Knowledge Distillation with the Reused Teacher Classifier

Defang Chen[1,2,3] Jian-Ping Mei[4] Hailin Zhang[1,2,3]
Can Wang[1,2,3*] Yan Feng[1,2,3] Chun Chen[1,2,3]
[1]Zhejiang University [2]Shanghai Institute for Advanced Study of Zhejiang University
[3]ZJU-Bangsun Joint Research Center [4]Zhejiang University of Technology
defchern@zju.edu.cn, jpmei@zjut.edu.cn, {zzzhl, wcan, fengyan, chenc}@zju.edu.cn

Abstract

Knowledge distillation aims to compress a powerful yet cumbersome teacher model into a lightweight student model without much sacrifice of performance. For this purpose, various approaches have been proposed over the past few years, generally with elaborately designed knowledge representations, which in turn increase the difficulty of model development and interpretation. In contrast, we empirically show that a ***simple knowledge distillation*** *technique is enough to significantly narrow down the teacher-student performance gap. We directly reuse the discriminative classifier from the pre-trained teacher model for student inference and train a student encoder through feature alignment with a single ℓ_2 loss. In this way, the student model is able to achieve exactly the same performance as the teacher model provided that their extracted features are perfectly aligned. An additional projector is developed to help the student encoder match with the teacher classifier, which renders our technique applicable to various teacher and student architectures. Extensive experiments demonstrate that our technique achieves state-of-the-art results at the modest cost of compression ratio due to the added projector.*

1. Introduction

Given a powerful teacher model with large numbers of parameters, the goal of knowledge distillation (KD) is to help another less-parameterized student model gain a similar generalization ability as the larger teacher model [4, 24]. A straightforward way to achieve this goal is by aligning their logits or class predictions given the same inputs [2, 24]. Due to its conceptual simplicity and practical effectiveness, KD technique has achieved great success in a variety of applications, such as object detection [8], semantic segmentation [32] and the training of transformers [47].

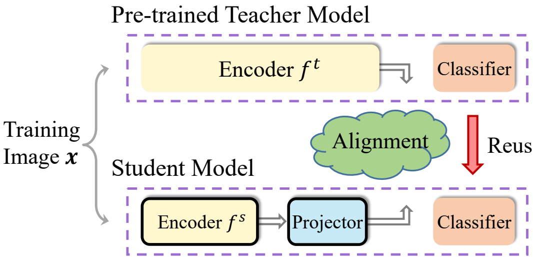


Figure 1. An overview of our proposed SimKD. A simple ℓ_2 loss is adopted for feature alignment in the preceding layer of the final classifier. Only the student feature encoder and dimension projector are updated during training (boxes with the black border). The pre-trained teacher classifier is reused for student inference.

One limitation of the vanilla KD is that the performance gap between the original teacher model and the distilled student model is still significant. To overcome this drawback, a bunch of approaches have been proposed in the last few years [19, 50]. Most of them benefit from exploiting additional supervision from the pre-trained teacher model, especially the intermediate layers [1, 6, 40, 46, 48, 52, 55]. Besides aligning the plain intermediate features [6, 40, 52], the existing efforts are typically based on elaborately designed knowledge representations, such as mimicking spatial attention maps [55], pairwise similarity patterns [36, 37, 48] or maximizing the mutual information between teacher and student features [1, 46, 58]. Although we indeed see constant improvements of these works in student performance, *neither* effective representations *nor* well-optimized hyperparameters ensuring their success are easily achievable in practice. Furthermore, the diversity of transferred knowledge hinders the emergence of a unified and clear interpretation of the final improvement in student performance.

In this paper, we present a simple knowledge distillation technique and demonstrate that it can significantly bridge the performance gap between teacher and student models with no need for elaborate knowledge representations. Our proposed "SimKD" technique is illustrated in Figure 1. We

[*] Corresponding author

argue that the powerful class prediction ability of a teacher model is credited to not only those expressive features but just as importantly, a discriminative classifier. Based on this argument, which is empirically supported later on, we train a student model through feature alignment in the preceding layer of the classifier and directly copy the teacher classifier for student inference. In this way, if we could perfectly align the student features with those of the teacher model, their performance gap will just disappear. That is to say, the feature alignment error alone accounts for the accuracy of student inference, which makes our knowledge transfer more comprehensible. According to our experimental results, a single ℓ_2 loss for feature alignment already works surprisingly well. Such a simple loss saves us from carefully tuning hyper-parameters as previous works do in order to balance the effect of multiple losses [1,6,24,40,46,48,52,55].

As the dimensions of extracted features from teacher and student models usually differ from each other, a projector is thus added after the student feature encoder to remedy this dimension mismatch. This projector generally incurs a less than 3% cost to the pruning ratio in teacher-to-student compression, but it makes our technique applicable to arbitrary model architectures. The pruning ratio could be even enlarged in a few cases where the parameter number of the added projector plus the reused teacher classifier is less than that of the original student classifier (see Figure 7). We conduct extensive experiments on standard benchmark datasets and observe that our SimKD consistently outperforms all compared state-of-the-art approaches with a variety of teacher-student architecture combinations. We also show that our simple technique generalizes well in different scenarios such as multi-teacher knowledge distillation and data-free knowledge distillation.

2. Related Work

Knowledge distillation (KD) is a technique to compress the knowledge from a powerful teacher model, such as an ensemble of multiple deep neural networks, into a smaller student model [4, 19, 24, 50]. The transferred knowledge is initially regarded as the conditional distribution of outputs given input samples [24]. From this viewpoint, the predictions, or soft targets, from the pre-trained teacher model play a major role in the improvement of student performance. A common belief behind the success of this technique is that those teacher-learned soft targets can capture the relationships among different categories and serve as an effective regularization during student training [2,7,24,53].

In order to make KD more practical for model compression, we need to further resist the performance degradation in teacher-to-student compression [19, 50]. Leveraging more information from the pre-trained teacher model especially the intermediate layers is a general solution towards this problem. A bunch of such works have sprung up seeking for better student performance in the last few years, collectively known as *feature distillation*. They mostly propose diverse representations to capture appropriate transferred knowledge, such as the crude intermediate feature maps [40] or their transformations [1, 23, 55], sample relations encoded by the pairwise similarity matrices [36,37,48] or modeled by contrastive learning [46, 51, 58]. More recently, a few works turn to designing cross-layer associations to make full use of those intermediate features of the teacher model [6, 10]. With the help of aforementioned knowledge representations or reformed transfer strategies, the student model will be trained with gradient information coming from not only the final layer, i.e., the classifier, but also from those early layers. However, additional hyper-parameters need careful tuning in these methods to balance the effect of different losses and it is still unclear how the newly introduced supervisory signal would exert positive influence on the final performance of student models.

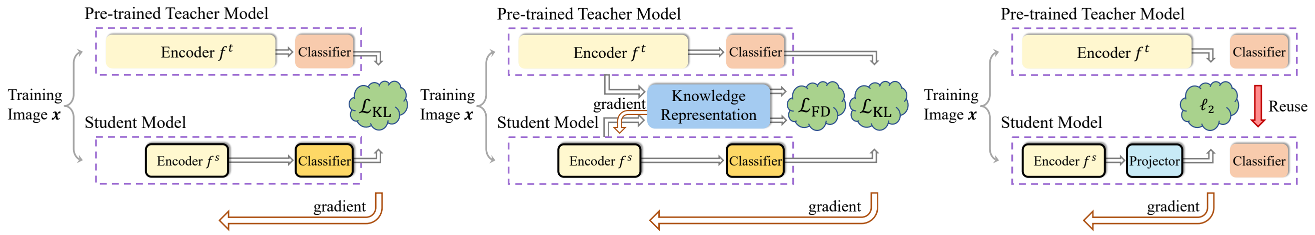
To some extent, our key idea of reusing the teacher classifier is related to the previous studies on hypothesis transfer learning (HTL) [39]. HTL aims to utilize the learned source domain classifier to help the training of the target domain classifier, on the condition that only a small amount of labeled target dataset and no source dataset are accessible [15,28,29]. A recent work further gets rid of the requirement of labeling target dataset and extends the vanilla HTL to the unsupervised domain adaptation setting by resorting to a pseudo-labeling strategy [31]. Different from this one, our goal is to reduce the teacher-student performance gap on the same dataset, rather than adapting the pre-trained model to achieve good performance on another dataset with a different distribution. In addition, our SimKD is much simpler than this work and still achieves surprisingly good results in the standard KD setting.

3. Method

3.1. Vanilla Knowledge Distillation

Generally, the popular deep neural networks designed for image classification tasks in the current era can be regarded as the stack of a *feature encoder* with multiple non-linear layers, together with a *classifier* that usually contains a single fully-connected layer with softmax activation function [22,25,33,42,57]. Both two components will be trained end-to-end with the back-propagation algorithm. The symbolic description is presented as follows.

Given a training sample $\boldsymbol{x}$ with one-hot label $\boldsymbol{y}$ from a K-category classification dataset, we denote the encoded feature in the penultimate layer of the student model as $\boldsymbol{f}^s = \mathcal{F}^s(\boldsymbol{x}; \boldsymbol{\theta}^s) \in \mathbb{R}^{C_s}$. This feature is subsequently passed into the classifier with weight $\boldsymbol{W}^s \in \mathbb{R}^{K \times C_s}$ to obtain the logits $\boldsymbol{g}^s = \boldsymbol{W}^s \boldsymbol{f}^s \in \mathbb{R}^K$ as well as the class prediction $\boldsymbol{p}^s = \sigma(\boldsymbol{g}^s/T) \in \mathbb{R}^K$ with a softmax activation



(a) Vanilla Knowledge Distillation (b) Feature Distillation (c) SimKD

Figure 2. Comparison of three kinds of knowledge distillation techniques. The main differences lie in how the gradient is formalized and where the gradient flow starts. (a) Vanilla KD calculates the gradient in class predictions and relies on this gradient to update the whole student model. (b) Feature distillation gathers more gradient information from the intermediate layers through various knowledge representations. Additional hyper-parameters need to be carefully turned for maximum performance. (c) Our SimKD calculates ℓ_2 loss in the preceding layer of the classifier and back propagates this gradient solely to update the student feature encoder and dimension projector. The cross entropy losses between predictions and ground-truth labels of the compared approaches in (a) and (b) are omitted for simplicity.

function $\sigma(\cdot)$ and the temperature T

$$p_i^s = \frac{\exp\left(g_i^s/T\right)}{\sum_{j=1}^{K}\exp\left(g_j^s/T\right)}, \tag{1}$$

where p_i^s/g_i^s denotes the i-th element of corresponding vectors and T is a hyper-parameter for softening effect[1].

Vanilla knowledge distillation consists of two losses [24]: one is the conventional cross entropy loss and another is the alignment loss in prediction pairs between $\boldsymbol{p}^s$ and soft targets $\boldsymbol{p}^t$ with Kullback-Leibler divergence [27]

$$\mathcal{L}_{\text{KD}} = \underbrace{\mathcal{L}_{\text{CE}}(\boldsymbol{y},\boldsymbol{p}^s)}_{T=1} + \underbrace{T^2\mathcal{L}_{\text{KL}}(\boldsymbol{p}^t,\boldsymbol{p}^s)}_{T>1}. \tag{2}$$

Compared to the cross entropy loss, the introduced prediction alignment loss gives extra information on incorrect classes to facilitate the student training [17,24]. Since probabilities assigned to those incorrect classes tend to be rather small after softmax transformation, the temperature T in this term needs raising to produce softer distributions for conveying more information [24].

3.2. Simple Knowledge Distillation

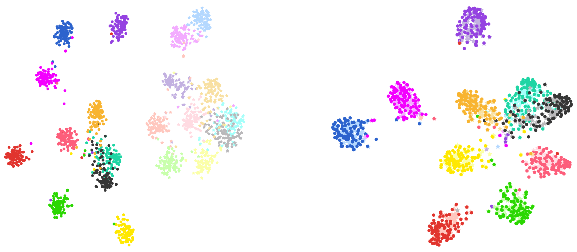
In recent years, various feature distillation approaches have been proposed. These works mainly collect and transmit extra gradient information from intermediate teacher-student layer pairs to train the student feature encoder better (Figure 2b). However, their success heavily depends on those particularly-designed knowledge representations to entail proper inductive bias [3, 6], and carefully chosen hyper-parameters to balance the effect of different losses. Both are labor-intensive and time-consuming. It is also difficult to conclude the actual role that a certain type of representation plays in the student training.

In contrast, we propose a simple knowledge distillation technique named as SimKD, which breaks away from these stringent demands while still achieving state-of-the-art results on extensive experiments. As shown in Figure 2c, a key ingredient of SimKD is the "*classifier-reusing*" operation, *i.e.*, we directly borrow the pre-trained teacher classifier for student inference rather than training a new one. This eliminates the need of label information to calculate the cross entropy loss and makes the feature alignment loss become the only source for generating gradient.

Overall, we argue discriminative information contained in the teacher classifier matters, but has been largely overlooked in the literature of KD. We then provide a plausible explanation for its important role. Consider a situation where one model is requested to handle several tasks with different data distributions, a basic practice is to freeze or share some shallow layers as the feature extractor across different tasks while fine-tuning the last layer to learn *task-specific* information [5, 13, 18, 30]. In this one-model multiple-task setting, existing works hold the opinion that *task-invariant* information could be shared while task-specific information needs to be independently identified, generally by the final classifier. As for KD where teacher and student models with different capabilities are trained on the same dataset, analogously, we could reasonably believe that there is some *capability-invariant* information in the data being easily gained across different models while the powerful teacher model may contain extra essential *capability-specific* information that is hard for a simpler student model to acquire. Furthermore, we hypothesize that most capability-specific information is contained in deep layers and expect that reusing these layers, even only the final classifier will be helpful for the student training.

Based on this hypothesis, which is supported later by empirical evidences from various aspects, we furnish the student model with the teacher classifier for inference and force their extracted features to be matched with the follow-

[1] We only present notations for the student model in this paragraph, but similar notations also hold for the teacher model.



(a) Vanilla KD [24]. (b) Our SimKD.

Figure 3. Visualization results of test images from CIFAR-100 with t-SNE [49]. We randomly sample 10 out of 100 classes. Features extracted by the teacher and student models are depicted with dark and light colors, respectively, and they are almost indistinguishable in our SimKD. Best viewed in color.

ing ℓ_2 loss function

$$\mathcal{L}_{\text{SimKD}} = \|\boldsymbol{f}^t - \mathcal{P}(\boldsymbol{f}^s)\|_2^2, \tag{3}$$

where a projector $\mathcal{P}(\cdot)$ is designed to match the feature dimensions at a relatively small cost while being effective enough to ensure accurate alignment. In effect, this simple loss has already been exploited before [40, 52], but we are actually attempting to reveal the potential value of reusing the teacher classifier rather than developing a sophisticated loss function for feature alignment. As shown in Figure 3, the extracted features from the pre-trained teacher model (dark colors) and the distilled student model in our SimKD (light colors) are compactly clustered within the same class and distinctly separated across different classes, which ensures the student features to be correctly classified latter with the reused teacher classifier.

Somewhat surprisingly, the performance degradation in teacher-to-student compression will be greatly alleviated by this simple technique. Along with high inference accuracy, the simplicity of this single-loss formulation provides our SimKD with good interpretability. Note that the reused part from a pre-trained teacher model is allowed to incorporate more layers but not just limited to the final classifier. Usually, reusing more layers leads to higher student accuracy, but will bring about the burden increase on the inference.

4. Experiments

In this section, we conduct extensive experiments to demonstrate the effectiveness of our proposed SimKD. We first compare it with several representative state-of-the-art approaches on standard benchmark datasets. Some empirical evidences are then given to show the superiority of our "classifier-reusing" operation on the student performance improvement. Although an additional projector is required for our student inference, experiments show that its effect to the pruning ratio could be controlled in an acceptable level. Finally, we employ our technique to the multi-teacher and data-free knowledge distillation settings.

Datasets and baselines. Two benchmark image classification datasets including CIFAR-100 [26] and ImageNet [41] are adopted for a series of experiments. We use the standard data augmentation and normalize all images by channel means and standard deviations as [22, 25, 54]. Besides the vanilla KD [24], various approaches are reproduced for comparison, including FitNet [40], AT [55], SP [48], VID [1], CRD [46], SRRL [52] and SemCKD [6]. All compared approaches except KD itself are implemented incorporating the vanilla KD loss, *i.e.*, Eq. (2).

Training details. We follow the training procedure of previous works [6, 46, 52] and report the performance of all competitors on our randomly associated teacher-student combinations. Specifically, we adopt SGD optimizer with 0.9 Nesterov momentum for all datasets. For CIFAR-100, the total training epoch is set to 240 and the learning rate is divided by 10 at 150th, 180th and 210th epochs. The initial learning rate is set to 0.01 for MobileNet/ShuffleNet-series architectures and 0.05 for other architectures. The mini-batch size is set to 64 and the weight decay is set to 5×10^{-4}. For ImageNet, the initial learning rate is set to 0.1 and then divided by 10 at 30th, 60th, 90th of the total 120 training epochs. The mini-batch size is set to 256 and the weight decay is set to 1×10^{-4}. All results are reported in means (standard deviations) over 4 trials, except for the results on ImageNet are reported in a single trial. The temperature T in the KD loss is set to 4 throughout this paper. *More detailed descriptions for reproducibility as well as more results are included in the technical appendix.*

4.1. Comparison of Test Accuracy

Table 1 to 3 present a comprehensive performance comparison of various approaches based on *fifteen* network combinations, where the teacher and student models are instantiated with similar or completely different architectures.

From the test accuracy comparison in Table 1 and 2, we can see that SimKD consistently outperforms all competitors on CIFAR-100 and the improvements are quite significant in some cases. For example, as for the "ResNet-8x4 & ResNet-32x4" combination, SimKD achieves 3.66% absolute accuracy improvement while the best competitor only achieves 1.81% absolute improvement on the basis of the vanilla KD. Moreover, as shown in the fourth and fifth columns of Table 1, given the same teacher model "ResNet-110x2", SimKD could train a lightweight student model "ResNet-110" with a projector containing 0.05M additional parameters to surpass all the competitors by a considerable margin even when they are employed on a "ResNet-116" containing about more 0.10M parameters than "ResNet-110". Test accuracy for different training epochs in Table 3 show that SimKD achieves faster convergence in training.

Student	WRN-40-1 71.92 ± 0.17	ResNet-8x4 73.09 ± 0.30	ResNet-110 74.37 ± 0.17	ResNet-116 74.46 ± 0.09	VGG-8 70.46 ± 0.29	ResNet-8x4 73.09 ± 0.30	ShuffleNetV2 72.60 ± 0.12
KD [24]	74.12 ± 0.29	74.42 ± 0.05	76.25 ± 0.34	76.14 ± 0.32	72.73 ± 0.15	75.28 ± 0.18	75.60 ± 0.21
FitNet [40]	74.17 ± 0.22	74.32 ± 0.08	76.08 ± 0.13	76.20 ± 0.17	72.91 ± 0.18	75.02 ± 0.31	75.82 ± 0.22
AT [55]	74.67 ± 0.18	75.07 ± 0.03	76.67 ± 0.28	76.84 ± 0.25	71.90 ± 0.13	75.74 ± 0.09	75.41 ± 0.10
SP [48]	73.90 ± 0.17	74.29 ± 0.07	76.43 ± 0.39	75.99 ± 0.26	73.12 ± 0.10	74.84 ± 0.08	75.77 ± 0.08
VID [1]	74.59 ± 0.17	74.55 ± 0.10	76.17 ± 0.22	76.53 ± 0.24	73.19 ± 0.23	75.56 ± 0.13	75.22 ± 0.07
CRD [46]	74.80 ± 0.33	75.59 ± 0.07	76.86 ± 0.09	76.83 ± 0.13	73.54 ± 0.19	75.78 ± 0.27	77.04 ± 0.61
SRRL [52]	74.64 ± 0.14	75.39 ± 0.34	76.75 ± 0.14	77.19 ± 0.09	73.23 ± 0.16	76.12 ± 0.18	76.19 ± 0.35
SemCKD [6]	74.41 ± 0.16	76.23 ± 0.04	76.62 ± 0.14	76.69 ± 0.48	75.27 ± 0.13	75.85 ± 0.16	77.62 ± 0.32
SimKD	**75.56 ± 0.27**	**78.08 ± 0.15**	**77.82 ± 0.15**	**77.90 ± 0.11**	**75.76 ± 0.12**	**76.75 ± 0.23**	**78.39 ± 0.27**
Teacher	WRN-40-2 76.31	ResNet-32x4 79.42	ResNet-110x2 78.18	ResNet-110x2 78.18	ResNet-32x4 79.42	WRN-40-2 76.31	ResNet-32x4 79.42

Table 1. Top-1 test accuracy (%) of various knowledge distillation approaches on CIFAR-100.

Student	ShuffleNetV1 71.36 ± 0.25	WRN-16-2 73.51 ± 0.32	ShuffleNetV2 72.60 ± 0.12	MobileNetV2 65.43 ± 0.29	MobileNetV2x2 69.06 ± 0.10	WRN-40-2 76.35 ± 0.18	ShuffleNetV2x1.5 74.15 ± 0.22
KD [24]	74.30 ± 0.16	74.90 ± 0.29	76.05 ± 0.34	69.07 ± 0.47	72.43 ± 0.32	77.70 ± 0.13	76.82 ± 0.23
FitNet [40]	74.52 ± 0.03	74.70 ± 0.35	76.02 ± 0.21	68.64 ± 0.27	73.09 ± 0.46	77.69 ± 0.23	77.12 ± 0.24
AT [55]	75.55 ± 0.19	75.38 ± 0.18	76.84 ± 0.19	68.62 ± 0.31	73.08 ± 0.14	78.45 ± 0.24	77.51 ± 0.31
SP [48]	74.69 ± 0.32	75.16 ± 0.32	76.60 ± 0.22	68.73 ± 0.17	72.99 ± 0.27	78.34 ± 0.08	77.18 ± 0.19
VID [1]	74.76 ± 0.22	74.85 ± 0.35	76.44 ± 0.32	68.91 ± 0.33	72.70 ± 0.22	77.96 ± 0.33	77.11 ± 0.35
CRD [46]	75.34 ± 0.24	75.65 ± 0.08	76.67 ± 0.27	70.28 ± 0.24	73.67 ± 0.26	78.15 ± 0.14	77.66 ± 0.22
SRRL [52]	75.18 ± 0.39	75.46 ± 0.13	76.71 ± 0.27	69.34 ± 0.16	73.48 ± 0.36	78.39 ± 0.19	77.55 ± 0.26
SemCKD [6]	76.31 ± 0.20	75.65 ± 0.23	77.67 ± 0.30	69.88 ± 0.30	73.98 ± 0.32	78.74 ± 0.17	79.13 ± 0.41
SimKD	**77.18 ± 0.26**	**77.17 ± 0.32**	**78.25 ± 0.24**	**70.71 ± 0.41**	**75.43 ± 0.26**	**79.29 ± 0.11**	**79.54 ± 0.26**
Teacher	ResNet-32x4 79.42	ResNet-32x4 79.42	ResNet-110x2 78.18	WRN-40-2 76.31	ResNet-32x4 79.42	ResNet-32x4 79.42	ResNet-32x4 79.42

Table 2. Top-1 test accuracy (%) of various knowledge distillation approaches on CIFAR-100.

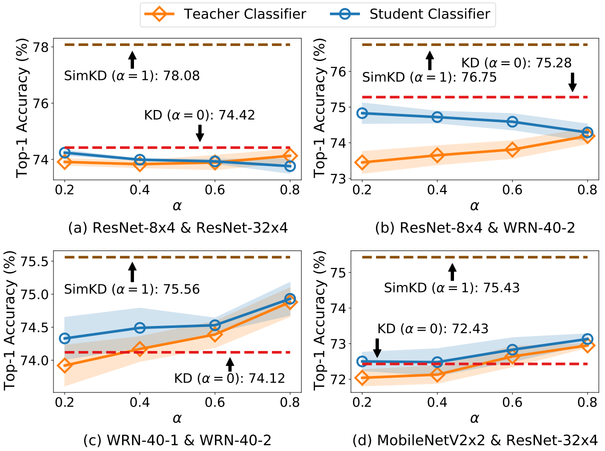
We also find that the student model trained with SimKD yields higher accuracy than its teacher model in the case of "ResNet-8x4 & WRN-40-2" and "ShuffleNetV2 & ResNet-110x2" combinations, which seems a bit confusing since even zero feature alignment loss only guarantees their accuracies to be exactly the same. A possible explanation from self-distillation is that the feature re-representation effect in Equation (3) may help the student model become more robust and thus achieve better results [12, 35].

4.2. Classifier-Reusing Operation Analysis

The "*classifier-reusing*" operation is our recipe for success in above performance comparisons. To better understand its crucial role, we conduct several experiments with two alternative strategies to deal with the student feature encoder and classifier: (1) joint training, (2) sequential training. The performance degradation resulted from these two variants confirms the value of discriminative information in the teacher classifier. Moreover, reusing more deep teacher layers will further improve the student performance.

Joint training. As the previous feature distillation approaches do (Figure 2b), we now train the student feature encoder and its associated classifier jointly. The results are

Feature layer kind training begins here..



(a) ResNet-8x4 & ResNet-32x4

(b) ResNet-8x4 & WRN-40-2

(c) WRN-40-1 & WRN-40-2

(d) MobileNetV2x2 & ResNet-32x4

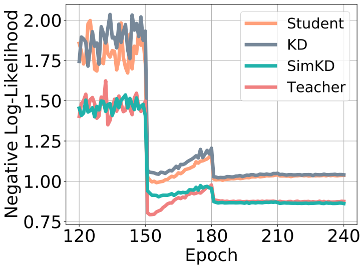
Figure 4. We train the student feature encoder with its associated classifier jointly and then report the test accuracies of student models by using their own classifiers or the reused teacher classifiers.

	Student	KD [24]	AT [55]	SP [48]	VID [1]	CRD [46]	SRRL [52]	SemCKD [6]	SimKD	Teacher
1/4 Epoch	49.34	52.75	52.85	53.57	53.22	55.44	55.14	53.14	**61.73**	54.50
1/2 Epoch	64.98	66.69	66.69	66.36	66.64	67.25	67.36	66.89	**69.26**	70.55
Full Epoch	70.58	71.29	71.18	71.08	71.11	71.25	71.46	71.41	**71.66**	76.26

Table 3. Top-1 test accuracy (%) comparison on ImageNet for different training epochs. We adopt ResNet-18 as the student model.

Student	Sequential	SimKD	Teacher
WRN-40-1	74.48 ± 0.04	75.56 ± 0.27	WRN-40-2
ResNet-8x4	51.97 ± 0.19	78.08 ± 0.15	ResNet-32x4
ResNet-110	77.63 ± 0.05	77.82 ± 0.15	ResNet-110x2
ResNet-116	77.75 ± 0.03	77.90 ± 0.11	ResNet-110x2
VGG-8	35.72 ± 1.33	75.76 ± 0.12	ResNet-32x4
ResNet-8x4	45.03 ± 0.44	76.75 ± 0.23	WRN-40-2
ShuffleNetV2	21.56 ± 0.31	78.39 ± 0.27	ResNet-32x4

Table 4. Training a new student classifier from scratch.



	Accuracy (%)
Student	73.09 ± 0.30
KD [24]	74.42 ± 0.05
SimKD	**78.08 ± 0.15**
SimKD+	**78.47 ± 0.08**
SimKD++	**78.88 ± 0.05**
Teacher	79.42

Figure 5. Comparison of the top-1 test accuracy (%) and negative log-likelihood (Student: ResNet-8x4, Teacher: ResNet-32x4).

obtained by training student models with an extra KD loss

$$\mathcal{L}_{\text{Joint}} = (1-\alpha)\mathcal{L}_{\text{KD}} + \alpha\mathcal{L}_{\text{SimKD}}, \tag{4}$$

where α is a hyper-parameter. To thoroughly assess the joint training effect, four different teacher-student combinations together with four uniformly-spaced α values are used.

As shown in Figure 4, the student performance based on whether its own classifier or the reused teacher classifier becomes far inferior to that of SimKD in all settings, which indicates that discriminative information in the teacher classifier might not be easily transferred into the student model in a joint training way. The substantial accuracy reduction also indicates that the added projector itself and the feature alignment loss do not necessarily improve the final performance, unless we discard joint training and resort to a more effective strategy, *i.e.*, using a single feature alignment loss for training and reusing the teacher classifier for inference. Figure 4 also shows that in order to surpass the performance of the vanilla KD, this two-loss approach requires a case-by-case hyper-parameter tuning.

Sequential training. The above results show the benefit of disassembling the training of student feature encoder and classifier. Additionally, the "*classifier-reusing*" operation carries the implication that a classifier with good discriminative ability is fairly hard to acquire. In this part, we provide evidence for this belief by training a new classifier from scratch rather than reusing the teacher classifier.

We adopt those teacher-student combinations in Table 1 as examples for evaluation. After performing feature alignment with Equation (3), we fix the student feature encoder, *i.e.*, freeze the extracted features, and train a randomly initialized student classifier (a fully-connected layer with softmax activation) with the regular training procedure. This is exactly same as the linear evaluation protocol used in unsupervised learning evaluation [11, 20, 21].

The results of this sequential training are given in Table 4. We find that apart from "WRN-40-1 & WRN-40-2" and "ResNet-110/116 & ResNet-110x2", the test accuracies of other student models appear a precipitous drop. Although we have tried tuning the initial learning rate a few times, it only makes a slight difference in performance. Results in Table 4 indicate that even when the extracted features have been aligned, it is still a challenge to train a satisfactory student classifier. Generally, we could achieve better student performance by tuning hyper-parameters in the classifier training step more carefully, but it is a non-trivial task. In contrast, directly reusing the pre-trained teacher classifier already works quite well. *Detailed training procedure and more results are provided in the appendix.*

Reusing more teacher layers. We now generalize our technique to the situation where more deep layers of the teacher model are reused for student inference and show that the student performance will be further improved.

We take ResNet architecture as an example and conduct experiments on CIAFR-100 dataset. Following the standard design, ResNet architecture consists of one convolutional layer, three building blocks and one fully-connected layer in a bottom-top fashion [22]. Every building blocks contain the same number of convolutional layers and changing these layer numbers leads to different ResNet architectures. For example, 10 layers for each building block make up a 32-layer ResNet model. Then, besides reusing the final classifier as our SimKD do, two new variants are introduced by reusing additional last one or two building blocks, and they are denoted as "SimKD+" and "SimKD++", respectively.

From Figure 5, we can see that SimKD significantly decreases negative log-likelihood by reusing only the teacher classifier, and its two variants further achieve higher performance as expected, though the associated complexity is also increased. These results support our hypothesis that reusing deep teacher layers is beneficial for the student performance improvement, probably due to most capability-specific information is contained in them. Another explanation is that reusing more deep teacher layers would make the approximation of shallow teacher layers easier achievable and thus incur less performance degradation. In practice, reusing only the final teacher classifier strikes a good balance between performance and parameter complexity.

4.3. Projector Analysis

The parameter-free "*classifier-reusing*" operation in our SimKD has been fully evaluated above. Next, we start to dig into another component—*projector* from several aspects. We first present its default implementation and then show that it only requires a small number of extra parameters for achieving state-of-the-art performance. Finally, several ablation studies on the projector are provided.

Implementation. The aim of the projector $\mathcal{P}(\cdot)$ in Equation (3) is to perfectly match the feature vectors $\boldsymbol{f}^t \in \mathbb{R}^{C_t}$ and $\boldsymbol{f}^s \in \mathbb{R}^{C_s}$. A naïve implementation is using one convolutional layer with batch normalization and ReLU activation, which has $C_s \times C_t + 2 \times C_t$ parameters [52]. However, this one-layer transformation may not suffice for accurate alignment due to the large capability gap between teacher and student models. We thus employ the last feature maps and a three-layer bottleneck transformation with dimension reduction factor r as alternatives, hoping that these will help the features aligned better. The total parameters are

$$\frac{C_t(C_s + C_t + 4)}{r} + \frac{9C_t^2}{r^2} + 2C_t. \quad (5)$$

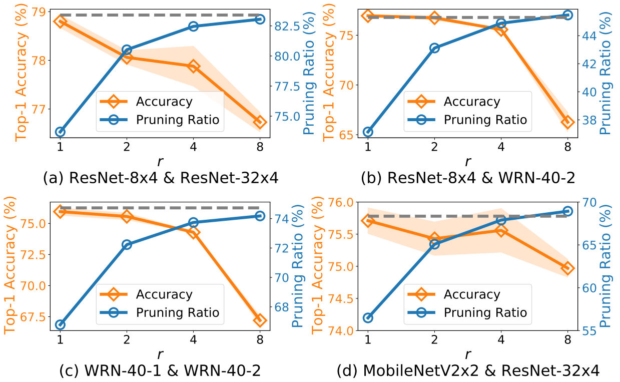
This formula implies that the added parameters will be reduced to between a quarter and a half if r is doubled, which enables us to control the parameter complexity within an acceptable level by changing r. *Detailed structure of the projector and analysis are provided in the technical appendix.*

Effect to pruning ratio. Figure 6 illustrates the trade-off between top-1 test accuracy and pruning ratio with different dimension reduction factor r. We adopt the following equation for the calculation of pruning ratio:

$$\text{Pruning Ratio} = 1 - \frac{\sharp\text{param}_{\text{se}} + \sharp\text{param}_{\text{proj}} + \Delta}{\sharp\text{param}_{\text{t}}} \quad (6)$$

$$\Delta = \sharp\text{param}_{\text{tc}} - \sharp\text{param}_{\text{sc}},$$

where $\sharp\text{param}_{\text{se}}$, $\sharp\text{param}_{\text{proj}}$, $\sharp\text{param}_{\text{t}}$ and $\sharp\text{param}_{\text{tc/sc}}$ refer to the parameter number of a student encoder, a projector, a whole teacher model and a teacher/student classifier, respectively. Its upper bound is approached when $\sharp\text{param}_{\text{proj}} \to 0$, which could be higher than the pruning ratio of the vanilla KD since $\sharp\text{param}_{\text{proj}} + \Delta$ may be less than zero. Figure 6 shows that increasing r will raise the pruning ratio, but in turn, cause the performance drop. This reduction may be attributed to that shrinking the bottleneck dimension of the projector will restrict its representation ability and thus affect the success of feature alignment.



(a) ResNet-8x4 & ResNet-32x4

(b) ResNet-8x4 & WRN-40-2

(c) WRN-40-1 & WRN-40-2

(d) MobileNetV2x2 & ResNet-32x4

Figure 6. Trade-off between test accuracy and pruning ratio. The pruning ratio of the vanilla KD is drawn with the gray dashed line.

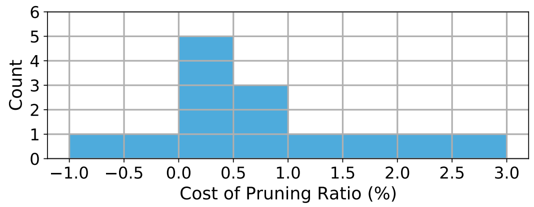


Figure 7. A histogram of the pruning ratio cost.

We then calculate the minimum pruning ratio cost of SimKD when it performs best in the competition on *fourteen* teacher-student combinations from Table 1 and 2. Figure 7 show that our added projector only incurs less than 1% pruning ratio cost in most cases (10/14). In some cases such as "MobileNetV2x2 & ResNet-32x4" and "ShuffleNetV1 & ResNet-32x4" with $r = 8$, we find the pruning ratios of SimKD are even higher than the vanilla KD, and all competitors accordingly. Throughout this paper, we set r equals 2 as default since this value strikes a good balance, i.e., gaining state-of-the-art results at the modest cost of pruning ratio. *The full results are presented in the appendix.*

Ablation study. We finally compare several implementations of the projector and loss function (see Appendix) for feature alignment. All results are obtained with the "ResNet-8x4 & ResNet-32x4" combination on CIFAR-100.

From Table 5, the default implementation of our projector (the last row) achieves the best performance. The accuracy drop resulted from its simplified counterparts indicates the benefit of employing a relatively powerful projector in feature alignment. Moreover, the lower accuracy (76.03 ±

Projector	Test loss (ℓ_2)	Accuracy (%)
1x1Conv	0.345 ± 0.001	75.15 ± 0.27
1x1Conv-1x1Conv	0.343 ± 0.001	75.71 ± 0.33
1x1Conv-3x3Conv (DW)-1x1Conv	0.306 ± 0.001	77.76 ± 0.12
1x1Conv-3x3Conv-1x1Conv	**0.301** ± **0.001**	**78.08** ± **0.15**

Table 5. Comparison of projectors. "1x1/3x3Conv" denotes a convolutional layer with 1x1/3x3 kernel size. "DW" denotes depthwise separable convolutions. Standard batch normalization and ReLU activation are used after each layer.

Method	①	②
Student	72.60 ± 0.12	72.60 ± 0.12
AVEG	75.94 ± 0.20	76.33 ± 0.14
AEKD [14]	75.99 ± 0.18	76.17 ± 0.43
AEKD-F [14]	77.24 ± 0.32	77.08 ± 0.28
SimKD$_v$	**77.43** ± **0.21**	**77.60** ± **0.23**
SimKD	**78.59** ± **0.31**	**78.59** ± **0.05**

Table 6. Results of the multi-teacher KD. We adopt ShuffleNetV2 as the student model and train it under two groups of pre-trained teacher models: ① includes three ResNet-32x4. ② includes two ResNet-32x4 and one ResNet-110x2.

0.40) obtained by aligning feature vectors $\boldsymbol{f}^s$ and $\boldsymbol{f}^t$ with a three-layer transformation validates the effectiveness of using the last feature maps instead. Since our ℓ_2 loss reflects the distance between extracted features, the lower test loss implies the closer alignment and thus the better test accuracy. This is consistent with the results in Table 5.

4.4. Application I: Multi-Teacher Knowledge Distillation

We then demonstrate the applicability of our technique in the multi-teacher KD setting where multiple pre-trained teacher models are available for the student training. Two representative approaches are compared: "AVEG" denotes a simple variant of the vanilla KD, which averages the predictions of multiple teachers; "AEKD" aggregates the teacher predictions with an adaptive weighting strategy and its improved version by incorporating intermediate features is denoted as "AEKD-F" [14]. As shown in Table 6, SimKD always achieves the best performance. Additionally, we provide the results of SimKD$_v$, where a fully-connected layer projector is first used to align the feature vectors and then merged into the associated teacher classifier. The weights of multiple teacher classifiers are averaged and reused for student inference, which incurs no extra parameters.

4.5. Application II: Data-Free Knowledge Distillation

Data-free knowledge distillation aims to exploit a pre-trained teacher model without accessing its training dataset to improve the student performance. A popular paradigm is to recover the original data manifold with a generative model first and then perform knowledge distillation on the synthesized dataset [9, 16, 34]. Our SimKD can be easily integrated into these existing approaches by replacing their KD training step as our "reusing-classifier" operation and the associated feature alignment. Table 7 shows that with the help of our SimKD, the student performance is also improved in the data-free knowledge distillation application.

Method	Require data?	WRN-40-1	WRN-16-2
Student	Yes	71.92 ± 0.17	73.51 ± 0.32
ZSKT [34]	No	33.60 ± 3.88	45.03 ± 1.73
DAFL [9]	No	45.32 ± 1.46	45.94 ± 1.66
CMI [16]	No	64.80 ± 0.35	65.11 ± 0.43
CMI+SimKD	No	**66.78** ± **0.29**	**67.31** ± **0.89**

Table 7. Results of the data-free KD. We adopt WRN-40-2 as the teacher model with two different student models.

5. Conclusion

In this paper, we have explored a simple knowledge distillation technique where the pre-trained teacher classifier is reused for student inference and the student model is trained with a single ℓ_2 loss for feature alignment. We design several experiments to analyze the workings of our technique and conduct extensive experiments to demonstrate its superiority over state-of-the-art approaches. We hope this study will be an important baseline for future research.

6. Limitation and Future Work

A simple parameter reusing is served as our first attempt to explore the potential value of the teacher classifier. This requires a projector when feature dimensions are mismatched and thus increases the model complexity. How to develop a projector-free alternative needs further investigation. Another limitation is that our technique is only applicable for supervised knowledge distillation, such as image classification [24], dense prediction [43] and machine translation [45]. It is also worthwhile to develop a successful variant of our technique for unsupervised learning scenario.

7. Acknowledgment

This work is supported by the Starry Night Science Fund of Zhejiang University Shanghai Institute for Advanced Study (Grant No: SN-ZJU-SIAS-001), National Natural Science Foundation of China (Grant No: U1866602) and Zhejiang Provincial Natural Science Foundation (Grant No: LY20F020023). The authors would like to thank Chunhua Shen and anonymous reviewers for their helpful comments.

A. Experimental Setting

A.1. Datasets and Training Details

We adopt two datasets including CIFAR-100 [26] and ImageNet [41] to conduct experiments. All images are normalized by channel means and standard deviations. A horizontal flip is used for data augmentation. **CIFAR-100**[2] contains 50,000 training images and 10,000 test images from 100 classes. Each training image is padded by 4 pixels on each size and randomly cropped as a 32 × 32 sample. **ImageNet**[3] contains about 1.3 million training images and 50,000 validation images from 1,000 classes. Each image is randomly cropped as a 224x224 sample without padding. The top-1 test accuracy of the teacher model (ResNet-50) is 76.26%.

Multi-Teacher Knowledge Distillation. The training hyper-parameters of multi-teacher KD are exactly the same as those of single-teacher KD on CIFAR-100. We first pre-train multiple teacher models with different initialization and then distill their knowledge into a student model. The accuracies of compared AEKD and AEKD-F [14] are obtained by running a public library[4] with default model hyper-parameters on our teacher-student combinations [56]. The top-1 test accuracy of two groups of teacher models used in our main submission are: ① Three ResNet-32x4 models (79.32, 79.43, 79.45), ② Two ResNet-32x4 models (79.43, 79.45) and one ResNet-110x2 model (78.18).

Data-Free Knowledge Distillation. We adopt a public library[5] to reproduce the results of compared approaches: ZSKT [34], DAFL [9] and CMI [16], with the default model hyper-parameters. In our experiment, the top-1 test accuracy of the teacher model (WRN-40-2) is 76.31%. The performance of the student model trained with original dataset is included for comparison.

Computing Infrastructure. All of the experiments are conducted with PyTorch [38]. CIFAR-100 experiments are conducted on a sever containing eight NVIDIA GeForce RTX 2080Ti GPUs with 11GB RAM. The CUDA version is 11.2. ImageNet experiments are conducted on a sever containing four NVIDIA A40 GPUs with 48GB RAM. The CUDA version is 11.4.

A.2. Network Architectures

We use a large number of teacher-student combinations for performance evaluation, which are composed of several popular neural network architectures: VGG [44], ResNet [22], WRN [54], MobileNetV2 [42], ShuffleNetV1 [57], ShuffleNetV2 [33]. The number behind “VGG-”, “ResNet-” denotes the depth of networks. “WRN-d-w” denotes the wide-ResNet with depth d and width factor w. As the previous works do [6, 46], we expand or shrink the number of convolution filters in intermediate layers of some network architectures with a certain ratio and put that ratio behind “x”, such as “ResNet-32x4”.

Input dimension	Operator	Output dimension
$H \times W \times C_s$	1x1 Conv	$H \times W \times C_t/r$
$H \times W \times C_t/r$	3x3 Conv	$H \times W \times C_t/r$
$H \times W \times C_t/r$	1x1 Conv	$H \times W \times C_t$

Table S.1. Projector structure. “1x1/3x3Conv” denotes a convolutional layer with 1x1/3x3 kernel size. Standard batch normalization and ReLU activation are used after each convolutional layer. r is the reduction ratio.

A.3. Projector

The detailed structure of our used projector is described in Table S.1. We assume that the spatial dimensions of involved feature maps are the same, and denote them with the notations H and W. Otherwise, an average pooling operation is used in advance for spatial dimension alignment to reduce the computational consumption, as the previous work do [6].

Given the feature maps of teacher and student models, the parameter number of the added projector is a function of the dimension reduction factor r

$$\mathcal{F}(r) = \frac{C_t(C_s + C_t + 4)}{r} + \frac{9C_t^2}{r^2} + 2C_t. \tag{7}$$

Proposition. The extra parameter number $\mathcal{F}(r)$ satisfies the inequality $2\mathcal{F}(2r) < \mathcal{F}(r) < 4\mathcal{F}(2r)$ under some mild conditions.

Proof.
We first prove the left part of the inequality:

$$\begin{aligned}
&2\mathcal{F}(2r) < \mathcal{F}(r) \\
2 \times \left(\frac{C_t(C_s + C_t + 4)}{2r} + \frac{9C_t^2}{4r^2} + 2C_t \right) &< \\
\frac{C_t(C_s + C_t + 4)}{r} &+ \frac{9C_t^2}{r^2} + 2C_t \\
2 \times \left(\frac{9C_t^2}{4r^2} + 2C_t \right) &< \frac{9C_t^2}{r^2} + 2C_t \\
2C_t - \frac{9C_t^2}{2r^2} &< 0 \\
2C_t \left(1 - \frac{9C_t}{4r^2} \right) &< 0.
\end{aligned} \tag{8}$$

Generally, the channel dimension C_t in the last feature maps of popular deep neural networks is greater than 128 on

[2] https://www.cs.toronto.edu/~kriz/cifar.html

[3] http://image-net.org/challenges/LSVRC/2012/index

[4] https://github.com/Rorozhl/CA-MKD

[5] https://github.com/zju-vipa/DataFree

Student	VGG-8 70.46 ± 0.29	WRN-16-2 73.51 ± 0.32	WRN-16-4 77.26 ± 0.24
KD [24]	73.38 ± 0.05	75.40 ± 0.34	79.24 ± 0.23
FitNet [40]	73.63 ± 0.11	75.44 ± 0.22	79.06 ± 0.16
AT [55]	73.51 ± 0.08	75.76 ± 0.29	79.38 ± 0.20
SP [48]	73.53 ± 0.23	75.61 ± 0.34	79.53 ± 0.20
VID [1]	73.63 ± 0.07	75.44 ± 0.24	79.40 ± 0.08
CRD [46]	74.31 ± 0.17	75.86 ± 0.17	79.46 ± 0.19
SRRL [52]	74.25 ± 0.35	75.89 ± 0.12	79.67 ± 0.17
SemCKD [6]	74.43 ± 0.25	75.77 ± 0.11	80.05 ± 0.27
SimKD	**74.93** ± **0.21**	**76.23** ± **0.14**	**80.36** ± **0.04**
Teacher	VGG-13 74.64	WRN-40-2 76.31	WRN-40-4 79.51

Table S.2. Top-1 test accuracy (%) comparison on CIFAR-100.

Network	Student	SimKD	Teacher
ResNet-34 & ResNet-50	74.01	**74.64**	76.26
ResNet-50 & ResNet-101	76.26	**77.60**	77.80

Table S.3. Top-1 test accuracy (%) comparison on ImageNet.

CIFAR-100 and is greater than 512 on ImageNet, which means that this equation holds when $r < 16$ and $r < 32$, respectively. This is easy to be satisfied in practice. Since a typical setting for r is 1, 2 and 4 in order to avoid substantial accuracy reduction as shown in Table S.8 and S.9.

We then prove the right part of the inequality:

$$
\begin{aligned}
&4\mathcal{F}(2r) > \mathcal{F}(r) \\
4 \times \left(\frac{C_t(C_s + C_t + 4)}{2r} + \frac{9C_t^2}{4r^2} + 2C_t \right) &> \\
\frac{C_t(C_s + C_t + 4)}{r} + \frac{9C_t^2}{r^2} + 2C_t & \\
4 \times \left(\frac{C_t(C_s + C_t + 4)}{2r} + 2C_t \right) &> \\
\frac{C_t(C_s + C_t + 4)}{r} + 2C_t & \\
2 \times \left(\frac{C_t(C_s + C_t + 4)}{2r} + 3C_t \right) &> 0.
\end{aligned}
\tag{9}
$$

Since the channel dimensions C_t and C_s are always greater than zero, this inequality holds automatically. □

B. More Experimental Results

B.1. Comparison of Test Accuracy

Table S.2 and S.3 presents more results on CIFAR-100 and ImageNet datasets with extra *five* teacher-student combinations. Similar observations are obtained as those in the main submission. For ImageNet dataset, we replace the 3x3 convolution as the 3x3 depth-wise separable convolution in the projector (Table S.1) to control the extra parameters.

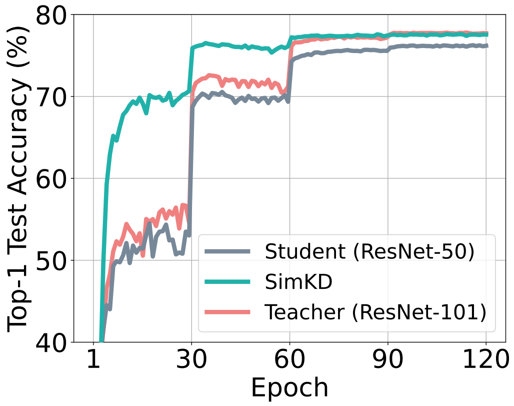


Figure S.1. The test accuracy (%) of ResNet-50 & ResNet-101 on ImageNet. Our SimKD achieves faster model convergence.

As shown in Figure S.1, our SimKD achieves *faster convergence* in the whole model training. For example, at 30th epoch, SimKD performance is on the par with the baseline student model performance at 60th epoch. Besides, at 60th epoch, SimKD already outperforms the baseline student model at 120th epoch.

B.2. Joint Training Results

Table S.4 and S.5 present the full joint training results with different hyper-parameters. In the case of $\alpha = 0$ or $\alpha = 1$, only the student classifier or teacher classifier produces meaningful results and the another one degrades into random guess. We denote these random guess as "–".

B.3. Sequential Training Results

The results of sequential training in the main submission are obtained with the regular training procedure. That is to say, we adopt SGD with 0.9 Nesterov momentum and 5×10^{-4} weight decay. The total training epoch is set to 240 and the learning rate is divided by 10 at 150th, 180th and 210th epochs. The initial learning rate is set to 0.01 for MobileNet/ShuffleNet-series architecture and 0.05 for other architectures. The mini-batch size is set to 64.

Table S.6 gives additional results with different initial learning rates. It is shown that the student accuracy always stays at about 50% when the learning rate ranges from 0.01 to 0.5, which indicates the difficulty of training a satisfactory student classifier from scratch. In contrast, our SimKD achieves **78.08** ± **0.15** test accuracy without any classifier retraining but just reusing the pre-trained teacher classifier.

B.4. Comparison of Loss Function

The default feature alignment loss in our main submission is implemented in the preceding layer of the teacher classifier with a ℓ_2 loss. Its result is reported in the second

Student	ResNet-8x4 73.09 ± 0.30		ResNet-8x4 73.09 ± 0.30	
	Student Classifier	Teacher Classifier	Student Classifier	Teacher Classifier
$\alpha = 0$ (KD)	**74.42 ± 0.05**	–	**75.28 ± 0.18**	–
$\alpha = 0.2$	74.42 ± 0.10	73.91 ± 0.12	74.83 ± 0.29	73.45 ± 0.31
$\alpha = 0.4$	73.99 ± 0.03	73.83 ± 0.13	74.72 ± 0.17	73.65 ± 0.27
$\alpha = 0.6$	73.93 ± 0.08	73.89 ± 0.26	74.59 ± 0.23	73.81 ± 0.25
$\alpha = 0.8$	73.76 ± 0.26	74.13 ± 0.24	74.29 ± 0.24	74.19 ± 0.22
$\alpha = 0.9$	73.52 ± 0.33	74.42 ± 0.26	73.98 ± 0.06	74.73 ± 0.13
$\alpha = 0.99$	30.77 ± 0.92	77.66 ± 0.22	23.46 ± 0.81	76.68 ± 0.13
$\alpha = 0.999$	5.59 ± 0.60	**77.98 ± 0.19**	4.55 ± 0.68	**76.84 ± 0.28**
$\alpha = 1$ (SimKD)	–	**78.08 ± 0.15**	–	**76.75 ± 0.23**
Teacher	ResNet-32x4 79.42		WRN-40-2 76.31	

Table S.4. Joint training the student feature encoder and classifier with different hyper-parameters.

Student	WRN-40-1 71.92 ± 0.17		MobileNetV2x2 69.06 ± 0.10	
	Student Classifier	Teacher Classifier	Student Classifier	Teacher Classifier
$\alpha = 0$ (KD)	**74.12 ± 0.29**	–	**72.43 ± 0.32**	–
$\alpha = 0.2$	74.33 ± 0.32	73.92 ± 0.31	72.50 ± 0.26	72.04 ± 0.23
$\alpha = 0.4$	74.49 ± 0.30	74.17 ± 0.19	72.48 ± 0.38	72.13 ± 0.25
$\alpha = 0.6$	74.53 ± 0.11	74.39 ± 0.20	72.83 ± 0.34	72.64 ± 0.29
$\alpha = 0.8$	74.93 ± 0.25	74.88 ± 0.22	73.13 ± 0.15	72.95 ± 0.06
$\alpha = 0.9$	75.18 ± 0.20	75.17 ± 0.17	73.44 ± 0.26	73.40 ± 0.27
$\alpha = 0.99$	74.17 ± 0.06	75.35 ± 0.15	74.91 ± 0.30	75.31 ± 0.16
$\alpha = 0.999$	18.39 ± 1.45	75.33 ± 0.08	13.49 ± 1.80	**75.43 ± 0.22**
$\alpha = 1$ (SimKD)	–	**75.56 ± 0.27**	–	**75.43 ± 0.26**
Teacher	WRN-40-2 76.31		ResNet-32x4 79.42	

Table S.5. Joint training the student feature encoder and classifier with different hyper-parameters.

Learning Rate	Test Accuracy
0.01	52.03 ± 0.15
0.05	**51.97 ± 0.19**
0.1	52.01 ± 0.17
0.5	51.93 ± 0.20

Table S.6. Training a new classifier from scratch with different initial learning rates (Student: ResNet-8x4, Teacher: ResNet-32x4).

column of Table S.7. Another implementation is to calculate the loss in the succeeding layer of the teacher classifier with a loss function $\|\boldsymbol{W}^t\boldsymbol{f}^t - \boldsymbol{W}^t\mathcal{P}(\boldsymbol{f}^s)\|_2^2$, and we report its results in the third column of Table S.7. From Table S.7, we find that our default feature alignment loss performs best. Moreover, the gradient comparison of different loss functions indicates that the effect of "Output (ℓ_2)" is to calibrate the gradient of "Input (ℓ_2)" with a symmetric matrix $\mathbf{W}^{t\mathrm{T}}\mathbf{W}^t$.

B.5. Comparison of Pruning Ratio

Table S.8 and S.9 present the top-1 test accuracy and the cost of pruning ratio (the first element in parenthesis) of SimKD versus different dimension reduction factors. We also provide the ratio of the projector parameters to the student parameters (the second element in parenthesis) for comparison. We make those results bold when SimKD

	Input (ℓ_2)	Output (ℓ_2)	Input (ℓ_2) + Output (ℓ_2)
Accuracy	**78.08 ± 0.15**	77.09 ± 0.09	77.88 ± 0.30
Loss function	$\|\boldsymbol{f}^t - \boldsymbol{f}^s\|_2^2$	$\|\boldsymbol{W}^t\boldsymbol{f}^t - \boldsymbol{W}^t\boldsymbol{f}^s\|_2^2$	$\|\boldsymbol{f}^t - \boldsymbol{f}^s\|_2^2 + \|\boldsymbol{W}^t\boldsymbol{f}^t - \boldsymbol{W}^t\boldsymbol{f}^s\|_2^2$
Gradient on $\boldsymbol{f}^s$	$-2\left(\boldsymbol{f}^t - \boldsymbol{f}^s\right)$	$-2\mathbf{W}^{t^{\mathrm{T}}}\mathbf{W}^t\left(\boldsymbol{f}^t - \boldsymbol{f}^s\right)$	$-2\{(\boldsymbol{I} + \boldsymbol{W}^{t^{\mathrm{T}}}\boldsymbol{W}^t)(\boldsymbol{f}^t - \boldsymbol{f}^s)\}$

Table S.7. Comparison of different loss functions (Student: ResNet-8x4, Teacher: ResNet-32x4). We omit the projector $\mathcal{P}(\cdot)$ for simplicity.

Student	WRN-40-1 71.92 ± 0.17	ResNet-8x4 73.09 ± 0.30	ResNet-110 74.37 ± 0.17	ResNet-116 74.46 ± 0.09	VGG-8 70.46 ± 0.29	ResNet-8x4 73.09 ± 0.30	ShuffleNetV2 72.60 ± 0.12
$r = 8$	67.20 ± 0.35 (0.55%, 1.05%)	**76.73** ± **0.20** (**0.35%**, 2.11%)	71.71 ± 1.00 (0.18%, 0.35%)	71.96 ± 1.09 (0.18%, 0.33%)	**74.74** ± **0.15** (**0.90%**, 0.86%)	66.26 ± 0.98 (-0.17%, 0.73%)	77.49 ± 0.31 (-0.35%, 3.76%)
$r = 4$	74.29 ± 0.03 (0.99%, 2.81%)	**77.88** ± **0.41** (**0.94%**, 5.67%)	**77.14** ± **0.22** (**0.32%**, 0.92%)	**77.18** ± **0.21** (**0.32%**, 0.87%)	**75.62** ± **0.28** (**1.62%**, 2.19%)	75.57 ± 0.03 (0.41%, 1.78%)	**78.21** ± **0.20** (**0.58%**, 8.85%)
$r = 2$	**75.56** ± **0.27** (**2.5%**, 8.77%)	**78.08** ± **0.15** (**2.88%**, 17.34%)	**77.82** ± **0.15** (**0.82%**, 2.88%)	**77.90** ± **0.11** (**0.82%**, 2.73%)	**75.76** ± **0.12** (**2.98%**, 6.23%)	**76.75** ± **0.23** (**2.18%**, 5.02%)	**78.39** ± **0.27** (**3.16%**, 23.01%)
$r = 1$	75.95 ± 0.30 (7.95%, 30.35%)	78.80 ± 0.13 (9.71%, 58.51%)	**78.00** ± **0.26** (**2.6%**, 9.96%)	**78.15** ± **0.30** (**2.6%**, 9.43%)	75.98 ± 0.21 (11.05%, 19.87%)	76.96 ± 0.07 (8.16%, 15.96%)	78.66 ± 0.08 (11.33%, 67.77%)
Teacher	WRN-40-2 76.31	ResNet-32x4 79.42	ResNet-110x2 78.18	ResNet-110x2 78.18	ResNet-32x4 79.42	WRN-40-2 76.31	ResNet-32x4 79.42

Table S.8. Top-1 test accuracy (%) and pruning ratio (the first element in parenthesis) of SimKD with various dimension reduction factor r on CIFAR-100. We also provide the ratio of the projector parameters to the student parameters (the second element in parenthesis).

achieves state-of-the-art performance and the added projector only requires less than or about 3% pruning ratio cost.

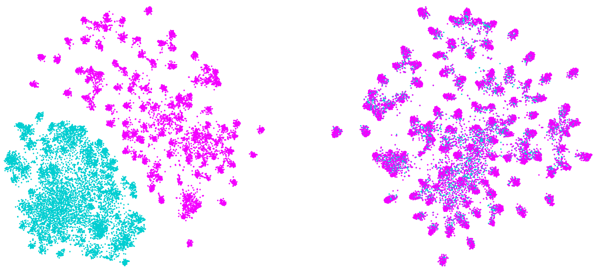
In some cases such as "MobileNetV2x2 & ResNet-32x4" and "ShuffleNetV1 & ResNet-32x4" with $r = 8$, we can see that the pruning ratios of SimKD are even higher than the vanilla KD training, and all competitors accordingly. Moreover, SimKD achieves the second best performance on "ShuffleNetV2 & ResNet-32x4" with $r = 8$ (SimKD: 77.49%, the best performance is achieved by SemCKD: 77.62%), "ShuffleNetV2x1.5 & ResNet-32x4" with $r = 8$ (SimKD: 78.96%, the best performance is achieved by SemCKD: 79.13%), and "ShuffleNetV2 & ResNet-110x2" with $r = 4$, (SimKD: 77.35%, the best performance is achieved by SemCKD: 77.67%). Although the projector needs retaining during the whole training and test stages, a series of trade-off experiments between test accuracy and pruning ratio show that the extra parameters it brought are negligible in most cases.

We further extend our technique to the situation where more deep teacher layers are reused for student inference and analyze the accompanying trade-off between accuracy enhancement and complexity increase. As shown in Table S.10, "SimKD+" and "SimKD++" achieve higher performance than "SimKD" but they also bring about a sharp drop of the pruning ratio, which indicates that simply reusing the final teacher classifier strikes a good balance between performance and parameter complexity.

B.6. Visualization

We adopt ResNet-8x4 as the student model and ResNet-32x4 as the teacher model for visualization experiments.

Ten randomly selected classes in the main submission includes "road", "bee" , "lawn_mower", "bottle", "shrew", "bridge", "man", "mouse", "sweet_pepper" and "cattle". We further visualize all 100 classes on CIFAR-100 with t-SNE in Figure S.2. The visualization results show that with the help of a simple ℓ_2 loss, the extracted features from teacher and student models become almost indistinguishable in SimKD, which ensures the student features to be correctly classified with the reused teacher classifier later.



(a) Vanilla KD [24]. (b) Our SimKD.

Figure S.2. Visualizations of all test images from CIFAR-100 with t-SNE [49]. Features extracted by the teacher and student models are depicted with magenta and cyan colors, respectively, and they are almost indistinguishable in our SimKD. Best viewed in color.

Student	ShuffleNetV1 71.36 ± 0.25	WRN-16-2 73.51 ± 0.32	ShuffleNetV2 72.60 ± 0.12	MobileNetV2 65.43 ± 0.29	MobileNetV2x2 69.06 ± 0.10	WRN-40-2 76.35 ± 0.18	ShuffleNetV2x1.5 74.15 ± 0.22
$r = 8$	**76.68 ± 0.20** (**-0.29%**, 5.16%)	75.41 ± 0.17 (0.47%, 3.13%)	73.50 ± 0.77 (-0.99%, 1.55%)	61.78 ± 1.21 (-4.00%, 3.08%)	**74.97 ± 0.14** (**-0.58%**, 2.51%)	78.55 ± 0.33 (0.13%, 0.98%)	78.96 ± 0.15 (-0.35%, 1.98%)
$r = 4$	**77.22 ± 0.22** (**0.60%**, 12.12%)	**76.69 ± 0.23** (**1.01%**, 8.81%)	77.35 ± 0.18 (-0.63%, 3.39%)	69.43 ± 0.21 (-2.67%, 6.77%)	**75.56 ± 0.34** (**0.45%**, 5.78%)	**79.23 ± 0.06** (**0.27%**, 2.75%)	**79.48 ± 0.12** (**0.58%**, 4.65%)
$r = 2$	**77.18 ± 0.26** (**3.14%**, 32.03%)	**77.17 ± 0.32** (**2.84%**, 28.13%)	**78.25 ± 0.24** (**0.31%**, 8.19%)	**70.71 ± 0.41** (**0.52%**, 15.62%)	**75.43 ± 0.26** (**3.26%**, 14.66%)	**79.29 ± 0.11** (**0.76%**, 8.78%)	**79.54 ± 0.26** (**3.16%**, 12.09%)
$r = 1$	77.58 ± 0.36 (11.20%, 95.15%)	77.65 ± 0.24 (9.45%, 98.01%)	78.58 ± 0.22 (2.99%, 21.83%)	70.90 ± 0.17 (9.43%, 40.34%)	75.71 ± 0.20 (11.87%, 41.86%)	79.26 ± 0.17 (2.55%, 30.59%)	79.72 ± 0.24 (11.33%, 35.61%)
Teacher	ResNet-32x4 79.42	ResNet-32x4 79.42	ResNet-110x2 78.18	WRN-40-2 76.31	ResNet-32x4 79.42	ResNet-110x4 80.20	ResNet-32x4 79.42

Table S.9. Top-1 test accuracy (%) and pruning ratio (the first element in parenthesis) of SimKD with various dimension reduction factor r on CIFAR-100. We also provide the ratio of the projector parameters to the student parameters (the second element in parenthesis).

	Test Accuracy	Pruning Ratio
Student	73.09 ± 0.30	83.40%
SimKD	**78.08 ± 0.15**	**80.52%**
SimKD+	**78.47 ± 0.08**	19.21%
SimKD++	**78.88 ± 0.05**	15.97%
Teacher	79.42	0%

Table S.10. Comparison of reusing different teacher layers.

References

[1] Sungsoo Ahn, Shell Xu Hu, Andreas C. Damianou, Neil D. Lawrence, and Zhenwen Dai. Variational information distillation for knowledge transfer. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 9163–9171, 2019. 1, 2, 4, 5, 6, 10

[2] Jimmy Ba and Rich Caruana. Do deep nets really need to be deep? In *Advances in Neural Information Processing Systems*, pages 2654–2662, 2014. 1, 2

[3] Yoshua Bengio, Aaron C. Courville, and Pascal Vincent. Representation learning: A review and new perspectives. *IEEE Trans. Pattern Anal. Mach. Intell.*, 35(8):1798–1828, 2013. 3

[4] Cristian Bucilua, Rich Caruana, and Alexandru Niculescu-Mizil. Model compression. In *Proceedings of the 12th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 535–541, 2006. 1, 2

[5] Rich Caruana. Multitask learning. *Machine Learning*, 28(1):41–75, 1997. 3

[6] Defang Chen, Jian-Ping Mei, Yuan Zhang, Can Wang, Zhe Wang, Yan Feng, and Chun Chen. Cross-layer distillation with semantic calibration. In *Proceedings of the AAAI Conference on Artificial Intelligence*, pages 7028–7036, 2021. 1, 2, 3, 4, 5, 6, 9, 10

[7] Defang Chen, Jian-Ping Mei, Can Wang, Yan Feng, and Chun Chen. Online knowledge distillation with diverse peers. In *Proceedings of the AAAI Conference on Artificial Intelligence*, pages 3430–3437, 2020. 2

[8] Guobin Chen, Wongun Choi, Xiang Yu, Tony X. Han, and Manmohan Chandraker. Learning efficient object detection models with knowledge distillation. In *Advances in Neural Information Processing Systems*, pages 742–751, 2017. 1

[9] Hanting Chen, Yunhe Wang, Chang Xu, Zhaohui Yang, Chuanjian Liu, Boxin Shi, Chunjing Xu, Chao Xu, and Qi Tian. Data-free learning of student networks. In *International Conference on Computer Vision*, pages 3513–3521, 2019. 8, 9

[10] Pengguang Chen, Shu Liu, Hengshuang Zhao, and Jiaya Jia. Distilling knowledge via knowledge review. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5008–5017, 2021. 2

[11] Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey E. Hinton. A simple framework for contrastive learning of visual representations. In *International Conference on Machine Learning*, pages 1597–1607, 2020. 6

[12] Xiang Deng and Zhongfei Zhang. Learning with retrospection. In *Proceedings of the AAAI Conference on Artificial Intelligence*, pages 7201–7209, 2021. 5

[13] Jeff Donahue, Yangqing Jia, Oriol Vinyals, Judy Hoffman, Ning Zhang, Eric Tzeng, and Trevor Darrell. Decaf: A deep convolutional activation feature for generic visual recognition. In *International Conference on Machine Learning*, volume 32, pages 647–655, 2014. 3

[14] Shangchen Du, Shan You, Xiaojie Li, Jianlong Wu, Fei Wang, Chen Qian, and Changshui Zhang. Agree to disagree: Adaptive ensemble knowledge distillation in gradient space. In *Advances in Neural Information Processing Systems*, 2020. 8, 9

[15] Simon S. Du, Jayanth Koushik, Aarti Singh, and Barnabás Póczos. Hypothesis transfer learning via transformation functions. In *Advances in Neural Information Processing Systems*, pages 574–584, 2017. 2

[16] Gongfan Fang, Jie Song, Xinchao Wang, Chengchao Shen, Xingen Wang, and Mingli Song. Contrastive model inversion for data-free knolwedge distillation. In *Proceedings of the International Joint Conference on Artificial Intelligence*, pages 2374–2380, 2021. 8, 9

[17] Tommaso Furlanello, Zachary Chase Lipton, Michael Tschannen, Laurent Itti, and Anima Anandkumar. Born-

again neural networks. In *International Conference on Machine Learning*, pages 1602–1611, 2018. 3

[18] Ross B. Girshick, Jeff Donahue, Trevor Darrell, and Jitendra Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. In *Proceedings of the European Conference on Computer Vision*, pages 580–587, 2014. 3

[19] Jianping Gou, Baosheng Yu, Stephen J. Maybank, and Dacheng Tao. Knowledge distillation: A survey. *Int. J. Comput. Vis.*, 129(6):1789–1819, 2021. 1, 2

[20] Jean-Bastien Grill, Florian Strub, Florent Altché, Corentin Tallec, Pierre H. Richemond, Elena Buchatskaya, Carl Doersch, Bernardo Ávila Pires, Zhaohan Guo, Mohammad Gheshlaghi Azar, Bilal Piot, Koray Kavukcuoglu, Rémi Munos, and Michal Valko. Bootstrap your own latent - A new approach to self-supervised learning. In *Advances in Neural Information Processing Systems*, 2020. 6

[21] Kaiming He, Haoqi Fan, Yuxin Wu, Saining Xie, and Ross B. Girshick. Momentum contrast for unsupervised visual representation learning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 9726–9735, 2020. 6

[22] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 770–778, 2016. 2, 4, 6, 9

[23] Byeongho Heo, Minsik Lee, Sangdoo Yun, and Jin Young Choi. Knowledge transfer via distillation of activation boundaries formed by hidden neurons. In *Proceedings of the AAAI Conference on Artificial Intelligence*, pages 3779–3787, 2019. 2

[24] Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. Distilling the knowledge in a neural network, 2015. 1, 2, 3, 4, 5, 6, 8, 10, 12

[25] Gao Huang, Zhuang Liu, Laurens Van Der Maaten, and Kilian Q Weinberger. Densely connected convolutional networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4700–4708, 2017. 2, 4

[26] Alex Krizhevsky and Geoffrey Hinton. Learning multiple layers of features from tiny images. *Technical Report*, 2009. 4, 9

[27] Solomon Kullback and Richard A Leibler. On information and sufficiency. *The Annals of Mathematical Statistics*, 22(1):79–86, 1951. 3

[28] Ilja Kuzborskij and Francesco Orabona. Stability and hypothesis transfer learning. In *International Conference on Machine Learning*, pages 942–950, 2013. 2

[29] Ilja Kuzborskij and Francesco Orabona. Fast rates by transferring from auxiliary hypotheses. *Mach. Learn.*, 106(2):171–195, 2017. 2

[30] Zhizhong Li and Derek Hoiem. Learning without forgetting. *IEEE Trans. Pattern Anal. Mach. Intell.*, 40(12):2935–2947, 2018. 3

[31] Jian Liang, Dapeng Hu, and Jiashi Feng. Do we really need to access the source data? source hypothesis transfer for unsupervised domain adaptation. In *International Conference on Machine Learning*, pages 6028–6039, 2020. 2

[32] Yifan Liu, Ke Chen, Chris Liu, Zengchang Qin, Zhenbo Luo, and Jingdong Wang. Structured knowledge distillation for semantic segmentation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2604–2613, 2019. 1

[33] Ningning Ma, Xiangyu Zhang, Hai-Tao Zheng, and Jian Sun. Shufflenet V2: practical guidelines for efficient CNN architecture design. In *Proceedings of the European Conference on Computer Vision*, pages 122–138, 2018. 2, 9

[34] Paul Micaelli and Amos J. Storkey. Zero-shot knowledge transfer via adversarial belief matching. In *Advances in Neural Information Processing Systems*, pages 9547–9557, 2019. 8, 9

[35] Hossein Mobahi, Mehrdad Farajtabar, and Peter L. Bartlett. Self-distillation amplifies regularization in hilbert space. In *Advances in Neural Information Processing Systems*, 2020. 5

[36] Wonpyo Park, Dongju Kim, Yan Lu, and Minsu Cho. Relational knowledge distillation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3967–3976, 2019. 1, 2

[37] Nikolaos Passalis and Anastasios Tefas. Learning deep representations with probabilistic knowledge transfer. In *European Conference on Computer Vision*, pages 283–299, 2018. 1, 2

[38] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Köpf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. Pytorch: An imperative style, high-performance deep learning library. In *Advances in Neural Information Processing Systems*, pages 8024–8035, 2019. 9

[39] Ievgen Redko, Emilie Morvant, Amaury Habrard, Marc Sebban, and Younès Bennani. A survey on domain adaptation theory: learning bounds and theoretical guarantees, 2020. 2

[40] Adriana Romero, Nicolas Ballas, Samira Ebrahimi Kahou, Antoine Chassang, Carlo Gatta, and Yoshua Bengio. Fitnets: Hints for thin deep nets. In *International Conference on Learning Representations*, 2015. 1, 2, 4, 5, 10

[41] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael S. Bernstein, Alexander C. Berg, and Fei-Fei Li. Imagenet large scale visual recognition challenge. *International Journal of Computer Vision*, 115(3):211–252, 2015. 4, 9

[42] Mark Sandler, Andrew G. Howard, Menglong Zhu, Andrey Zhmoginov, and Liang-Chieh Chen. Mobilenetv2: Inverted residuals and linear bottlenecks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4510–4520, 2018. 2, 9

[43] Changyong Shu, Yifan Liu, Jianfei Gao, Zheng Yan, and Chunhua Shen. Channel-wise knowledge distillation for dense prediction. In *International Conference on Computer Vision*, pages 5291–5300, 2021. 8

[44] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. In *In-*

ternational Conference on Learning Representations, 2015. 9

[45] Xu Tan, Yi Ren, Di He, Tao Qin, Zhou Zhao, and Tie-Yan Liu. Multilingual neural machine translation with knowledge distillation. In *International Conference on Learning Representations*, 2019. 8

[46] Yonglong Tian, Dilip Krishnan, and Phillip Isola. Contrastive representation distillation. In *International Conference on Learning Representations*, 2020. 1, 2, 4, 5, 6, 9, 10

[47] Hugo Touvron, Matthieu Cord, Matthijs Douze, Francisco Massa, Alexandre Sablayrolles, and Hervé Jégou. Training data-efficient image transformers & distillation through attention. In *International Conference on Machine Learning*, pages 10347–10357, 2021. 1

[48] Frederick Tung and Greg Mori. Similarity-preserving knowledge distillation. In *International Conference on Computer Vision*, pages 1365–1374, 2019. 1, 2, 4, 5, 6, 10

[49] Laurens van der Maaten and Geoffrey Hinton. Visualizing data using t-sne. *Journal of Machine Learning Research*, 9(86):2579–2605, 2008. 4, 12

[50] Lin Wang and Kuk-Jin Yoon. Knowledge distillation and student-teacher learning for visual intelligence: A review and new outlooks. *IEEE Trans. Pattern Anal. Mach. Intell.*, 2021. 1, 2

[51] Guodong Xu, Ziwei Liu, Xiaoxiao Li, and Chen Change Loy. Knowledge distillation meets self-supervision. In *European Conference on Computer Vision*, pages 588–604, 2020. 2

[52] Jing Yang, Brais Martínez, Adrian Bulat, and Georgios Tzimiropoulos. Knowledge distillation via softmax regression representation learning. In *International Conference on Learning Representations*, 2021. 1, 2, 4, 5, 6, 7, 10

[53] Li Yuan, Francis EH Tay, Guilin Li, Tao Wang, and Jiashi Feng. Revisiting knowledge distillation via label smoothing regularization. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2020. 2

[54] Sergey Zagoruyko and Nikos Komodakis. Wide residual networks. In *Proceedings of the British Machine Vision Conference*, 2016. 4, 9

[55] Sergey Zagoruyko and Nikos Komodakis. Paying more attention to attention: improving the performance of convolutional neural networks via attention transfer. In *International Conference on Learning Representations*, 2017. 1, 2, 4, 5, 6, 10

[56] Hailin Zhang, Defang Chen, and Can Wang. Confidence-aware multi-teacher knowledge distillation. *arXiv preprint arXiv:2201.00007*, 2021. 9

[57] Xiangyu Zhang, Xinyu Zhou, Mengxiao Lin, and Jian Sun. Shufflenet: An extremely efficient convolutional neural network for mobile devices. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 6848–6856, 2018. 2, 9

[58] Sheng Zhou, Yucheng Wang, Defang Chen, Jiawei Chen, Xin Wang, Can Wang, and Jiajun Bu. Distilling holistic knowledge with graph neural networks. In *International Conference on Computer Vision*, pages 10367–10376, 2021. 1, 2