Decoupled Knowledge Distillation

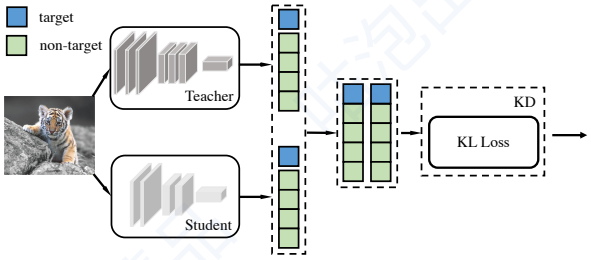
Borui Zhao[1] Quan Cui[2] Renjie Song[1] Yiyu Qiu[1,3] Jiajun Liang[1]
[1]MEGVII Technology [2]Waseda University [3]Tsinghua University
zhaoborui.gm@gmail.com, cui-quan@toki.waseda.jp,
chouyy18@mails.tsinghua.edu.cn, {songrenjie, liangjiajun}@megvii.com

Abstract

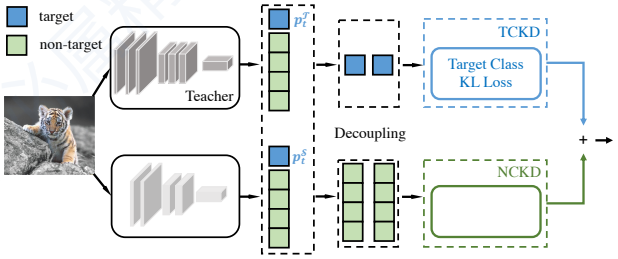
State-of-the-art distillation methods are mainly based on distilling deep features from intermediate layers, while the significance of logit distillation is greatly overlooked. To provide a novel viewpoint to study logit distillation, we reformulate the classical KD loss into two parts, i.e.*, target class knowledge distillation (TCKD) and non-target class knowledge distillation (NCKD). We empirically investigate and prove the effects of the two parts: TCKD transfers knowledge concerning the "difficulty" of training samples, while NCKD is the prominent reason why logit distillation works. More importantly, we reveal that the classical KD loss is a coupled formulation, which (1)* ***suppresses the effectiveness of NCKD*** *and (2)* ***limits the flexibility to balance these two parts****. To address these issues, we present Decoupled Knowledge Distillation (DKD), enabling TCKD and NCKD to play their roles more efficiently and flexibly. Compared with complex feature-based methods, our DKD achieves comparable or even better results and has better training efficiency on CIFAR-100, ImageNet, and MS-COCO datasets for image classification and object detection tasks. This paper proves the great potential of logit distillation, and we hope it will be helpful for future research. The code is available at https://github.com/megvii-research/mdistiller.*

1. Introduction

In the last decades, the computer vision field has been revolutionized by deep neural networks (DNN), which successfully boost various real-scenario tasks, *e.g.*, image classification [9, 13, 21], objection detection [8, 27], and semantic segmentation [31, 45]. However, powerful networks normally benefit from large model capacities, introducing high computational and storage costs. Such costs are not preferable in industrial applications, where lightweight models are widely deployed. In the literature, a potential direction of cutting down the costs is knowledge distillation (KD).



(a) Classical Knowledge Distillation (KD).



(b) Decoupled Knowledge Distillation (DKD).

Figure 1. Illustration of the classical KD [12] and **our DKD**. We reformulate KD into a weighted sum of two parts, *i.e.*, TCKD and NCKD. The first equation shows that KD (1) couples NCKD with $p_t^{\mathcal{T}}$ (the teacher's confidence on the target class), and (2) couples the importance of two parts. Furthermore, we demonstrate that the first coupling suppresses the effectiveness, and the second limits the flexibility for knowledge transfer. We propose DKD to address these issues, which employs hyper-parameters α for TCKD and β for NCKD, killing the two birds with one stone.

KD represents a series of methods concentrating on transferring knowledge from a heavy model (teacher) to a light one (student), which can improve the light model's performance without introducing extra costs.

The concept of KD was firstly proposed in [12] to transfer the knowledge via minimizing the KL-Divergence between prediction *logits* of teachers and students (Figure 1a). Since [28], most of the research attention has been drawn to distill knowledge from deep *features* of intermediate layers. Compared with logits-based methods, the performance

of feature distillation is superior on various tasks, so research on logit distillation has been barely touched. However, training costs of feature-based methods are unsatisfactory, because extra computational and storage usage are introduced (*e.g.*, network modules and complex operations) for distilling deep features during training time.

Logit distillation requires marginal computational and storage costs, but the performance is inferior. Intuitively, logit distillation should achieve comparable performance as feature distillation, since logits are in higher semantic level than deep features. We suppose that the potential of logit distillation is limited by unknown reasons, causing the unsatisfactory performance. To revitalize logits-based methods, we start this work by delving into the mechanism of KD. Firstly, we divide a classification prediction into two levels: (1) a binary prediction for the target class and *all the non-target classes* and (2) a multi-category prediction for *each non-target class*. Based on this, we reformulate the classical KD loss [12] into two parts, as shown in Figure 1b. One is a binary logit distillation for the target class and the other is a multi-category logit distillation for non-target classes. For simplification, we respectively name them as target classification knowledge distillation (TCKD) and non-target classification knowledge distillation (NCKD). The reformulation allows us to study the effects of the two parts independently.

TCKD transfers knowledge via binary logit distillation, which means only the prediction of the target class is provided while the specific prediction of each non-target class is unknown. A reasonable hypothesis is that TCKD transfers knowledge about the "difficulty" of training samples, *i.e.*, the knowledge describes how difficult it is to recognize each training sample. To validate this, we design experiments from three aspects to increase the "difficulty" of training data, *i.e.*, stronger augmentation, noisier label and inherently challenging dataset.

NCKD only considers the knowledge among non-target logits. Interestingly, we empirically prove that only applying NCKD achieves comparable or even better results than the classical KD, indicating the vital importance of knowledge contained in non-target logits, which could be the prominent "dark knowledge".

More importantly, our reformulation demonstrates that the classical KD loss is a highly coupled formulation (as shown in Figure 1b), which could be the reason why the potential of logit distillation is limited. Firstly, the NCKD loss term is weighted by a coefficient that negatively correlates with the teacher's prediction confidence on the target class. Thus larger prediction scores would lead to smaller weights. The coupling significantly suppresses the effects of NCKD on well-predicted training samples. Such suppression is not preferable since *the more confident the teacher is in the training sample, the more reliable and valuable knowledge it could provide*. Secondly, the significance of TCKD and NCKD are coupled, *i.e.*, weighting TCKD and NCKD separately is not allowed. Such limitation is not preferable since *TCKD and NCKD should be separately considered since their contributions are from different aspects*.

To address these issues, we propose a flexible and efficient logit distillation method named Decoupled Knowledge Distillation (DKD, Figure 1b). DKD decouples the NCKD loss from the coefficient negatively correlated with the teacher's confidence by replacing it with a constant value, improving the distillation effectiveness on well-predicted samples. Meanwhile, NCKD and TCKD are also decoupled so that their importance can be separately considered by adjusting the weight of each part.

Overall, our contributions are summarized as follows:

- We provide an insightful view to study logit distillation by dividing the classical KD into TCKD and NCKD. Additionally, the effects of both parts are respectively analyzed and proved.
- We reveal limitations of the classical KD loss caused by its highly coupled formulation. Coupling NCKD with the teacher's confidence suppresses the effectiveness of knowledge transfer. Coupling TCKD with NCKD limits the flexibility to balance the two parts.
- We propose an effective logit distillation method named DKD to overcome these limitations. DKD achieves state-of-the-art performances on various tasks. We also empirically validate the higher training efficiency and better feature transferability of DKD compared with feature-based distillation methods.

2. Related work

The concept of knowledge distillation (KD) was firstly proposed by Hinton *et al*. in [12]. KD defines a learning manner where a bigger teacher network is employed to guide the training of a smaller student network for many tasks [12, 17, 18]. The "dark knowledge" is transferred to students via soft labels from teachers. For raising the attention on negative logits, the hyper-parameter temperature was introduced. The following works can be divided into two types, distillation from logits [3, 6, 22, 40, 44] and intermediate features [10, 11, 14, 15, 23, 25, 28, 33, 34, 41, 43].

Previous works of logit distillation mainly focus on proposing effective regularization and optimization methods rather than novel methods. DML [44] proposes a mutual learning manner to train students and teachers simultaneously. TAKD [22] introduces an intermediate-sized network named "teacher assistant" to bridge the gap between teachers and students. Besides, several works also focus on interpreting the classical KD method [2, 26].

State-of-the-art methods are mainly based on intermediate features, which can directly transfer representations from the teacher to the student [10, 11, 28] or transfer the

correlation between samples captured in the teacher to the student [23, 33, 34]. Most of the feature-based methods could achieve preferable performances (significant higher than logits-based methods), yet involving considerably high computational and storage costs.

This paper focuses on analyzing what limits the potential of logits-based methods and revitalizing logit distillation.

3. Rethinking Knowledge Distillation

In this section, we delve into the mechanism of knowledge distillation. We reformulate KD loss into a weighted sum of two parts, one is relevant to the target class, and the other is not. We explore the effect of each part in the knowledge distillation framework and reveal some limitations of the classical KD. Inspired by the findings, we further propose a novel logit distillation method, achieving remarkable performance on various tasks.

3.1. Reformulating KD

Notations. For a training sample from the t-th class, the classification probabilities can be denoted as $\mathbf{p} = [p_1, p_2, ..., p_t, ..., p_C] \in \mathbb{R}^{1\times C}$, where p_i is the probability of the i-th class and C is the number of classes. Each element in $\mathbf{p}$ can be obtained by the softmax function:

$$p_i = \frac{\exp(z_i)}{\sum_{j=1}^{C}\exp(z_j)}, \tag{1}$$

where z_i represents the logit of the i-th class.

To separate the predictions relevant and irrelevant to the target class, we define the following notations. $\mathbf{b} = [p_t, p_{\setminus t}] \in \mathbb{R}^{1\times 2}$ represents the *binary probabilities* of the target class (p_t) and all the other non-target classes ($p_{\setminus t}$), which can be calculated by:

$$p_t = \frac{\exp(z_t)}{\sum_{j=1}^{C}\exp(z_j)}, p_{\setminus t} = \frac{\sum_{k=1,k\neq t}^{C}\exp(z_k)}{\sum_{j=1}^{C}\exp(z_j)}.$$

Meanwhile, we declare $\hat{\mathbf{p}} = [\hat{p}_1, ..., \hat{p}_{t-1}, \hat{p}_{t+1}, ..., \hat{p}_C] \in \mathbb{R}^{1\times(C-1)}$ to independently model probabilities among non-target classes (*i.e.*, without considering the t-th class). Each element is calculated by:

$$\hat{p}_i = \frac{\exp(z_i)}{\sum_{j=1,j\neq t}^{C}\exp(z_j)}. \tag{2}$$

Reformulation. In this part[1], we attempt to reformulate KD with the binary probabilities $\mathbf{b}$ and the probabilities among non-target classes $\hat{\mathbf{p}}$. $\mathcal{T}$ and $\mathcal{S}$ denote the teacher and the student, respectively. The classical KD uses KL-Divergence as the loss function, which can be written as[2]:

$$\begin{aligned}\mathrm{KD} &= \mathrm{KL}(\mathbf{p}^{\mathcal{T}}||\mathbf{p}^{\mathcal{S}}) \\ &= p_t^{\mathcal{T}}\log(\frac{p_t^{\mathcal{T}}}{p_t^{\mathcal{S}}}) + \sum_{i=1,i\neq t}^{C} p_i^{\mathcal{T}}\log(\frac{p_i^{\mathcal{T}}}{p_i^{\mathcal{S}}}).\end{aligned} \tag{3}$$

According to Eqn.(1) and Eqn.(2) we have $\hat{p}_i = p_i/p_{\setminus t}$, so we can rewrite Eqn.(3) as:

$$\begin{aligned}\mathrm{KD} &= p_t^{\mathcal{T}}\log(\frac{p_t^{\mathcal{T}}}{p_t^{\mathcal{S}}}) + p_{\setminus t}^{\mathcal{T}}\sum_{i=1,i\neq t}^{C}\hat{p}_i^{\mathcal{T}}(\log(\frac{\hat{p}_i^{\mathcal{T}}}{\hat{p}_i^{\mathcal{S}}}) + \log(\frac{p_{\setminus t}^{\mathcal{T}}}{p_{\setminus t}^{\mathcal{S}}})) \\ &= \underbrace{p_t^{\mathcal{T}}\log(\frac{p_t^{\mathcal{T}}}{p_t^{\mathcal{S}}}) + p_{\setminus t}^{\mathcal{T}}\log(\frac{p_{\setminus t}^{\mathcal{T}}}{p_{\setminus t}^{\mathcal{S}}})}_{\mathrm{KL}(\mathbf{b}^{\mathcal{T}}||\mathbf{b}^{\mathcal{S}})} + \underbrace{p_{\setminus t}^{\mathcal{T}}\sum_{i=1,i\neq t}^{C}\hat{p}_i^{\mathcal{T}}\log(\frac{\hat{p}_i^{\mathcal{T}}}{\hat{p}_i^{\mathcal{S}}})}_{\mathrm{KL}(\hat{\mathbf{p}}^{\mathcal{T}}||\hat{\mathbf{p}}^{\mathcal{S}})}.\end{aligned} \tag{4}$$

Then, Eqn.(4) can be rewritten as:

$$\mathrm{KD} = \mathrm{KL}(\mathbf{b}^{\mathcal{T}}||\mathbf{b}^{\mathcal{S}}) + (1-p_t^{\mathcal{T}})\mathrm{KL}(\hat{\mathbf{p}}^{\mathcal{T}}||\hat{\mathbf{p}}^{\mathcal{S}}) \tag{5}$$

As reflected by Eqn.(5), the KD loss is reformulated into a weighted sum of two terms. $\mathrm{KL}(\mathbf{b}^{\mathcal{T}}||\mathbf{b}^{\mathcal{S}})$ represents the similarity between the teacher's and student's binary probabilities of the target class. Thus, we name it Target Class Knowledge Distillation(TCKD). Meanwhile, $\mathrm{KL}(\hat{\mathbf{p}}^{\mathcal{T}}||\hat{\mathbf{p}}^{\mathcal{S}})$ represents the similarity between the teacher's and student's probabilities among non-target classes, named Non-Target Class Knowledge Distillation(NCKD). Eqn.(5) could be rewritten as:

$$\mathrm{KD} = \mathrm{TCKD} + (1-p_t^{\mathcal{T}})\mathrm{NCKD}. \tag{6}$$

Obviously, the weight of NCKD is coupled with $p_t^{\mathcal{T}}$.

The reformulation above inspires us to investigate the *individual* effects of TCKD and NCKD, which will reveal the limitations of the classical coupled formulation.

3.2. Effects of TCKD and NCKD

Performance gain of each part. We individually study the effects of TCKD and NCKD on CIFAR-100 [16]. ResNet [9], WideResNet (WRN) [42] and ShuffleNet [21] are selected as training models, among which both the same and different architectures are considered. The experimental results are reported in Table 1. For each teacher-student pair, we report the results of (1) the student baseline (vanilla training), (2) the classical KD (where TCKD and NCKD are both used), (3) singly TCKD and (4) singly NCKD. The weight of each loss is set as 1.0 (including the default cross-entropy loss). Other implementation details are the same as those in Sec 4.

Intuitively, TCKD concentrates on the knowledge related to the target class since the corresponding loss function considers only binary probabilities. Conversely, NCKD focuses on the knowledge among non-target classes. We notice that

[1] More mathematical formulations are in the supplement.

[2] We omit the temperature (T) in [12] without loss of generality

student	TCKD	NCKD	top-1	Δ
ResNet32×4 as the teacher				
ResNet8×4			72.50	-
	✓	✓	73.63	+1.13
	✓		68.63	-3.87
		✓	74.26	+1.76
ShuffleNet-V1			70.50	-
	✓	✓	74.29	+3.79
	✓		70.52	+0.02
		✓	74.91	+4.41
WRN-40-2 as the teacher				
WRN-16-2			73.26	-
	✓	✓	74.96	+1.70
	✓		70.96	-2.30
		✓	74.76	+1.50
ShuffleNet-V1			70.50	-
	✓	✓	74.92	+4.42
	✓		70.62	+0.12
		✓	75.12	+4.62

Table 1. Accuracy(%) on the CIFAR-100 validation set. Δ represents the performance improvement over the baseline.

singly applying TCKD could be unhelpful (*e.g.*, 0.02% and 0.12% gain on ShuffleNet-V1) or even harmful (*e.g.*, 2.30% drop on WRN-16-2 and 3.87% drop on ResNet8×4) for the student. However, the distillation performances of NCKD are comparable and even better than the classical KD (*e.g.*, 1.76% *vs*. 1.13% on ResNet8×4). The ablation results suggest that the target-class-related knowledge could not be as important as knowledge among non-target classes. To dive into this phenomenon, we provide further analyses presented as follows.

TCKD transfers the knowledge concerning the "difficulty" of training samples. According to Eqn.(5), TCKD transfers "dark knowledge" via the binary classification task, which could be related to the sample "difficulty". For instance, a training sample with $p_t^{\mathcal{T}} = 0.99$ could be "easier" for the student to learn compared with another one with $p_t^{\mathcal{T}} = 0.75$. Since TCKD conveys the "difficulty" of training samples, we suppose the effectiveness would be revealed when the training data becomes challenging. However, the CIFAR-100 training set is easy to fit[3]. Thus the knowledge of "difficulty" provided by the teacher is not informative. In this part, experiments from three perspectives are performed to validate: *The more difficult the training data is, the more benefits TCKD could provide*[4].

(1) Applying Strong Augmentation is a straightforward way to increase the difficulty of training data. We train a ResNet32×4 model as the teacher with AutoAugment [5] on CIFAR-100, achieving 81.29% top-1 validation accuracy. As for students, we train ResNet8×4 and ShuffleNet-V1 models with/without TCKD. Results in Table 2 reveal that TCKD obtains significant performance gains if strong augmentations are applied.

student	TCKD	top-1	Δ
ResNet8×4		73.82	-
	✓	75.33	+1.51
ShuffleNet-V1		77.13	-
	✓	77.98	+0.85

Table 2. Accuracy(%) on the CIFAR-100 validation. We set ResNet32×4 as the teacher and ResNet8×4 as the student. Both teachers and students are trained with AutoAugment [5].

noisy ratio	TCKD	top-1	Δ
0.1		70.99	-
	✓	70.96	-0.03
0.2		67.55	-
	✓	68.03	+0.48
0.3		64.62	-
	✓	65.26	+0.64

Table 3. Accuracy(%) on the CIFAR-100 validation with different noisy ratios on the training set. We set ResNet32×4 as the teacher and ResNet8×4 as the student.

(2) Noisy Labels can also increase the difficulty of training data. We train ResNet32×4 models as teachers and ResNet8×4 as students on CIFAR-100 with {0.1, 0.2, 0.3} symmetric noisy ratios, following [7, 35]. As reported in Table 3, the results indicate that TCKD achieves more performance promotions on noisier training data.

(3) Challenging Datasets (*e.g.*, ImageNet [29]) are also considered. It shows that TCKD could bring +0.32% performance gain on ImageNet in Table 4.

TCKD	top-1	Δ
	70.71	-
✓	71.03	+0.32

Table 4. Accuracy(%) on the ImageNet validation. We set ResNet-34 as the teacher and ResNet-18 as the student.

Conclusively, we demonstrate the effectiveness of TCKD by experimenting with various strategies to increase the difficulty of training data (*e.g.* strong augmentation, noisy labels, difficult tasks). The results validate that the knowledge concerning the "difficulty" of training samples could be more beneficial when distilling knowledge on more challenging training data.

NCKD is the prominent reason why logit distillation works but is greatly suppressed. Interestingly, we notice in Table 1 when only NCKD is applied, the performances are comparable or even better than the classical KD. It shows that the knowledge among non-target classes is of vital importance to logit distillation, which can be the prominent "dark knowledge". However, by reviewing Eqn.(5), we notice that the NCKD loss is coupled with $(1 - p_t^{\mathcal{T}})$,

[3]Training accuracies on CIFAR-100 could be 100% after convergence.

[4]All experiments from these perspectives are performed with NCKD, since we suppose that TCKD should not be singly employed according to the results in Table 1. The probable reasons and analyzes are attached in the supplement.

where $p_t^{\mathcal{T}}$ represents the teacher's confidence on the target class. Therefore, more confident predictions result in smaller NCKD weights. We suppose that *the more confident the teacher is in the training sample, the more reliable and valuable knowledge it could provide.* However, the loss weights are highly suppressed by such confident predictions. We suppose that this fact would limit the effectiveness of knowledge transfer, which is firstly investigated thanks to our reformulation of KD in Eqn. (5).

We design an ablation experiment to verify that well-predicted samples do transfer better knowledge than the others. Firstly we rank the training samples according to $p_t^{\mathcal{T}}$, and evenly split them into two sub-sets. For clarity, one sub-set includes samples with top-50% $p_t^{\mathcal{T}}$ while remaining samples are in the other sub-set. Then we train student networks with NCKD on each subset to compare the performance gain (while the cross-entropy loss is still on the whole set). Table 5 shows that utilizing NCKD on the top-50% samples achieves better performance, suggesting that the knowledge of well-predicted samples is richer than others. However, the loss weight of well-predicted samples are suppressed by the high confidence of the teacher.

0-50%	50-100%	top-1
✓	✓	74.26
✓		74.23
	✓	73.96

Table 5. Accuracy(%) on the CIFAR-100 validation set. We set ResNet32×4 as the teacher and ResNet8×4 as the student.

3.3. Decoupled Knowledge Distillation

So far, we have reformulated the classical KD loss into a weighted sum of two independent parts, and further validated the effectiveness of TCKD and revealed the suppression of NCKD. Specifically, TCKD transfers knowledge concerning the "difficulty" of training samples. More significant improvements could be obtained by TCKD on more challenging training data. NCKD transfers knowledge among non-target classes, which would be suppressed in the condition that the weight $(1-p_t^{\mathcal{T}})$ is relatively small.

Instinctively, both TCKD and NCKD are indispensable and crucial. However, in the classical KD formulation, TCKD and NCKD are coupled from the following aspects:

- For one thing, NCKD is coupled with $(1-p_t^{\mathcal{T}})$, which could suppress NCKD on the well-predicted samples. Since results in Table 5 show that well-predicted samples could bring more performance gain, the coupled form could limit the effectiveness of NCKD.
- For another, weights of NCKD and TCKD are coupled under the classical KD framework. It's not allowed to change each term's weight to balance the importance. We suppose that TCKD and NCKD should be separately considered since their contributions are from different aspects.

Algorithm 1 Pseudo code of DKD in a PyTorch-like style.

```
# l_stu: student output logits
# l_tea: teacher output logits
# T: temperature for KD & DKD
# t: labels, (N, C), bool type
# alpha, beta: hyper-parameters for DKD

p_stu = F.softmax(l_stu / T)
p_tea = F.softmax(l_tea / T)
# pt & pnt: (N, 1), Eqn.(2)
pt_stu, pnt_stu = p_stu[t], p_stu[1-t].sum(1)
pt_tea, pnt_tea = p_tea[t], p_tea[1-t].sum(1)
# pnct: (N, C-1), Eqn.(3)
pnct_stu = F.softmax(l_stu[1-t]/T)
pnct_tea = F.softmax(l_tea[1-t]/T)

# TCKD
tckd = kl_div(log(pt_stu), pt_tea) +
       kl_div(log(pnt_stu), pnt_tea)
# NCKD
nckd = F.kl_div(log(pnct_stu), pnct_tea)

# ori KD
# kd_loss = (tckd + pnt_tea*nckd) * T**2
# DKD
dkd_loss = (alpha*tckd + beta*nckd) * T**2
```

Benefiting from our reformulation of KD, we propose a novel logit distillation method named Decoupled Knowledge Distillation(DKD) to address the above issues. Our proposed DKD independently considers TCKD and NCKD in a decoupled formulation, as shown in Figure 1b. Specifically, we introduce two hyper-parameters α and β, as the weights of TCKD and NCKD, respectively. The loss function of DKD can be written as follows:

$$\text{DKD} = \alpha\text{TCKD} + \beta\text{NCKD}. \tag{7}$$

In DKD, $(1-p_t^{\mathcal{T}})$, which would suppress NCKD's effectiveness, is replaced by β. What's more, it's allowed to adjust α and β to balance the importance of TCKD and NCKD. Through decoupling NCKD and TCKD, DKD provides an efficient and flexible manner for logit distillation. Algorithm 1 provides the pseudo-code of DKD in a PyTorch-like [24] style.

4. Experiments

We mainly experiment on two representative tasks, *i.e.*, image classification and object detection, including:

CIFAR-100 *[16]* is a well-known image classification dataset, containing 32×32 images of 100 categories. Training and validate sets are composed of 50k and 10k images.

ImageNet *[29]* is a large-scale classification dataset that

distillation manner	teacher	ResNet56 72.34	ResNet110 74.31	ResNet32×4 79.42	WRN-40-2 75.61	WRN-40-2 75.61	VGG13 74.64
	student	ResNet20 69.06	ResNet32 71.14	ResNet8×4 72.50	WRN-16-2 73.26	WRN-40-1 71.98	VGG8 70.36
features	FitNet [28]	69.21	71.06	73.50	73.58	72.24	71.02
	RKD [23]	69.61	71.82	71.90	73.35	72.22	71.48
	CRD [33]	71.16	73.48	75.51	75.48	74.14	73.94
	OFD [10]	70.98	73.23	74.95	75.24	74.33	73.95
	ReviewKD [1]	71.89	73.89	75.63	76.12	**75.09**	**74.84**
logits	KD [12]	70.66	73.08	73.33	74.92	73.54	72.98
	DKD	**71.97**	**74.11**	**76.32**	**76.24**	74.81	74.68
	Δ	+1.31	+1.03	+2.99	+1.32	+1.27	+1.70

Table 6. **Results on the CIFAR-100 validation.** Teachers and students are in the **same** architectures. And Δ represents the performance improvement over the classical KD. All results are the average over 5 trials.

consists of 1000 classes. The training set contains 1.28 million images and the validation set contains 50k images.

***MS-COCO** [20]* is an 80-category general object detection dataset. The `train2017` split contains 118k images, and the `val2017` split contains 5k images.

All implementation details are attached in supplement due to the page limit.

4.1. Main Results

Firstly, we demonstrate the improvements contributed by decoupling (1) NCKD and $p_t^{\mathcal{T}}$ and (2) NCKD and TCKD, respectively. Then, we benchmark our method on image classification and object detection tasks.

Ablation: α and β. The two tables below report the student accuracy (%) with different α and β. ResNet32×4 and ResNet8×4 are set as the teacher and the student, respectively. Firstly, we prove that decoupling $(1 - p_t^{\mathcal{T}})$ and NCKD can bring reasonable performance gain (73.63% *vs*. 74.79%) in the first table. Then, we demonstrate that decoupling weights of NCKD and TCKD could contribute to further improvements (74.79% *vs*. 76.32%). Moreover, the second table indicates that TCKD is indispensable, and the improvements from TCKD are stable with different α around 1.0[5].

β	$1 - p_t^{\mathcal{T}}$	1.0	2.0	4.0	8.0	10.0
top-1	73.63	74.79	75.44	75.94	**76.32**	76.18
α	0.0	0.2	0.5	1.0	2.0	4.0
top-1	75.30	75.64	76.12	**76.32**	76.11	75.42

CIFAR-100 image classification. We discuss experimental results on CIFAR-100 to examine our DKD. The validation accuracy is reported in Table 6 and Table 7. Table 6 contains the results where teachers and students are of the same network architectures. Table 7 shows the results where teachers and students are from different series.

[5]We fix α as 1.0 for simplification in the first table, and β as 8.0 in the second table since it achieves the best performance in the first one.

Notably, DKD achieves consistent improvements on all teacher-student pairs, compared with the baseline and the classical KD. Our method achieves $1 \sim 2\%$ and $2 \sim 3\%$ improvements on teacher-student pairs of the same and different series, respectively. It strongly supports the effectiveness of DKD. Furthermore, DKD achieves comparable or even better performances than feature-based distillation methods, significantly improving the trade-off between distillation performance and training efficiency, which will be further discussed in Sec 4.2.

ImageNet image classification. Top-1 and top-5 accuracies of image classification on ImageNet are reported in Table 8 and Table 9. Our DKD achieves a significant improvement. It's worth mentioning that the performance of DKD is better than the most state-of-the-art results of feature distillation methods.

MS-COCO object detection. As discussed in previous works, the performance of the object detection task greatly depends on the quality of deep features to locate interested objects. This rule also stands in transferring knowledge between detectors [17, 37], *i.e*., feature mimicking is of vital importance since logits are not capable of providing knowledge for object localization. As shown in Table 10, singly applying DKD can hardly achieve outstanding performances, but expectedly surpasses the classical KD. Thus, we introduce the feature-based distillation method ReviewKD [1] to obtain satisfactory results. It can be observed that our DKD can further boost AP metrics, even the distillation performance of ReviewKD is relatively high. Conclusively, new state-of-the-art results are obtained by combining our DKD with feature-based distillation methods on the object detection task.

4.2. Extensions

For a better understanding of DKD, we provide extensions from four perspectives. First of all, we comprehensively compare the training efficiency of DKD with representative state-of-the-art methods. Then, we provide a new

distillation manner						
	teacher	ResNet32×4 79.42	WRN-40-2 75.61	VGG13 74.64	ResNet50 79.34	ResNet32×4 79.42
	student	ShuffleNet-V1 70.50	ShuffleNet-V1 70.50	MobileNet-V2 64.60	MobileNet-V2 64.60	ShuffleNet-V2 71.82
features	FitNet [28]	73.59	73.73	64.14	63.16	73.54
	RKD [23]	72.28	72.21	64.52	64.43	73.21
	CRD [33]	75.11	76.05	69.73	69.11	75.65
	OFD [10]	75.98	75.85	69.48	69.04	76.82
	ReviewKD [1]	**77.45**	**77.14**	**70.37**	69.89	**77.78**
logits	KD [12]	74.07	74.83	67.37	67.35	74.45
	DKD	76.45	76.70	69.71	**70.35**	77.07
	Δ	+2.38	+1.87	+2.34	+3.00	+2.62

Table 7. **Results on the CIFAR-100 validation.** Teachers and students are in **different** architectures. And Δ represents the performance improvement over the classical KD. All results are the average over 5 trials.

distillation manner			features				logits		
	teacher	student	AT [43]	OFD [10]	CRD [33]	ReviewKD [1]	KD [12]	KD*	**DKD**
top-1	73.31	69.75	70.69	70.81	71.17	71.61	70.66	71.03	**71.70**
top-5	91.42	89.07	90.01	89.98	90.13	**90.51**	89.88	90.05	90.41

Table 8. **Top-1 and top-5 accuracy (%) on the ImageNet validation.** We set **ResNet-34** as the teacher and **ResNet-18** as the student. KD* represents the result of our implementation. All results are the average over 3 trials.

distillation manner			features				logits		
	teacher	student	AT [43]	OFD [10]	CRD [33]	ReviewKD [1]	KD [12]	KD*	**DKD**
top-1	76.16	68.87	69.56	71.25	71.37	**72.56**	68.58	70.50	72.05
top-5	92.86	88.76	89.33	90.34	90.41	91.00	88.98	89.80	**91.05**

Table 9. **Top-1 and top-5 accuracy (%) on the ImageNet validation.** We set **ResNet-50** as the teacher and **MobileNet-V1** as the student. KD* represents the result of our implementation. All results are the average over 3 trials.

perspective to explain why *bigger models are not always better teachers* and alleviate this problem by utilizing DKD. Moreover, following [33], we examine the transferability of deep features learned by DKD. And we also present some visualizations to validate the superiority of DKD.

Training efficiency. We assess the training costs of state-of-the-art distillation methods, proving the high training efficiency of DKD. As shown in Figure 2, our DKD achieves the best trade-off between model performances and training costs (*e.g.*, training time and extra parameters). Since DKD is reformulated from the classical KD, it needs almost the same computational complexity as KD, and of course no extra parameters. However, feature-based distillation methods require extra training time for distillation intermediate layer features, as well as the GPU memory costs.

Improving performances of big teachers. We provide a new potential explanation on the *bigger models are not always better teachers* issue. Specifically, bigger teachers are expected but cannot transfer more beneficial knowledge, even achieving worse performances than smaller ones.

Previous works [3, 36] explained this phenomenon with the large capacity gap between big teachers and small students. However, we suppose that the main reason is the suppression of NCKD, *i.e.*, the $(1-p_t^{\mathcal{T}})$ would become smaller with the teacher getting bigger. What's more, related works on this problem also could be explained from this perspective, *e.g.*, ESKD [3] employs early-stopped teacher models to alleviate this problem, and these teachers would be under-convergence and yield smaller $p_t^{\mathcal{T}}$.

To validate our conjecture, we perform our DKD on a series of teacher models. Experimental results in Table 11 and Table 12 consistently indicate that our DKD alleviates the *bigger models are not always better teachers* problem.

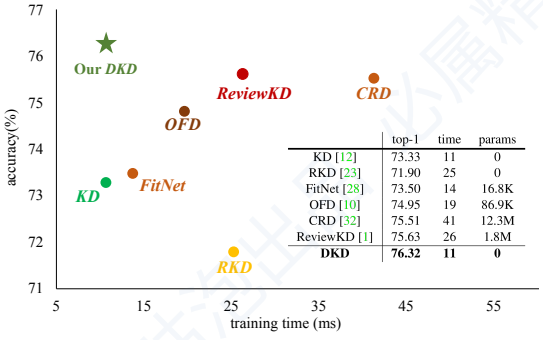


Figure 2. Training time (per batch) *vs.* accuracy on CIFAR-100. We set ResNet32×4 as the teacher and ResNet8×4 as the student. The table shows the number of extra parameters for each method.

Feature transferability. We perform experiments to evaluate the transferability of deep features to verify that our DKD transfers more generalizable knowledge. Follow-

	R-101 & R-18			R-101 & R-50			R-50 & MV2		
	AP	AP_{50}	AP_{75}	AP	AP_{50}	AP_{75}	AP	AP_{50}	AP_{75}
teacher	42.04	62.48	45.88	42.04	62.48	45.88	40.22	61.02	43.81
student	33.26	53.61	35.26	37.93	58.84	41.05	29.47	48.87	30.90
KD [12]	33.97	54.66	36.62	38.35	59.41	41.71	30.13	50.28	31.35
FitNet [28]	34.13	54.16	36.71	38.76	59.62	41.80	30.20	49.80	31.69
FGFI [38]	35.44	55.51	38.17	39.44	60.27	43.04	31.16	50.68	32.92
ReviewKD [1]	36.75	56.72	34.00	40.36	60.97	44.08	33.71	53.15	36.13
DKD	35.05	56.60	37.54	39.25	60.90	42.73	32.34	53.77	34.01
DKD+ReviewKD	**37.01**	**57.53**	**39.85**	**40.65**	**61.51**	**44.44**	**34.35**	**54.89**	**36.61**

Table 10. **Results on MS-COCO based on Faster-RCNN [27]-FPN [19]**: AP evaluated on `val2017`. Teacher-student pairs are ResNet-101 (R-101) & ResNet-18 (R-18), ResNet-101 & ResNet-50 (R-50) and ResNet-50 & MobileNet-V2 (MV2) respectively. All results are the average over 3 trials. More details are attached in supplement.

teacher	W-28-2	W-40-2	W-16-4	W-28-4
	75.45	75.61	77.51	78.60
KD	75.37	74.92	75.79	75.04
DKD	75.92	76.24	76.00	76.45

Table 11. Results on CIFAR-100. We set WRN-16-2 as the student and WRN series networks as teachers.

teacher	VGG13	WRN-16-4	ResNet50
	74.64	77.51	79.34
KD	74.93	75.79	75.36
DKD	75.45	76.00	76.60

Table 12. Results on CIFAR-100. We set WRN-16-2 as the student and networks from different series as teachers.

ing [33], we use the WRN-16-2 distilled from WRN-40-2 as the feature extractor. Then linear probing tasks are performed on downstream datasets, *i.e.* STL-10 [4] and Tiny-ImageNet[6]. Results are reported in Table 13, proving the outstanding transferability of features learned with our DKD. Implementation details are in the supplement.

	baseline	KD	FitNet	CRD	ReviewKD	**DKD**
STL-10	69.7	70.9	70.3	71.6	72.4	**72.9**
TI	33.7	33.9	33.5	35.6	36.6	**37.1**

Table 13. **Comparison with previous methods on transferring features** from CIFAR-100 to STL-10 and Tiny-ImageNet (TI).

Visualizations. We present visualizations from two perspectives (with setting teacher as ResNet32x4 and student as ResNet8x4 on CIFAR-100). (1) The t-SNE (Fig. 3) results show that representations of DKD are more separable than KD, proving that DKD benefits the discriminability of deep features. (2) We also visualize the difference of correlation matrices of student and teacher logits (Fig. 4). Compared with KD, DKD helps the student to output more similar logits with the teacher, *i.e.*, achieving better distillation performances.

5. Discussion and Conclusion

This paper provides a novel viewpoint to interpret logit distillation by reformulating the classical KD loss into two parts, *i.e.*, target class knowledge distillation (TCKD) and non-target class knowledge distillation (NCKD). The effects of both parts are respectively investigated and proved. More importantly, we reveal that the coupled formulation of KD limits the effectiveness and flexibility of knowledge transfer. To overcome these issues, we propose Decoupled Knowledge Distillation (DKD), which achieves significant improvements on CIFAR-100, ImageNet and MS-COCO datasets for image classification and object detection tasks. Besides, the superiority of DKD in training efficiency and feature transferability is also demonstrated. We hope this paper will contribute to future logit distillation research.

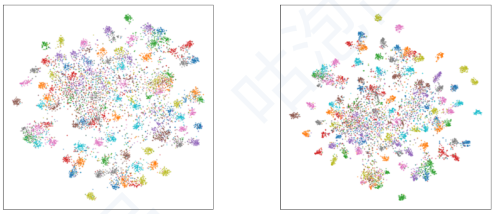


Figure 3. t-SNE of features learned by KD (left) and DKD (right).

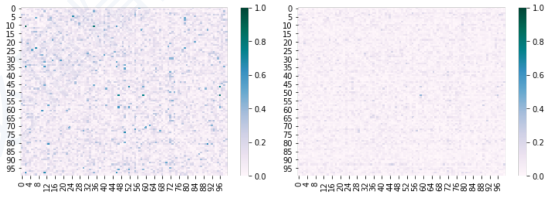


Figure 4. Difference of correlation matrices of student and teacher logits. Obviously, DKD (right) leads to a smaller difference (more similar prediction) than KD (left).

limitations and future works. Noticeable limitations are discussed as follows. DKD could not outperform state-of-the-art feature-based methods (*e.g.*, ReviewKD [1]) on object detection tasks because logits-based methods cannot transfer knowledge about localization. Besides, we have provided an intuitive guidance on how to adjust β in our supplement. However, the strict correlation between the distillation performance and β is not fully investigated, which will be our future research direction.

[6] https://www.kaggle.com/c/tiny-imagenet

References

[1] Pengguang Chen, Shu Liu, Hengshuang Zhao, and Jiaya Jia. Distilling knowledge via knowledge review. In *CVPR*, 2021. 6, 7, 8, 11, 13

[2] Xu Cheng, Zhefan Rao, Yilan Chen, and Quanshi Zhang. Explaining knowledge distillation by quantifying the knowledge. In *CVPR*, 2020. 2

[3] Jang Hyun Cho and Bharath Hariharan. On the efficacy of knowledge distillation. In *ICCV*, 2019. 2, 7

[4] Adam Coates, Andrew Ng, and Honglak Lee. An analysis of single-layer networks in unsupervised feature learning. In *AISTATS*, 2011. 8

[5] Ekin D Cubuk, Barret Zoph, Dandelion Mane, Vijay Vasudevan, and Quoc V Le. Autoaugment: Learning augmentation strategies from data. In *CVPR*, 2019. 4, 12

[6] Tommaso Furlanello, Zachary Lipton, Michael Tschannen, Laurent Itti, and Anima Anandkumar. Born again neural networks. In *ICML*, 2018. 2

[7] Bo Han, Quanming Yao, Xingrui Yu, Gang Niu, Miao Xu, Weihua Hu, Ivor Tsang, and Masashi Sugiyama. Co-teaching: Robust training of deep neural networks with extremely noisy labels. *arXiv:1804.06872*, 2018. 4, 12

[8] Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross Girshick. Mask R-CNN. In *ICCV*, 2017. 1

[9] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *CVPR*, 2016. 1, 3, 11

[10] Byeongho Heo, Jeesoo Kim, Sangdoo Yun, Hyojin Park, Nojun Kwak, and Jin Young Choi. A comprehensive overhaul of feature distillation. In *ICCV*, 2019. 2, 6, 7, 13

[11] Byeongho Heo, Minsik Lee, Sangdoo Yun, and Jin Young Choi. Knowledge transfer via distillation of activation boundaries formed by hidden neurons. In *AAAI*, 2019. 2

[12] Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. Distilling the knowledge in a neural network. In *arXiv:1503.02531*, 2015. 1, 2, 3, 6, 7, 8, 12

[13] Jie Hu, Li Shen, and Gang Sun. Squeeze-and-excitation networks. In *CVPR*, 2018. 1

[14] Zehao Huang and Naiyan Wang. Like what you like: Knowledge distill via neuron selectivity transfer. *arXiv:1707.01219*, 2017. 2

[15] Jangho Kim, SeongUk Park, and Nojun Kwak. Paraphrasing complex network: Network compression via factor transfer. In *NeurIPS*, 2018. 2

[16] Alex Krizhevsky, Geoffrey Hinton, et al. Learning multiple layers of features from tiny images. 2009. 3, 5, 12

[17] Quanquan Li, Shengying Jin, and Junjie Yan. Mimicking very efficient network for object detection. In *CVPR*, 2017. 2, 6

[18] Zheng Li, Jingwen Ye, Mingli Song, Ying Huang, and Zhigeng Pan. Online knowledge distillation for efficient pose estimation. In *ICCV*, 2021. 2

[19] Tsung-Yi Lin, Piotr Dollár, Ross Girshick, Kaiming He, Bharath Hariharan, and Serge Belongie. Feature pyramid networks for object detection. In *CVPR*, 2017. 8, 11

[20] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. Microsoft coco: Common objects in context. In *ECCV*, 2014. 6

[21] Ningning Ma, Xiangyu Zhang, Hai-Tao Zheng, and Jian Sun. Shufflenet V2: Practical guidelines for efficient cnn architecture design. In *ECCV*, 2018. 1, 3, 11

[22] Seyed Iman Mirzadeh, Mehrdad Farajtabar, Ang Li, Nir Levine, Akihiro Matsukawa, and Hassan Ghasemzadeh. Improved knowledge distillation via teacher assistant. In *AAAI*, 2020. 2

[23] Wonpyo Park, Dongju Kim, Yan Lu, and Minsu Cho. Relational knowledge distillation. In *CVPR*, 2019. 2, 3, 6, 7

[24] 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, Zach 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, 2019. 5

[25] Baoyun Peng, Xiao Jin, Jiaheng Liu, Dongsheng Li, Yichao Wu, Yu Liu, Shunfeng Zhou, and Zhaoning Zhang. Correlation congruence for knowledge distillation. In *CVPR*, 2019. 2

[26] Mary Phuong and Christoph Lampert. Towards understanding knowledge distillation. In *ICML*, 2019. 2

[27] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. In *NeurIPS*, 2015. 1, 8, 11

[28] Adriana Romero, Nicolas Ballas, Samira Ebrahimi Kahou, Antoine Chassang, Carlo Gatta, and Yoshua Bengio. Fitnets: Hints for thin deep nets. *ICLR*, 2015. 1, 2, 6, 7, 8

[29] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, et al. ImageNet large scale visual recognition challenge. *IJCV*, 2015. 4, 5

[30] Mark Sandler, Andrew Howard, Menglong Zhu, Andrey Zhmoginov, and Liang-Chieh Chen. MobilenetV2: Inverted residuals and linear bottlenecks. In *CVPR*, 2018. 11

[31] Evan Shelhamer, Jonathan Long, and Trevor Darrell. Fully convolutional networks for semantic segmentation. *T-PAMI*, 2016. 1

[32] K. Simonyan and A Zisserman. Very deep convolutional networks for large-scale image recognition. In *ICLR*, May 2015. 11

[33] Yonglong Tian, Dilip Krishnan, and Phillip Isola. Contrastive representation distillation. In *ICLR*, 2020. 2, 3, 6, 7, 8, 11, 12, 13

[34] Frederick Tung and Greg Mori. Similarity-preserving knowledge distillation. In *ICCV*, 2019. 2, 3

[35] Brendan Van Rooyen, Aditya Krishna Menon, and Robert C Williamson. Learning with symmetric label noise: The importance of being unhinged. *arXiv:1505.07634*, 2015. 4, 12

[36] Lin Wang and Kuk-Jin Yoon. Knowledge distillation and student-teacher learning for visual intelligence: A review and new outlooks. *T-PAMI*, 2021. 7

[37] Tao Wang, Li Yuan, Xiaopeng Zhang, and Jiashi Feng. Distilling object detectors with fine-grained feature imitation. In *CVPR*, 2019. 6

[38] Tao Wang, Li Yuan, Xiaopeng Zhang, and Jiashi Feng. Distilling object detectors with fine-grained feature imitation. In *CVPR*, 2019. 8

[39] Yuxin Wu, Alexander Kirillov, Francisco Massa, Wan-Yen Lo, and Ross Girshick. Detectron2. https://github.com/facebookresearch/detectron2, 2019. 11

[40] Chenglin Yang, Lingxi Xie, Chi Su, and Alan L Yuille. Snapshot distillation: Teacher-student optimization in one generation. In *CVPR*, 2019. 2

[41] Junho Yim, Donggyu Joo, Jihoon Bae, and Junmo Kim. A gift from knowledge distillation: Fast optimization, network minimization and transfer learning. In *CVPR*, 2017. 2

[42] Sergey Zagoruyko and Nikos Komodakis. Wide residual networks. In *BMVC*, 2016. 3, 11

[43] Sergey Zagoruyko and Nikos Komodakis. Paying more attention to attention: Improving the performance of convolutional neural networks via attention transfer. *ICLR*, 2017. 2, 7

[44] Ying Zhang, Tao Xiang, Timothy M Hospedales, and Huchuan Lu. Deep mutual learning. In *CVPR*, 2018. 2

[45] Hengshuang Zhao, Jianping Shi, Xiaojuan Qi, Xiaogang Wang, and Jiaya Jia. Pyramid scene parsing network. In *CVPR*, 2017. 1

A. Appendix

A.1. Details about the reformulation in Sec 3.1

Details of the mathematical derivation in Sec 3.1 of the manuscript are as follows (notations are the same in Sec 3.1 of the manuscript):

$$\begin{aligned}\text{KD} &= \text{KL}(\mathbf{p}^{\mathcal{T}}||\mathbf{p}^{\mathcal{S}}) \\ &= \sum_{i=1}^{C} p_i^{\mathcal{T}} \log(\frac{p_i^{\mathcal{T}}}{p_i^{\mathcal{S}}}) \\ &= p_t^{\mathcal{T}} \log(\frac{p_t^{\mathcal{T}}}{p_t^{\mathcal{S}}}) + \sum_{i=1,i\neq t}^{C} p_i^{\mathcal{T}} \log(\frac{p_i^{\mathcal{T}}}{p_i^{\mathcal{S}}}).\end{aligned} \tag{8}$$

According to Eqn.(1) and Eqn.(2) of the manuscript, we have $\hat{p}_i = p_i/p_{\setminus t}$. Thus, we can rewrite Eqn.(8) to:

$$\begin{aligned}\text{KD} &= p_t^{\mathcal{T}} \log(\frac{p_t^{\mathcal{T}}}{p_t^{\mathcal{S}}}) + \sum_{i=1,i\neq t}^{C} p_{\setminus t}^{\mathcal{T}}\hat{p}_i^{\mathcal{T}} \log(\frac{p_{\setminus t}^{\mathcal{T}}\hat{p}_i^{\mathcal{T}}}{p_{\setminus t}^{\mathcal{S}}\hat{p}_i^{\mathcal{S}}}) \\ &= p_t^{\mathcal{T}} \log(\frac{p_t^{\mathcal{T}}}{p_t^{\mathcal{S}}}) + \sum_{i=1,i\neq t}^{C} p_{\setminus t}^{\mathcal{T}}\hat{p}_i^{\mathcal{T}} (\log(\frac{\hat{p}_i^{\mathcal{T}}}{\hat{p}_i^{\mathcal{S}}}) + \log(\frac{p_{\setminus t}^{\mathcal{T}}}{p_{\setminus t}^{\mathcal{S}}})) \\ &= p_t^{\mathcal{T}} \log(\frac{p_t^{\mathcal{T}}}{p_t^{\mathcal{S}}}) + \sum_{i=1,i\neq t}^{C} p_{\setminus t}^{\mathcal{T}}\hat{p}_i^{\mathcal{T}} \log(\frac{\hat{p}_i^{\mathcal{T}}}{\hat{p}_i^{\mathcal{S}}}) \\ &+ \sum_{i=1,i\neq t}^{C} p_{\setminus t}^{\mathcal{T}}\hat{p}_i^{\mathcal{T}} \log(\frac{p_{\setminus t}^{\mathcal{T}}}{p_{\setminus t}^{\mathcal{S}}}).\end{aligned} \tag{9}$$

Since $p_{\setminus t}^{\mathcal{T}}$ and $p_{\setminus t}^{\mathcal{S}}$ are irrelevant to the class index i, we have:

$$\begin{aligned}\sum_{i=1,i\neq t}^{C} p_{\setminus t}^{\mathcal{T}}\hat{p}_i^{\mathcal{T}} \log(\frac{p_{\setminus t}^{\mathcal{T}}}{p_{\setminus t}^{\mathcal{S}}}) &= p_{\setminus t}^{\mathcal{T}} \log(\frac{p_{\setminus t}^{\mathcal{T}}}{p_{\setminus t}^{\mathcal{S}}}) \sum_{i=1,i\neq t}^{C} \hat{p}_i^{\mathcal{T}} \\ &= p_{\setminus t}^{\mathcal{T}} \log(\frac{p_{\setminus t}^{\mathcal{T}}}{p_{\setminus t}^{\mathcal{S}}}).\end{aligned} \tag{10}$$

Then,

$$\text{KD} = \underbrace{p_t^{\mathcal{T}} \log(\frac{p_t^{\mathcal{T}}}{p_t^{\mathcal{S}}}) + p_{\setminus t}^{\mathcal{T}} \log(\frac{p_{\setminus t}^{\mathcal{T}}}{p_{\setminus t}^{\mathcal{S}}})}_{\text{KL}(\mathbf{b}^{\mathcal{T}}||\mathbf{b}^{\mathcal{S}})} + \underbrace{p_{\setminus t}^{\mathcal{T}} \sum_{i=1,i\neq t}^{C} \hat{p}_i^{\mathcal{T}} \log(\frac{\hat{p}_i^{\mathcal{T}}}{\hat{p}_i^{\mathcal{S}}})}_{\text{KL}(\hat{\mathbf{p}}^{\mathcal{T}}||\hat{\mathbf{p}}^{\mathcal{S}})}. \tag{11}$$

According to the definition of KL-Divergence, Eqn.(11) can be rewritten as (which is the same as Eqn.(5) of the manuscript):

$$\text{KD} = \text{KL}(\mathbf{b}^{\mathcal{T}}||\mathbf{b}^{\mathcal{S}}) + (1 - p_t^{\mathcal{T}})\text{KL}(\hat{\mathbf{p}}^{\mathcal{T}}||\hat{\mathbf{p}}^{\mathcal{S}}) \tag{12}$$

A.2. Implementation: Experiments in Sec 4

CIFAR-100: Our implementation for CIFAR-100 follows the practice in [33]. Teachers and students are trained for 240 epochs with SGD. As the batch size is 64, the learning rates are 0.01 for ShuffleNet [21] and MobileNet-V2 [30], 0.05 for the other series (*e.g.* VGG [32], ResNet [9] and WRN [42]). The learning rate is divided by 10 at 150, 180 and 210 epochs. The weight decay and the momentum are set to 5e-4 and 0.9. The weight for the cross-entropy loss is set to 1.0. The temperature is set as 4 and α is set as 1.0 for all experiments. The proper value of β could be different for different teachers, and the details and discussions are in the next section. And we utilize a 20-epoch linear warmup for all experiments since the value of β could be high leading to a large initial loss.

ImageNet: Our implementation for ImageNet follows the standard practice. We train the models for 100 epochs. As the batch size is 512, the learning rate is initialized to 0.2 and divided by 10 for every 30 epochs. Weight decay is 1e-4 and the weight for the cross-entropy loss is set to 1.0. We set temperature as 1 and α as 0.5 for all experiments. Strictly following [1, 33], for distilling networks of the same architecture, the teacher is ResNet-34 model, the student is ResNet-18, and β is set to 0.5. For different series, the teacher is ResNet-50 model, the student is MobileNet-V1, and β is set to 2.0.

MS-COCO: Our implementation for MS-COCO follows the settings in [1]. We use the two-stage method Faster R-CNN [27] with FPN [19] as the feature extractors. ResNet [9] models and MobileNet-V2 [30] are selected as teachers and students. All students are trained with the 1x scheduler (schedulers and task-specific loss weights follow Detectron2 [39]). We employ the DKD loss on the R-CNN head, and set α as 1.0, β as 0.25, and temperature as 1 for all experiments.

Results of compared methods are reported in their original papers or reproduced by previous works [1, 33].

A.3. Guidance for tuning β

We suppose that the importance of NCKD in knowledge transfer could be related to the confidence of the teacher. Intuitively, the more confident the teacher is, the more valuable the NCKD could be, and the larger β should be applied. However, NCKD could increase the gradient contributed by logits of non-target classes. Thus, an improper large β could harm the correctness of the student's prediction. If the logit value of the target class is much higher than all non-target classes, the teacher could be regarded as more confident and a large $beta$ could be more reasonable. Thus, we suppose that the value of β could be related to the logit value *gap* between the target and all non-target classes. Specifically, the *gap* between the logit of the target class (*i.e.*, z_t, where z represents the output logit and t represents the target class) and the max logit among non-target classes could be reliable guidance for tuning β, which can be denoted as $z_t - z_{max}$, where $z_{max} = \max(\{z_i|i \neq t\})$.

β	ResNet56	WRN-40-2	ResNet32x4
1.0	76.02	75.94	74.95
2.0	**76.32**	76.25	75.64
4.0	75.91	76.17	75.82
6.0	75.62	**76.70**	76.34
8.0	75.33	76.44	**76.45**
10.0	75.35	76.21	76.32
$z_t - z_{max}$	5.36	7.24	8.40

Table A.1. Accuracy(%) on CIFAR-100 [16] with different β and different teachers. The *gap* ($z_t - z_{max}$) is also reported.

We report experimental results on CIFAR-100 [16] to verify this conjecture. We select ResNet56, WRN-40-2 and ResNet32×4 as teachers and ShuffleNet-V1 as the student, and apply different β. Both top-1 accuracy (%) and the *gap* $z_t - z_{max}$ (averaged over all training samples) are reported. As shown in Table A.1, the best value of β could be positively proportional to the *gap*, which we suppose could be guidance of tuning β and a direction for further research. Based on this, the value of β for each teacher in Table 6 and Table 7 of the manuscript is set as follows (in Table A.2):

teacher	$z_t - z_{max}$	β
ResNet56	5.36	2.0
ResNet110	6.73	2.0
WRN-40-2	7.24	6.0
VGG13	8.25	6.0
ResNet50	8.53	8.0
ResNet32×4	8.40	8.0

Table A.2. The value of β for different teachers in Table 6 and Table 7 of the manuscript.

A.4. Implementation: Experiments in Sec 3.2

In this part, we report the implementation details of the experiments in Sec 3.2 of the manuscript.

Basic settings. We set the loss term of KD and CE as 1.0 and 1.0, respectively (instead of the default $0.1CE + 0.9KD$ setting in [33]). The setting in [33] follows the loss form proposed by [12], which assumes that the sum of all terms' weights should be 1.0. However, the NCKD loss is target-irrelevant, which means the target-relevant loss could be 0.1 if we utilize the original setting when only applying NCKD. Based on this, we set the loss weight of all terms (*e.g.*, KD, TCKD, NCKD and CE) as 1.0 for all experiments in Sec 3.2 of the manuscript.

Strong augmentation. We employ the AutoAugment [5] to reveal the effectiveness of TCKD in Sec 3.2 of the manuscript. Specifically, we add the CIFAR AutoAugment policy[7] after applying the default augmentation (random crop and horizontal flip). Then we train the teacher and the student with the same augmentation policy.

[7] https://github.com/DeepVoltaire/AutoAugment

Noisy labels. We also perform experiments on noisy training data to verify that TCKD conveys the knowledge about sample "difficulty". Specifically, we follow the settings of [7, 35], utilizing the symmetric noise type[8]. We train a teacher network on the noisy training data and select the best epoch to distill the student (on the same training data).

A.5. Explanation about why TCKD brings performance drop in Table 1

In Table 1 of the manuscript, we reveal that singly applying TCKD could bring performance drop sometimes. An explanation for this phenomenon is that the high temperature (T=4) will lead to a great gradient to increase the non-target classes' logits, which will harm the correctness of the student's prediction. Without NCKD, the information about the class similarity (or the prominent dark knowledge) is not available, so that TCKD's gradient could do no good but lead to performance drop (since TCKD could bring marginal performance gain on easy-fitting training data). To verify that the large temperature is not proper when singly applying TCKD, we perform experiments with different temperatures (T) in the table below. Results in Table A.3 show that the performance is almost the same as the vanilla training baseline (73.26 in Table 1 of the manuscript) when the temperature is set as 1. And the performance drop is positively related to the temperature.

T	1	2	3	4
top-1	73.24	73.05	71.69	70.96

Table A.3. Accuracy (%) with different temperature(T) when only applying TCKD. The teacher and the student are set as WRN-40-2 and WRN-16-2, respectively.

A.6. How to employ DKD on detectors

In this paper, we employ our DKD on the two-stage object detector Faster R-CNN. We only employ our DKD on the R-CNN head. Specifically, given a student network, we utilize the labels assigned to the proposals (generated by the RPN module) as the target class(if $IoU(proposal) < 0.5$, the target class is set as "background"). Then, we use a teacher network to get the R-CNN prediction logits of the *same proposals* (locations are the same, while the features are from the teacher's backbone). Thus, we can employ our DKD by minimizing the KL-Divergence (*i.e.*, TCKD and NCKD) between the student's logits and the teacher's.

A.7. Implementation: Experiments in Sec 4.2

Training efficiency. We report the training time of each distillation method in Figure 2 of the manuscript. The

[8] https://github.com/bhanML/Co-teaching/blob/master/data/cifar.py

training time (per batch) is the sum of (1) the data processing time (*e.g.*, including the time to sample the contrast examples in [33]), (2) the network forward time and the gradient backward time and (3) the memory updating time (*e.g.*,including the time to update the contrast memory in [33]). We also report the number of extra parameters for each method. Besides the learnable parameters (*e.g.*, the connectors in [10] and the ABF modules in [1]), we also calculate the extra dictionary memory(*e.g.*, the contrast memory in [33]).

Feature transferability. We perform linear probing experiments to verify the feature transferability of our DKD in Sec 4.2 of the manuscript. We use the WRN-16-2 distilled from a WRN-40-2 teacher as the feature extractor (only using the feature generated by the final global average pooling module), then train linear fully-connected (FC) layers as classifier modules for STL-10 and Tiny-ImageNet datasets (the feature extractor is fixed during training). We train the FC via an SGD optimizer with 0.9 momentum and 0.0 weight decay. The number of total epochs is set as 40, and the learning rate is set to 0.1 for a 128 batch size and divided by 10 for every 10 epochs.