, how to augment reversed distillation without much extra overhead is an important issue for the application. Besides fine-tuning, building a proxy teacher model (see Figure 2 (a)) to inherit the knowledge of pre-trained models and receive reversed distillation from students also enjoys the same benefits. As shown in Table 1 (H), this proxy teacher model (SHAKE (alone)) can achieve competitive gains than fine-tuning the whole teacher model (KD$^\dagger$). Recent weight sharing strategies in AutoML [45, 30**?** ] and can effectively save training overhead and peserve logit consistency. This encourages us to adopt the same student architecture and weight sharing strategy to generate the proxy teacher model. Thus, we allow this proxy teacher to share the student's backbone but use an individual shadow head to preserve the diversity of logits representations (see Figure 2 (b)). As shown in Table 1 (I), this

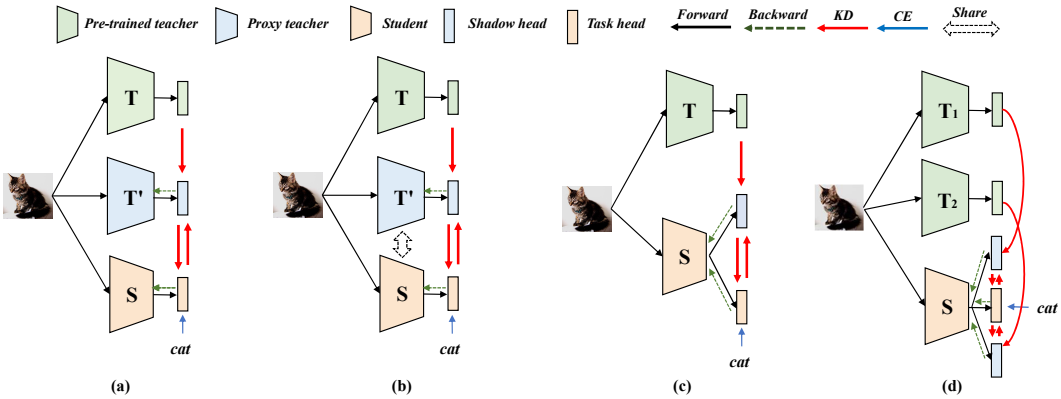


Figure 2: Evolution of our **SHA**dow **K**nowl**E**dge distillation (SHAKE). (a) We build a proxy teacher model to inherit knowledge from pre-trained models as an alternative to costly fine-tuning teacher models. (b) This proxy teacher model could adopt the same student structure and reuse the student's backbone to reduce training budgets. (c) Our SHAKE framework: Base on (b), SHAKE uses the shadow head as the proxy teacher to perform bidirectional KD with student, updated with the soft label of pre-trained teacher. Both teacher models and shadow head are present only during training and can be discarded during inference. The implementations of (b) and (c) are identical. (d) Our SHAKE for multiple teachers: SHAKE leverages multiple shadow heads to individually follow various teacher models.

sharing strategy in SHAKE (share) presents four benefits: (1) more than $3\times$ training acceleration than DML and fine-tuning the whole teacher model. (2) No need for architecture selection costs for proxy teacher. (3) Additional accuracy gains than individual proxy teacher because knowledge inherited by proxy teachers also directly improves the representation of students' backbone in the weight-sharing process. (4) In the shared backbone setting, the reversed distillation works with additional regularization [49] and logit consistency [27].

Based on the above observations, we propose **SHA**dow **K**nowl**E**dge distillation (SHAKE), a novel and effective logits distillation framework. Our SHAKE builds a proxy teacher model and updates its weights via the original teacher model predictions. In this way, SHAKE enjoys the same benefits with teacher fine-tuning to mine the knowledge of the pre-trained model and can perform bidirectional supervision with student. To optimize training costs, this proxy teacher shares the backbone with the students but leverages an individual head to preserve the diversity of logits representations. This head is named shadow head since it imitates the original teacher model just like its shadow. During training, we only need to additionally train this shadow head with small training budgets (only $1.18\times$ costs than KD). After training, all teacher models and shadow head can be deprecated, and student model can be separately applied for inference without any overhead. Moreover, we extend the SHAKE to multi-teacher model scenarios using multiple shadow heads to inherit diverse knowledge.

In principle, SHAKE alters the chain of knowledge transfer from *pre-trained teacher*$\rightarrow$*student* in KD to *pre-trained teacher*$\rightarrow$*proxy teacher*$\rightleftarrows$*student*. Other adaptive KD [28] only employs middle networks to distill as *large teacher*$\rightarrow$*middle teacher*$\rightarrow$*student*, which is not optimized for students and requires multi-step training. The merits of SHAKE lie in three-fold. First, it effectively reduces the teacher-student capability gap with reversed distillation, bringing significant gains when pre-trained teacher models are available. Second, for the scenario without pre-trained teacher models, SHAKE also enables the offline KD methods more effective to alleviate the unstable optimization issues of online KD methods. Thus, SHAKE bridges the offline and online KD methods and enjoys the advantages of both methods. Third, SHAKE achieves favorable trade-offs between accuracy and training budget. By contrast, other adaptive KD [28] needs to sequentially train multiple models with lots of additional training time and resources. We hope that these intriguing observations in SHAKE would expand the application of KD and facilitate future research for KD work to some extent.

We conduct extensive experiments on classification and detection to verify the superiority of the proposed method. SHAKE achieves a consistent and significant accuracy boost in various neural networks and data augmentations, which outperforms other methods by large margins. For example, SHAKE obtains $1.21\% \sim 3.65\%$ accuracy gains and more than $3\times$ training acceleration than DML on CIFAR-100. On the challenging ImageNet dataset, SHAKE can improve the Top-1 accuracy of

ResNet-18 from 69.75% to 72.37% and MobileNet from 70.13% to 72.96% , which are state-of-the-art performances among KD methods. On vision transformer architecture [40], our approach achieves 75.22% Top-1 accuracy and 2.76% gain for training a VIT-T model. On the object detection task, SHAKE improves the AP by 1.20 for RetinaNet and 1.02 for Faster R-CNN on the MS-COCO dataset, demonstrating the generality of our approach.

In summary, we make the following principle contributions in this paper:

- By analyzing and exploring the difference between offline and online KD methods, we empirically show that the reversed distillation hinders the performance gain, which fixes the discrepancy between teacher-student capability. This motivates us to propose a new **SHA**dow **K**nowl**E**dge distillation (SHAKE) framework to bridge two training fashions, which, to the best of our knowledge, is not achieved in the area of knowledge distillation.
- SHAKE achieves remarkable trade-offs between accuracy and training efficiency with an extra shadow head. The shadow head inherits knowledge from pre-trained models, introduces reversed distillation with students, and accelerates training process by sharing the student's backbone.
- We perform thorough evaluations on classification and detection. SHAKE achieves state-of-the-art performances in multiple datasets and architectures (e.g. CNN and vision transformer). Specifically, ResNet-18, MobileNet, and VIT-T with SHAKE achieve 72.37%, 72.96% and 75.22% Top-1 accuracy on ImageNet, outperforming KD by 1.71%, 2.28% and 3.02%, respectively.

## 2 Shadow Knowledge Distillation

In this section, we first revisit the offline and online KD methods. Then, we present the formulation of SHAKE and its expansion for multi-teacher models. The evolution of our approach is shown in Figure 2.

### 2.1 Revisit of offline and online KD methods

We first review the formulations of offline and online KD. For simplicity, We choose two typical frameworks (*i.e.*, original KD [16] and DML [48]) for analysis. Given training dataset $(X, Y)$ where $X = \{x_i\}_{i=1}^n$ are training samples and $Y = \{y_i\}_{i=1}^n$ are their labels. Let $f_T$ be the output logits of the fixed teacher $T$ and let $f_S$ be the output of student $S$, respectively. In KD, the student network $f_S$ is trained by minimizing:

$$\mathcal{L}_S = \mathcal{L}_{CE}(f_S, Y) + \lambda \mathcal{L}_{KL}(f_S, f_T), \tag{1}$$

where $\lambda$ is a weight for balancing these two terms. $\mathcal{L}_{CE}$ is the regular cross-entropy objective:

$$\mathcal{L}_{CE}(f_S, Y) = H(Y, \sigma(f_S)), \tag{2}$$

where $H(\cdot, \cdot)$ is the cross-entropy loss and $\sigma$ is the softmax function. $\mathcal{L}_{KL}$ in Eq. 1 is the distillation objective for transferring knowledge from a teacher to a student:

$$\mathcal{L}_{KL}\left(f_S, f_T\right) = \tau^2 KL\left(\sigma\left(\frac{f_T}{\tau}\right), \sigma\left(\frac{f_S}{\tau}\right)\right), \tag{3}$$

where $\tau$ is the temperature to generate soft labels and $KL$ represents Kullback-Leibler (KL) divergence. This distillation loss can be considered as a modified cross-entropy loss using the probabilistic outputs of the pre-trained teacher network as the soft labels instead of the one-hot ground-truth labels.

Different from the one-way distillation in KD, DML presents a two-way knowledge transfer strategy in which the probabilistic outputs from both teacher and student networks can be used to guide the training of each other. DML jointly trains the teacher and student networks in an end-to-end manner via interleavingly optimizing two objectives:

$$\begin{aligned} \mathcal{L}_T &= \mathcal{L}_{CE}(f_T, Y) + \lambda \mathcal{L}_{KL}(f_T, f_S) \\ \mathcal{L}_S &= \mathcal{L}_{CE}(f_S, Y) + \lambda \mathcal{L}_{KL}(f_S, f_T), \end{aligned} \tag{4}$$

where the default value of $\lambda$ is 1 in DML.

### 2.2 Formulation of SHAKE

As shown in Figure 2 (c), SHAKE proposes a proxy teacher model with output $f_{T'}$ via shadow head to augment reverse distillation in the offline framework. During the training process, $f_{T'}$ is updated with the output of the pre-trained teacher model $f_T$ and performs mutual distillation with the student models $f_S$ as:

$$\begin{aligned} \mathcal{L}_{T'} &= \mathcal{L}_{KL}(f_{T'}, f_T) + \lambda \mathcal{L}_{KL}(f_{T'}, f_S) \\ \mathcal{L}_S &= \mathcal{L}_{CE}(f_S, Y) + \lambda \mathcal{L}_{KL}(f_S, f_{T'}), \end{aligned} \tag{5}$$

where $N$ is the total number of teacher models.

For multiple teacher models in Figure 2 (d), we build several proxy teachers with outputs ($f_{T'_1}$,$f_{T'_2}$,...,$f_{Ti'}$) to follow outputs ($f_{T_1}$,$f_{T_2}$,...,$f_{T_i}$) of multiple teacher models with various shadow heads. The mutual distillation also exists between the task head of the student and these multiple shadow heads. Similar to Eq. 5, the total optimization function for the multi-teacher scenario can be defined as:

$$\begin{aligned} \mathcal{L}_{T'} &= \sum_{i=1}^{N} \mathcal{L}_{CE}(f_{T'_i}, f_{Ti}) + \lambda \sum_{i=1}^{N} \mathcal{L}_{KL}(f_{T'_i}, f_S) \\ \mathcal{L}_S &= \mathcal{L}_{CE}(f_S, Y) + \lambda \sum_{i=1}^{N} \mathcal{L}_{KL}(f_S, f_{T'_i}). \end{aligned} \tag{6}$$

where $N$ is the total number of teacher models. Furthermore, SHAKE can be combined with advanced multi-teacher methods (*e.g.*, adaptive weights [37] for different teachers in part IV of Table 2).

**Comparison of SHAKE with other KD methods**. Eq. 5, Eq. 4 and Eq. 1 clearly illustrate the difference of SHAKE and KD/DML. Compared to KD, SHAKE introduces reversed distillation so that the teacher can be optimized by the student. In contrast to DML, SHAKE leverages the knowledge of pre-trained models, resulting in additional accuracy gains. As shown in Figure 3, SHAKE achieves a robust boost than KD and baseline during the training process. Meanwhile, training curve of DML shows a highly dynamic oscillation due to unreliable predictions of its teacher trained from scratch. Other adaptive KD methods use the same knowledge transfer way as KD and do not optimize teacher with feedback supervision of student. Thus, these methods, including ATKD [28] and ESKD [6] do not essentially address the teacher-student capacity disparity, and our SHAKE surpasses them with significant margins ($1.48\% \sim 1.59\%$ on ImageNet in Figure 3).

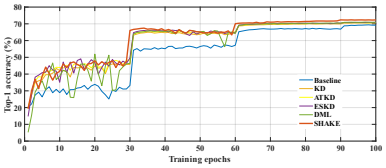


Figure 3: Comparison of training curves of baseline (69.75%), KD (70.66%), ATKD (70.78%), ESKD (70.89%), DML (71.13%) and our SHAKE (72.37%) for ResNet-18 with single ResNet-34 as teacher on ImageNet.

.

## 3 Experiments

In this section, we first evaluate our approach for the classification task on CIFAR-100/ImageNet and the object detection task on MS-COCO. Then, comprehensive ablation experiments are performed to analyze the key design in our SHAKE. As a novel offline logit approach, the main competitor of SHAKE is the original KD [16]. Thus, we perform detailed experimental comparisons between them. Moreover, we compare the performance with recent advanced KD methods. For fair comparisons, we use the public codes of these approaches with the same training and data preprocessing settings throughout the experiments. For SHAKE, we set $\lambda$ and $\tau$ as 1 and 4, respectively. Please refer to the supplementary materials for more training settings.

### 3.1 Experiments on CIFAR-100

**Implementation**. On CIFAR-100 [20], we conduct experiments on various teacher-student models under the same or different architecture style with CRD's settings [39], whose training epochs are 240. We use a mini-batch size of 64 and a standard SGD optimizer with a weight decay of 0.0005. The multi-step learning rate is initialized to 0.05, decayed by 0.1 at 150, 180, and 210 epochs.

Table 2: Comparison of results with advanced distillation methods under the same training setting of 240 epochs. ↑ refers to the performance gain than baseline. Most results of other methods refer to the CRD. SHAKE‡ denotes SHAKE with two teachers. R32x4, R8x4, R50, MV2, Sv1 and Sv2 stand for ResNet32x4, ResNet8x4, ResNet50, MobileNetV2, ShuffleNetV1 and ShuffleNetV2. We report Top-1 mean (std) accuracies (%) over 3 runs.

| | | Same architectural style | | | | Different architectural style | | |
|---|---|---|---|---|---|---|---|---|
| | Teacher<br>Student | WRN-40-2<br>WRN-16-2 | WRN-40-2<br>WRN-40-1 | R32x4<br>R8x4 | VGG13<br>VGG8 | R50<br>MV2 | R32x4<br>SV1 | R32x4<br>SV2 |
| | Teacher | 75.61 | 75.61 | 79.42 | 74.64 | 79.34 | 79.42 | 79.42 |
| | Student | 73.26 | 71.98 | 72.50 | 70.36 | 64.60 | 70.50 | 71.82 |
| I | FitNets [35] | 73.58±0.13 | 72.24±0.14 | 73.50±0.08 | 71.02±0.09 | 63.16±0.28 | 73.59±0.26 | 73.54±0.22 |
| | SP [41] | 73.83±0.12 | 72.43±0.14 | 72.94±0.16 | 72.68±0.27 | 68.08±0.09 | 73.48±0.07 | 74.56±0.11 |
| | RKD [29] | 73.35±0.24 | 72.22±0.25 | 71.90±0.18 | 71.48±0.26 | 64.43±0.08 | 72.28±0.09 | 73.21±0.18 |
| | CRD [39] | 75.48±0.18 | 74.14±0.22 | 75.51±0.17 | 73.94±0.18 | 69.11±0.14 | 75.11±0.13 | 75.65±0.22 |
| | Review [5] | 76.12±0.12 | 75.09±0.26 | 75.63±0.25 | 74.84±0.15 | 70.37±0.17 | 77.14±0.22 | 77.78±0.18 |
| | CL [38] | 74.25±0.12 | 72.63±0.14 | 73.10±0.18 | 71.26±0.22 | 65.76±0.22 | 73.62±0.24 | 73.98±0.23 |
| | AFD [7] | 73.70±0.14 | 72.37±0.15 | 72.98±0.27 | 70.88±0.25 | 64.93±0.21 | 73.68±0.16 | 74.32±0.18 |
| | ONE [22] | 74.68±0.24 | 73.43±0.26 | 73.51±0.28 | 72.01±0.27 | 66.26±0.22 | 74.35±0.25 | 75.12±0.21 |
| | KD [16] | 74.92±0.12 | 73.54±0.15 | 73.33±0.18 | 72.98±0.22 | 67.35±0.24 | 74.07±0.26 | 74.45±0.28 |
| | KD† | 75.58±0.22 | 74.24±0.22 | 74.91±0.22 | 73.65±0.22 | 68.81±0.22 | 75.21±0.12 | 75.95±0.16 |
| | DML [48] | 75.33±0.18 | 73.98±0.22 | 74.30±0.24 | 73.64±0.15 | 68.52±0.22 | 75.58±0.18 | 76.44±0.16 |
| | DML† | 74.83±0.24 | 73.26±0.21 | 73.15±0.25 | 72.86±0.26 | 67.27±0.28 | 74.02±0.15 | 74.32±0.16 |
| | SHAKE | 76.82±0.12 | 75.62±0.14 | 77.95±0.12 | 74.99±0.18 | 70.18±0.16 | 77.46±0.18 | 78.51±0.16 |
| II | KD+FitNets | 75.12±0.25 | 73.86±0.26 | 74.66±0.24 | 73.22±0.22 | 66.81±0.26 | 74.86±0.28 | 75.15±0.26 |
| | KD+CRD | 75.64±0.24 | 74.38±0.24 | 75.46±0.25 | 74.29±0.28 | 69.54±0.26 | 75.12±0.25 | 76.05±0.29 |
| | SHAKE+FitNets | 76.91±0.12 | 75.73±0.22 | 78.06±0.20 | 75.15±0.18 | 70.23±0.16 | 77.62±0.16 | 78.69±0.14 |
| | SHAKE+CRD | 77.17±0.12 | 75.89±0.16 | 78.13±0.18 | 75.26±0.12 | 70.42±0.18 | 77.86±0.16 | 78.82±0.18 |
| III | KD+Mixup | 76.58±0.24 | 76.10±0.22 | 77.07±0.26 | 75.58±0.19 | 71.29±0.22 | 78.22±0.28 | 79.14±0.18 |
| | KD+CutMixp [36] | 76.81±0.24 | 76.45±0.26 | 76.90±0.24 | 75.50±0.25 | 71.10±0.32 | 77.92±0.19 | 79.53±0.20 |
| | SHAKE+Mixup | 78.20±0.18 | 77.36±0.16 | 79.45±0.15 | 78.32±0.12 | 73.88±0.16 | 79.52±0.18 | 80.86±0.15 |
| | SHAKE+CutMix | 78.45±0.22 | 77.53±0.23 | 79.59±0.16 | 78.56±0.18 | 74.25±0.16 | 79.98±0.24 | 81.22±0.20 |
| IV | KD‡ (AVER) | 75.22±0.24 | 73.92±0.29 | 74.99±0.23 | 74.07±0.28 | 70.21±0.28 | 76.30±0.25 | 75.87±0.28 |
| | KD‡ (AEKD [37]) | 75.68±0.24 | 74.24±0.36 | 75.15±0.32 | 74.11±0.23 | 70.47±0.24 | 76.34±0.22 | 75.95±0.22 |
| | SHAKE‡ (AVER) | 77.32±0.15 | 76.22±0.22 | 78.59±0.28 | 75.60±0.13 | 71.9±0.21 | 78.61±0.22 | 78.98±0.29 |
| | SHAKE‡ (AEKD [37]) | 77.88±0.14 | 76.69±0.18 | 78.90±0.22 | 76.26±0.19 | 72.38±0.25 | 78.94±0.19 | 79.41±0.18 |

**Comparison with offline and online KD methods**. In part I of Table 2, we compare our approach to some advanced offline/online KD methods with the same training settings. For the same architecture style of the teacher-students, SHAKE obtains $3.35\% \sim 5.45\%$ absolute accuracy gains and outperforms KD with $1.41\% \sim 4.62\%$ margins and Review with $0.70\% \sim 2.49\%$ margins. Besides, on cross-architecture teacher-student pairs, SHAKE achieves more significant gains with $5.58\% \sim 6.96\%$ margins than baseline. Compared to other KD methods, SHAKE outperforms KD with $2.83\% \sim 4.06\%$ margins and Review with $0.64\% \sim 2.46\%$ margins, which illustrates the effectiveness of SHAKE in reducing the teacher-student network gap. Compared to DML and AFD [7] under the same teacher-student pair in Table 2, SHAKE obtains $1.21\% \sim 3.65\%$ relative gains. Compare to online KDs (*e.g.*, CL [38], ONE [22]) with multiple branches as teacher models, SHAKE also obtains $0.70\% \sim 1.77\%$ relative accuracy gains. These significant improvements demonstrate the superiority of SHAKE by leveraging the knowledge of pre-trained models.

**Orthogonal to other KDs and data augmentations**. As acting on the output logits, SHAKE is orthogonal to feature and relation KD methods because of transferring different knowledge. As shown in Part II of Table 2, the combination of FitNets with SHAKE surpasses FitNets + KD with $0.98\% \sim 3.54\%$ margins. Moreover, for CRD, its combination with SHAKE yields more dramatic gains than KD. In addition, SHAKE outperforms KD (see part III of Table 2) under strong data augmentation.

**Extension to multiple teacher models.** In the part IV of Table 2, we compare SHAKE and KD with average and adaptive weighting in two teacher networks. The results illustrate that the design of SHAKE with multi-proxy model can effectively inherit the knowledge of different teachers bringing significant performance gains.

### 3.2 Experiments on ImageNet

**Implementation**. For the standard ResNet-18 [14] and MobileNet [17] models, we adopt the same training setting as with most distillation methods, whose training epochs are 100. The multi-step

Table 3: Top-1 accuracies (%) on ImageNet dataset. All results of other methods are reproduced under the same training settings. SHAKE$^{\ddagger}$ denotes SHAKE with two ResNet-34/ResNet-50/RegNetY-16GF as teachers.

| Teacher | Student | KD [16] | ESKD [6] | ATKD [28] | DMLL [48] | CRD [39] | Review [5] | **SHAKE** | **SHAKE$^{\ddagger}$** |
|---|---|---|---|---|---|---|---|---|---|
| ResNet-34 (73.40) | ResNet-18 (69.75) | 70.66 | 70.89 | 70.78 | 71.03 | 71.17 | 71.61 | **72.37** | **72.73** |
| Teacher | Student | KD [16] | AT [47] | OFD [15] | DML [48] | CRD [39] | Review [5] | **SHAKE** | **SHAKE$^{\ddagger}$** |
| ResNet-50 (76.16) | MobileNet (70.13) | 70.68 | 70.72 | 71.25 | 71.13 | 71.40 | 72.56 | **72.96** | **73.42** |
| Teacher | Student | Tf-KD [46] | Soft KD [40] | Hard KD [40] | DeiT [40] | - | - | **SHAKE** | **SHAKE$^{\ddagger}$** |
| RegNetY-16GF (82.90) | ViT-T (72.20) | 72.35 | 72.20 | 74.30 | 74.50 | - | - | **75.22** | **75.92** |

Table 4: Comparison of results of object detection on MS-COCO.

| **Model** | **AP** | **AP$_L$** | **AP$_M$** | **AP$_S$** | **Model** | **AP** | **AP$_L$** | **AP$_M$** | **AP$_S$** |
|---|---|---|---|---|---|---|---|---|---|
| Faster R-CNN-R101 [T] | 42.04 | 54.60 | 45.55 | 25.22 | RetinaNet-R101 [T] | 40.40 | 52.18 | 44.34 | 24.03 |
| Faster R-CNN-R50 [S] | 37.93 | 49.10 | 41.14 | 22.44 | RetinaNet-R50 [S] | 36.15 | 46.95 | 40.25 | 21.37 |
| + KD (logits) | 38.35 | 49.48 | 41.80 | 22.73 | + KD (logits) | 36.76 | 48.17 | 40.56 | 21.87 |
| + Review [5] (feature) | 40.36 | 52.87 | 43.81 | 23.60 | + Review [5] (feature) | 38.48 | 51.15 | 42.72 | 22.67 |
| **+ SHAKE (logits)** | **38.95** | **50.78** | **42.32** | **22.88** | **+ SHAKE (logits)** | **37.35** | **50.35** | **40.88** | **21.61** |
| **+ SHAKE & Review** | **40.67** | **52.96** | **43.92** | **23.85** | **+ SHAKE & Review** | **40.55** | **52.80** | **43.78** | **23.55** |

learning rate is initialized to 0.1, decayed by 0.1 at 30, 60, and 90 epochs. Recent vision transformers achieve great success on different vision tasks [8, 40]. We also extend SHAKE to VIT-T [40] with the same training settings (*e.g.*, data augmentation and distillation token) on ImageNet. Implementation details are available in supplementary materials.

**Comparison results**. Table 3 reports the performance of our approach on ImageNet. SHAKE improves the baseline models of ResNet-18 with $2.62\%$ gains in Top-1 accuracy (see Figure 3 for detailed accuracy curves) and MobileNet with $2.83\%$ gains. Compared to other KD methods, SHAKE outperforms KD with $1.71\% \sim 2.28\%$ margins and CRD with $1.20\% \sim 1.56\%$ margins, which supports the superiority of SHAKE on the large-scale dataset. Equipped with the distillation of two teacher models, SHAKE$^{\ddagger}$ obtains 2.98% gain for ResNet-18 and 3.29% gain for MobileNet than baseline. As shown in Table 3, SHAKE obtains 2.76% accuracy gains than baseline and surpasses KD under the soft or hard label of the CNN teacher (RegNet [31]), verifying its effectiveness on different architectures.

### 3.3 Experiments on object detection

**Implementation**. We evaluate SHAKE on MS-COCO dataset [25] and use the most popular open-source Detectron2 [24] as the strong baseline. We apply SHAKE to two-stage detector (*e.g.*, Faster R-CNN [33]) and the one-stage detector (*e.g.*, RetinaNet [24]), which are widely used object detection frameworks. Following common practice [24], all models are trained with a $2\times$ learning schedule (24 epochs). For SHAKE, we build an extra shadow head with the same architecture as the original classification head, which performs distillation in the detector fine-tuning stage. All distillation performances are evaluated in Average Precision (AP).

**Comparison results**. Table 4 demonstrates the experimental results of the baseline detector and our approach. As shown in Table 4, our SHAKE improves the AP by 1.20 on RetinaNet and 1.02 on Faster R-CNN, which outperform KD [16]. The performance of object detection greatly depends on the quality of deep features to locate interested objects, while logits are not capable of providing knowledge for object localization. Thus, SHAKE naturally weaker than Review (recent state-of-the-art feature-based KDs) and we further introduce Review to obtain satisfactory results. It can be observed that SHAKE can obtain new state-of-the-art results with feature-based KDs. The success of challenging object detection tasks demonstrates the generality and effectiveness of our approach.

### 3.4 Ablation study

In this section, we isolate the impact of each component of our approach. All experiments are conducted on the CIFAR-100 dataset. For each setting, we run 3 times and report Top-1 mean accuracies.

Table 5: Top-1 accuracies (%) of student (S), proxy teachers (T'), and their average ensemble with different settings on CIFAR-100. Note that the ensemble with bring extra latency at inference, while SHAKE only adopts student model for inference without extra costs.

| Method | WRN-16-2 (73.26) | | | VGG8 (70.36) | | |
|---|---|---|---|---|---|---|
| | S | T' | Ensemble | S | T' | Ensemble |
| SHAKE | 76.82 | 76.78 | 76.91 | 74.99 | 74.92 | 75.06 |
| (A) *w/o* Reversed distillation | 74.98 | 74.16 | 75.01 | 73.01 | 71.86 | 73.02 |
| (B) *w/o* Shadow head | 75.08 | 75.08 | 75.08 | 73.12 | 73.12 | 73.12 |
| (C) *w* Stronger shadow head | 76.96 | 76.92 | 75.01 | 75.12 | 75.08 | 75.18 |
| (D) *w/o* Share backbone | 76.36 | 76.22 | 76.45 | 74.35 | 74.28 | 74.41 |
| (E) *w/o* Pre-trained teacher | 74.22 | 73.86 | 74.23 | 71.37 | 71.14 | 71.41 |
| (F) *w* Ground truth supervision for T' | 76.42 | 76.56 | 76.66 | 74.48 | 74.54 | 74.61 |
| (G) *w* Pre-teacher distillation for S | 76.56 | 76.51 | 76.56 | 74.66 | 74.51 | 74.68 |

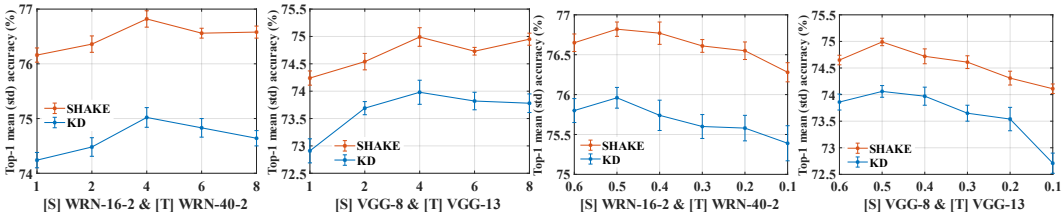


Figure 4: Top-1 mean (std) accuracies (%) of varying $\tau$ for WRN-16-2 & VGG-8 (left) and varying $\lambda$ for WRN-16-2 & VGG-8 (right) on CIFAR-100.

**Detailed ablation study of SHAKE** As shown in Table 5, an ablation study has been conducted to demonstrate the individual effectiveness of different components in SHAKE. It is observed that (a) Reversed distillation optimizes the teacher model to adapt to the student, improving distillation efficiency. SHAKE without reversed distillation only obtains marginal gains. (b) Shadow heads are effective in preserving the diversity of knowledge, and SHAKE brings accuracy reduction if there is no separate shadow head. (c) Stronger shadow heads for storing and teaching more pre-trained knowledge when the capacity gap enlarges. (d) Backbone sharing settings can accelerate the convergence of the student model and results in performance gains. (e) Similar to the self-distillation and regularization methods, SHAKE without pre-trained teacher model also yields slight accuracy gains because of more training weights and logit consistency. (f) Our proxy teacher is present to inherit the pre-training knowledge with its supervision. Under our backbone sharing settings, adding extra supervision from ground truth labels for proxy teacher not conducive to the decoupling of supervision from pre-trained teachers/labels and leads to accuracy reduction. (h) Proxy teachers in our SHAKE to transfer the knowledge of the original pre-trained teacher model for the student model to learn more easily. As shown in the Table 5 (H), the extra distillation from pre-trained teachers to students based on SHAKE does not yield additional significant performance gains. (G) The original pre-teacher model would perform (*i.e.*, KD) weaker than SHAKE for direct distillation because of the gap with the students. As shown in Table 5 (G), addition of pre-trained teachers' supervision for student based on SHAKE does not result in additional accuracy gains. These results demonstrate that our SHAKE proxy teacher design can indeed serve as a good bridge between the original teacher and student. Reverse KD reduces teacher-student gap to facilitate knowledge transfer.

**Sensitivity study for temperature $\tau$ and weight $\lambda$.** $\tau$ in Eq. 3 controls the softness of teacher's supervision. As $\tau$ increases, the output of the softmax function becomes smoother. As shown in Figure 4, SHAKE presents superior gains and robustness than the original KD in different $\tau$. $\lambda$ is the hyper-parameter that balances the weight of KD loss. The small $\lambda$ limits distillation gains, and $\lambda = 0.5$ is the best option for SHAKE.

**Analyzing teacher-student similarity.** The similarity between student and teacher network is an important measure for KD tasks. We employ KL-divergence as metrics of similarity [6], where lower KL-divergence implies higher similarity. Figure 5 presents the similarities and performances between the outputs of ResNet with different depths as teachers (*i.e.*, ResNet-110, ResNet-56, ResNet-44 and ResNet-32) and the ResNet-20 as student. The results show that the distillation from SHAKE always gives a higher similarity than KD and DML, resulting in significant gains under teacher models of different depths. Figure 6 presents the attention map visualization of ResNet-18 with different methods on ImageNet. The attention maps of SHAKE pay more attention to the important regions

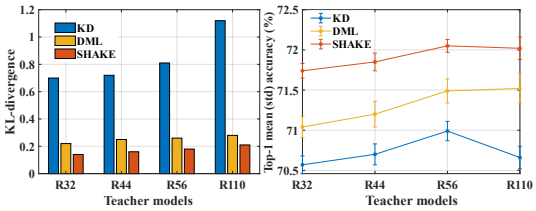


Figure 5: KL-divergence and Top-1 accuracy (%).

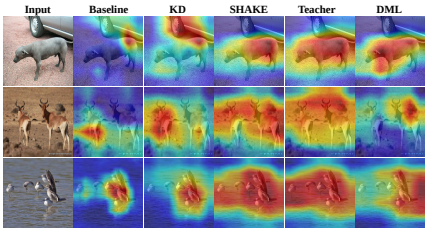


Figure 6: Grad-CAM++ ([3]) visualization.

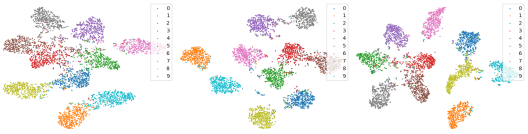

Figure 7: The penultimate layer visualization of ResNet-32 (student) with KD (left), SHAKE (middle) and the teacher (right) on CIFAR-10.

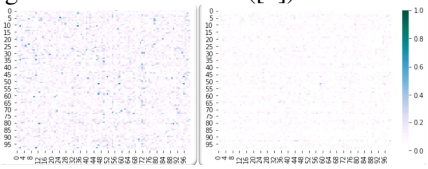

Figure 8: Logits correlation visualization of teacher-student for the student (ResNet-32) with KD (left) and SHAKE (right) on CIFAR-100.

than KD and baseline, which have high similarity with teacher. In summary, SHAKE bridges KD and DML to obtain student-friendly distillations, improving teacher-student similarity. In addition, Figure 7 illustrates that comparing with KD, applying 3 SHAKE training helps learn more scattered embeddings and logits correlation visualization in Figure 8 clearly demonstrate that teacher model in SHAKE owns student-aware logits.

## 4 Related Work

**Offline distillations**. In the offline framework [16], the teacher is pre-trained and fixed, and then its soft logits are used as extra supervision to distill student. Although subsequent methods explored to transfer feature knowledges [35, 47] and relation knowledges [29, 39]), the effective original KD still outperforms most distillations and is widely used for different tasks [10, 19]. However, the pre-trained teacher model is not optimized for the student, limiting its distillation gains. To address this issue, our SHAKE enhances KD using a shadow head to introduce optimization from the student.

**Online distillations**. Online KD methods simplify the KD process by training all models simultaneously. DML [48] performs bidirectional distillation for the peer networks and ONE [22] presents an on-the-fly ensemble distillation among multiple branches. Subsequent studies focus on how to balance multiple teacher [4] or construct the online teacher [44]. Online KD methods always obtain better performance than offline ones. However, large teacher models trained from scratch sometimes perform unstable predictions and bring lots of training costs during distillation. Therefore, we propose SHAKE to combine the advantages of the two pipelines to facilitate application.

**Adaptive distillations**. Capacity gaps between teacher-student models for their different architectures would limit distillation gains [26]. There are two types of existing works to alleviate this gap in terms of training paradigms [9] and architectural adaptation [18, 13]. For instance, ESKD [6] proposes stopping the training of the teacher early, and ATKD [28] uses a medium-size teacher assistant to perform sequence distillation with large overheads. However, these methods do not optimize the teacher model for the student, resulting in minor benefits. In sharp contrast to these methods, SHAKE is a student-aware offline KD method and opens a new direction for adaptive distillation design.

## 5 Conclusion

In this paper, we present SHAKE, a simple, effective, and new student-aware logits distillation to bridge offline and online knowledge transfer. Based on our insight of online KD methods success, SHAKE achieves this goal by building an extra shadow head as a proxy teacher model to perform mutual distillation with the student model. Thorough evaluations are performed on classification and detection, and SHAKE achieves significant performance gains in various neural networks without extra inference overheads. We hope that this elegant and practical approach would inspire future research on knowledge distillation design and understanding.

# References


[1] Lei Jimmy Ba and Rich Caruana. Do deep nets really need to be deep? In *NeurIPS*, 2014.

[2] Tom B. Brown, Benjamin Mann, Nick Ryderand Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M, Ziegler, Jeffrey Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. Language models are few-shot learners. *arXiv preprint, arXiv:2005.14165*, 2020.

[3] Aditya Chattopadhay, Anirban Sarkar, Prantik Howlader, and Vineeth N Balasubramanian. Grad-cam++: Generalized gradient-based visual explanations for deep convolutional networks. In *WACV*, 2018.

[4] Defang Chen, Jian-Ping Mei, Can Wang, Yan Feng, and Chun Chen. Online knowledge distillation with diverse peers. In *AAAI*, 2020.

[5] Pengguang Chen, Shu Liu, Hengshuang Zhao, and Jiaya Jia. Distilling knowledge via knowledge review. In *CVPR*, 2021.

[6] Jang Hyun Cho and Bharath Hariharan. On the efficacy of knowledge distillation. In *ICCV*, 2019.

[7] Inseop Chung, Seonguk Park, J. Kim, and Nojun Kwak. Feature-map-level online adversarial knowledge distillation. In *ICML*, 2020.

[8] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*, 2020.

[9] Mengya Gao, Yujun Shen, Quanquan Li, Junjie Yan, Liang Wan, Dahua Lin, Chen Change Loy, and Xiaoou Tang. An embarrassingly simple approach for knowledge distillation. *arXiv preprint arXiv:1812.01819*, 2018.

[10] Nuno C Garcia, Pietro Morerio, and Vittorio Murino. Modality distillation with multiple stream networks for action recognition. In *ECCV*, 2018.

[11] Ross Girshick, Jeff Donahue, Trevor Darrell, and Jitendra Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. In *CVPR*, 2014.

[12] Jianping Gou, Baosheng Yu, Stephen John Maybank, and Dacheng Tao. Knowledge distillation: A survey. *arXiv:2006.05525*, 2020.

[13] Jindong Gu and Volker Tresp. Search for better students to learn distilled knowledge. *arXiv preprint arXiv:2001.11612*, 2020.

[14] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *CVPR*, 2016.

[15] Byeongho Heo, Jeesoo Kim, Sangdoo Yun, Hyojin Park, Nojun Kwak, and Jin Young Choi. A comprehensive overhaul of feature distillation. In *ICCV*, 2019.

[16] Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*, 2015.

[17] Andrew G. Howard, Menglong Zhu, Bo Chen, Dmitry Kalenichenko, Weijun Wang, Tobias Weyand, Marco Andreetto, and Hartwig Adam. Mobilenets: Efficient convolutional neural networks for mobile vision applications. *arXiv preprint, arXiv:1704.04861*, 2017.

[18] Minsoo Kang, Jonghwan Mun, and Bohyung Han. Towards oracle knowledge distillation with neural architecture search. In *AAAI*, 2020.

[19] Yoon Kim and Alexander M Rush. Sequence-level knowledge distillation. In *EMNLP*, 2016.

[20] Alex Krizhevsky and Geoffrey Hinton. Learning multiple layers of features from tiny images. *Tech Report*, 2009.

[21] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In *NeurIPS*, 2012.

[22] Xu Lan, Xiatian Zhu, and Shaogang Gong. Knowledge distillation by on-the-fly native ensemble. In *NeurIPS*, 2018.

[23] Hao Li, Asim Kadav, Igor Durdanovic, Hanan Samet, and Hans Peter Graf. Pruning filters for efficient convnets. In *ICLR*, 2017.

[24] Tsung-Yi Lin, Priya Goyal, Ross Girshick, Kaiming He, and Piotr Dollar. Focal loss for dense object detection. In *ICCV*, 2017.

[25] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollar, and C Lawrence Zitnick. Microsoft coco: Common objects in context. In *ECCV*, 2014.

[26] Yu Liu, Xuhui Jia, Mingxing Tan, Raviteja Vemulapalli, Yukun Zhu, Bradley Green, and Xiaogang Wang. Search to distill: Pearls are everywhere but not the eyes. In *CVPR*, 2020.

[27] Andrey Malinin, Bruno Mlodozeniec, and Mark Gales. Ensemble distribution distillation. In *ICLR*, 2020.

[28] Seyed Iman Mirzadeh, Mehrdad Farajtabar, Ang Li, Nir Levine, Akihiro Matsukawa, and Hassan Ghasemzadeh. Improved knowledge distillation via teacher assistant. In *AAAI*, 2020.

[29] Wonpyo Park, Yan Lu, Minsu Cho, and Dongju Kim. Relational knowledge distillation. In *CVPR*, 2019.

[30] Hieu Pham, Melody Y. Guan, Barret Zoph, Quoc V. Le, and Jeff Dean. Efficient neural architecture search via parameter sharing. In *ICML*, 2018.

[31] Ilija Radosavovic, Raj Prateek Kosaraju, Ross Girshick, Kaiming He, and Piotr Dollar. Designing network design spaces. In *CVPR*, 2020.

[32] Mohammad Rastegari, Vicente Ordonez, Joseph Redmon, and Ali Joseph. Xnor-net: Imagenet classification using binary convolutional neural networks. In *ECCV*, 2016.

[33] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. *arXiv preprint, arXiv:1506.01497*, 2015.

[34] Carlos Riquelme, Joan Puigcerver, Basil Mustafa, Maxim Neumann, Rodolphe Jenatton, Andre Susano Pinto, Daniel Keysers, and Neil Houlsby. Scaling vision with sparse mixture of experts. *arXiv preprint arXiv:2106.05974*, 2021.

[35] Adriana Romero, Nicolas Ballas, Samira Ebrahimi Kahou, Antoine Chassang, Carlo Gatta, and Yoshua Bengio. Fitnets: Hints for thin deep nets. In *ICLR*, 2015.

[36] Seong Joon Oh Sanghyuk Chun Junsuk Choe Youngjoon Yoo Sangdoo Yun, Dongyoon Han. Cutmix: Regularization strategy to train strong classifiers with localizable features. *ICCV*, 2019.

[37] Xiaojie Li Jianlong Wu Fei Wang Chen Qian Shangchen Du, Shan You and Changshui Zhang. Agree to disagree: Adaptive ensemble knowledge distillation in gradient space. In *NeurIPS*, 2020.

[38] Guocong Song and Wei Chai. Collaborative learning for deep neural networks. In *NeurIPS*, 2018.

[39] Yonglong Tian, Dilip Krishnan, and Phillip Isola. Contrastive representation distillation. In *ICLR*, 2020.

[40] Hugo Touvron, Matthieu Cord, Matthijs Douze, Francisco Massa, Alexandre Sablayrolles, and Herve Jegou. Training data-efficient image transformers amp; distillation through attention. In *ICML*, 2021.

[41] Frederick Tung and Greg Mori. Similarity-preserving knowledge distillation. In *ICCV*, 2019.

[42] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *NeurIPS*, 2017.

[43] Lin Wang and Kuk-Jin Yoon. Knowledge distillation and student-teacher learning for visual intelligence: A review and new outlooks. *T-PAMI*, 2021.

[44] Guile Wu and Shaogang Gong. Peer collaborative learning for online knowledge distillation. *arXiv preprint arXiv:2006.04147*, 2020.

[45] Jiahui Yu and Thomas Huang. Autoslim: Towards one-shot architecture search for channel numbers. *CVPR*, 2019.

[46] Li Yuan, Francis EH Tay, Guilin Li, Tao Wang, and Jiashi Feng. Revisiting knowledge distillation via label smoothing regularization. In *CVPR*, 2020.

[47] Sergey Zagoruyko and Nikos Komodakis. Paying more attention to attention: Improving the performance of convolutional neural networks via attention transfer. In *ICLR*, 2017.

[48] Ying Zhang, Tao Xiang, Timothy M Hospedales, and Huchuan Lu. Deep mutual learning. In *CVPR*, 2018.

[49] Zhilu Zhang and Mert R Sabuncu. Self-distillation as instance-specific label smoothing. *arXiv preprint, arXiv:2006.05065*, 2020.

## Checklist

1. For all authors...
   (a) Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope? [Yes]
   (b) Did you describe the limitations of your work? [Yes] . Our method will add additional training overhead.
   (c) Did you discuss any potential negative societal impacts of your work? [No] Our approach is a new knowledge distillation application technique, which helps lightweight models to achieve better performance without additional teacher architectures. It does not bring negative social impact.
   (d) Have you read the ethics review guidelines and ensured that your paper conforms to them? [Yes]
2. If you are including theoretical results...
   (a) Did you state the full set of assumptions of all theoretical results? [No]
   (b) Did you include complete proofs of all theoretical results? [No]
3. If you ran experiments...
   (a) Did you include the code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL)? [Yes] See implementation details in our experiment Section 3.2 and supplemental materials.
   (b) Did you specify all the training details (e.g., data splits, hyperparameters, how they were chosen)? [Yes]
   (c) Did you report error bars (e.g., with respect to the random seed after running experiments multiple times)? [Yes] See Table 2 and more results.
   (d) Did you include the total amount of compute and the type of resources used (e.g., type of GPUs, internal cluster, or cloud provider)? [Yes] See Table 1.
4. If you are using existing assets (e.g., code, data, models) or curating/releasing new assets...

(a) If your work uses existing assets, did you cite the creators? [Yes]

(b) Did you mention the license of the assets? [No] The code and the data are proprietary.

(c) Did you include any new assets either in the supplemental material or as a URL? [Yes]

(d) Did you discuss whether and how consent was obtained from people whose data you're using/curating? [N/A] We evaluate our approach in a generic public dataset, following lots of previous research works.

(e) Did you discuss whether the data you are using/curating contains personally identifiable information or offensive content? [N/A] The datasets we use does not contain personally identifiable information.

5. If you used crowdsourcing or conducted research with human subjects...

(a) Did you include the full text of instructions given to participants and screenshots, if applicable? [N/A] Our approach is a generic training framework, which does not concern human subjects.

(b) Did you describe any potential participant risks, with links to Institutional Review Board (IRB) approvals, if applicable? [N/A]

(c) Did you include the estimated hourly wage paid to participants and the total amount spent on participant compensation? [N/A]