Multi-level Logit Distillation

Ying Jin[1] Jiaqi Wang[2]✉* Dahua Lin[1,2]

[1]CUHK-SenseTime Joint Lab, The Chinese University of Hong Kong
[2]Shanghai AI Laboratory

{jy021,dhlin}@ie.cuhk.edu.hk, wjqdev@gmail.com

Abstract

Knowledge Distillation (KD) aims at distilling the knowledge from the large teacher model to a lightweight student model. Mainstream KD methods can be divided into two categories, logit distillation, and feature distillation. The former is easy to implement, but inferior in performance, while the latter is not applicable to some practical circumstances due to concerns such as privacy and safety. Towards this dilemma, in this paper, we explore a stronger logit distillation method via making better utilization of logit outputs. Concretely, we propose a simple yet effective approach to logit distillation via ***multi-level prediction alignment****. Through this framework, the prediction alignment is not only conducted at the instance level, but also at the batch and class level, through which the student model learns instance prediction, input correlation, and category correlation simultaneously. In addition, a prediction augmentation mechanism based on model calibration further boosts the performance. Extensive experiment results validate that our method enjoys consistently higher performance than previous logit distillation methods, and even reaches competitive performance with mainstream feature distillation methods. Code is available at* https://github.com/Jin-Ying/Multi-Level-Logit-Distillation.

1. Introduction

The last few decades have witnessed the prosperity of deep learning in computer vision tasks, such as image classification [3, 7, 14, 26], object detection [21], and segmentation [25, 37]. However, due to their overwhelming large model size, many deep models rely heavily on computation and storage resources, which makes it nearly impossible to deploy them in some practical scenarios, such as mobile devices. Towards this challenge, Knowledge Distillation [10] (KD) was introduced to reduce model capacity. Concretely, the KD framework consists of one teacher model (large) and one student model (small). The main objective of KD is to distill the knowledge in the teacher model to the lightweight student model, which can be readily deployed. Various KD methods [1, 8, 19, 22, 27] have been proposed and proved to be effective.

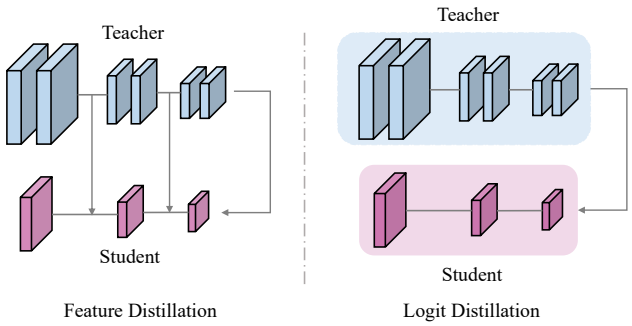


Figure 1. **Problem Setting.** Feature distillation methods utilize features in the intermediate layers as well as logit outputs. On the contrary, logit distillation methods conduct knowledge distillation merely with logit outputs.

Mainstream KD methods fall into two lines of work, 1) logit distillation and 2) feature distillation. Logit distillation conveys knowledge from the teacher model to the student merely on the logit level. The earliest KD method [10], which distills knowledge by reducing the divergence of predictions, is an example of logit distillation method. Towards better utilization of teacher knowledge, recent researches [1, 22] shed light on the intermediate layers in the teacher model, conducting distillation by matching feature distributions as well as logit outputs among the teacher and student model. These methods are coined feature distillation methods. We compare feature distillation with logit distillation in Figure 1.

Utilizing the feature knowledge in intermediate layers, feature distillation methods are more likely to reach superior performance. However, in some real-world applications, the intrinsic architecture of the teacher model is invis-

*✉Corresponding author.

Table 1. **Comparison of different methods.** Compared with previous logit distillation and feature distillation methods, our method conducts prediction alignment at the instance level, batch level and category level simultaneously.

Method	Instance-level Alignment	Batch-level Alignment	Category-level Alignment
Logit Distillation (Previous)	✓	✗	✗
Feature Distillation	✓	✓	✗
Logit Distillation (Ours)	✓	✓	✓

ible due to commercial, privacy, and safety concerns, making these methods invalid under such circumstances. For example, when launching adversarial attacks [5], it is much easier for hackers to recover the train data when all intermediate layers in the model are available. Such a leak of data may cause highly negative influences on financial or medical applications.

Facing such a dilemma, we pay attention to logit distillation, which does not need to have access to the features in the intermediate layers. To mitigate the performance gap between logit distillation and feature distillation, we shall make better utilization of logit outputs. Concretely, in this paper, we propose multi-level logit distillation, a simple yet effective approach to absorb more information from logit outputs. We propose a multi-level alignment to reduce the divergence of predictions between the teacher and student at the instance, batch, and class level. Through this alignment, the student model absorbs knowledge from the teacher model not only in *instance*-level prediction, but in *batch*-level input correlation and *class*-level category correlation as well. We compare our multi-level logit distillation method with previous methods in Table 1. The previous logit distillation merely conducts instance-level alignment, and the feature distillation incorporates batch-level alignment. On the contrary, our method implements alignment at multiple levels with logit outputs alone. In addition, we also introduce a prediction augmentation based on model calibration. It enables the student model to learn from more diverse predictions, pushing the performance of our method to a higher level.

Extensive experiment results on mainstream benchmarks validate that our method surpasses previous logit distillation methods, in both homogenous and heterogeneous network knowledge distillation settings. Meanwhile, our method also reaches competitive performance over previous feature distillation methods, proving that our method excels at utilizing logit outputs.

2. Related Work

Proposed in [10], Knowledge Distillation (KD) defines a new model compression framework. It consists of one large teacher model and one lightweight student model. Its goal is to distill (transfer) the knowledge in the teacher model to the student model. Concretely, it forces the student model to mimic the teacher outputs by minimizing the divergence between the predictions from the teacher and student model. Towards the over-confidence / miscalibration phenomenon [6] in neural networks, temperature re-scaling is applied to alleviate the influence. In our method, we also implement prediction augmentations by incorporating multiple temperatures.

Upon proposal, various methods have been proposed for knowledge distillation. These methods fall into two lines of work: 1) logit distillation methods [2, 4, 18, 30, 35] and 2) feature distillation methods [8, 9, 11, 12, 19, 20, 22, 27, 28, 31, 33].

Logit Distillation Logit distillation methods distill knowledge merely with output logits. For instance, the earliest KD method is a logit distillation method. Other logit distillation methods boost knowledge distillation by introducing a mutual-learning paradigm [35] or additional teacher assistant module [18]. The logit distillation methods appear to be straightforward and are ready to be applied to any scenario. However, their performance is often inferior to feature-level methods.

Feature Distillation To further boost knowledge distillation, another line of work, feature distillation, is proposed to conduct distillation on intermediate features as well as logit outputs. Concretely, some of them [8, 9, 22] mitigate the divergence between features in the teacher and student model, which enforces the student model to imitate the teacher model at the feature level. Other methods [19, 27, 28] also convey teacher knowledge by distilling the input correlation. Feature distillation methods are more likely to gain high performance since they absorb rich knowledge from the teacher model. Different from these feature-level methods that transfer input correlation via intermediate features, our distillation method learns input correlation by logit outputs. Our method also considers class correlation, which previous works rarely pay attention to.

3. Methodology

To smooth the presentation, we start from preliminaries. Then we introduce our multi-level logit distillation.

3.1. Preliminaries

Knowledge Distillation We start from the original Knowledge Distillation (KD) method, which was proposed in [10]. To illustrate the procedure of KD, we consider C-way classification task and denote the logit output of a single input as $z \in \mathbb{R}^C$, then the class probability is

$$p_j = \frac{e^{z_j/T}}{\sum_{c=1}^{C} e^{z_c/T}}, \tag{1}$$

where p_j and z_j is the probability value on the j-th class. We compute the Softmax value and T is the temperature scaling hyper-parameter [6]. In knowledge distillation, T is often larger than 1.0, which alleviates the over-confidence phenomenon in neural network [6]. When $T = 1.0$, the output will shrink to vanilla Softmax output. The objective of KD is to distill the knowledge from the large teacher model to the lightweight student model. With rescaled outputs, the original KD method implements distillation by minimizing the KL divergence between the outputs from the teacher and student model,

$$L_{KD} = KL(p^{tea}||p^{stu}) = \sum_{j=1}^{C} p_j^{tea} log(\frac{p_j^{tea}}{p_j^{stu}}), \tag{2}$$

where L_{KD} is the knowledge distillation loss, p_j^{tea} and p_j^{stu} indicates the probability value on the j-th category of the teacher and student output, respectively.

The original KD method, minimizing the divergence on logit outputs, serves as the most fundamental baseline in KD research. As a logit distillation method, its performance is inferior to feature distillation methods. In this paper, we strive to seek a stronger logit distillation method.

3.2. Multi-level Logit Distillation

In this section, we will introduce our multi-level logit distillation. Here, we consider the output of a batch of data instead of a single data. We denote the logit output as $z \in \mathbb{R}^{B\times C}$, where B is the batch size and C means C-way classification. Our method has two core components: 1) prediction augmentation and 2) multi-level alignment.

3.2.1 Prediction Augmentation

To gain richer knowledge from predictions, we propose a prediction augmentation mechanism, through which we can expand a single output to multiple ones. Concretely, we conduct prediction augmentation through model calibration. We take temperature scaling, a widely adopted calibration method [6],

$$p_{i,j,k} = \frac{e^{z_{i,j}/T_k}}{\sum_{c=1}^{C} e^{z_{i,c}/T_k}}, \tag{3}$$

where $p_{i,j,k}$ is the probability value of the i-th input on the j-th category, with temperature hyper-parameter T_k. In our mechanism, $T_0, T_1, ..., T_K$ forms a pool with K temperatures, which enables us to augment one prediction to K diverse outputs.

As shown in Figure 2, take $K = 2$ as an instance, outputs from the teacher and student model are augmented respectively. Through the prediction augmentation mechanism, we convert one prediction to K outputs that are diverse in probability sharpness.

3.2.2 Multi-level Alignment

With augmented predictions, as shown in Figure 2, we propose to align the teacher output and the corresponding student output (according to the temperature) one by one. Instead of the original logit alignment through KL divergence, we propose a novel multi-level alignment, which includes 1) instance-level, 2) batch-level, and 3) class-level alignment.

Instance-level Alignment We inherit the original mechanism in KD to implement instance-level alignment in our method. Concretely, as shown below, we minimize the KL divergence between augmented predictions from the teacher and student model one by one,

$$\begin{aligned} L_{ins} &= \sum_{i=1}^{N}\sum_{k=1}^{K} KL(p_{i,k}^{tea}||p_{i,k}^{stu}) \\ &= \sum_{i=1}^{N}\sum_{k=1}^{K}\sum_{j=1}^{C} p_{i,j,k}^{tea} log(\frac{p_{i,j,k}^{tea}}{p_{i,j,k}^{stu}}), \end{aligned} \tag{4}$$

where L_{ins} means the instance-level alignment loss, $p_{i,j,k}^{tea}$ and $p_{i,j,k}^{stu}$ indicate the teacher and student outputs on the i-th instance, j-th category, that are augmented by T_k. The instance-level alignment forces the student model to mimic the teacher predictions on each instance, which plays the most fundamental role in knowledge distillation. When compared with the vanilla KD [10] method, the core difference of our alignment is that we adopt prediction augmentation by temperature scaling, which transfers more diverse knowledge from the teacher model to the student model.

Batch-level Alignment Instead of aligning predictions at merely instance level, we propose to conduct batch-level alignment by input correlation, the relation between two inputs, which is modeled via features in previous works. In our method, we take logit predictions to quantify it. Specifically, we compute the Gram Matrix on the model predictions as follows,

$$G^k = p_k p_k^T, G_{ab}^k = \sum_{j=1}^{C} p_{a,j,k} \cdot p_{b,j,k}, \tag{5}$$

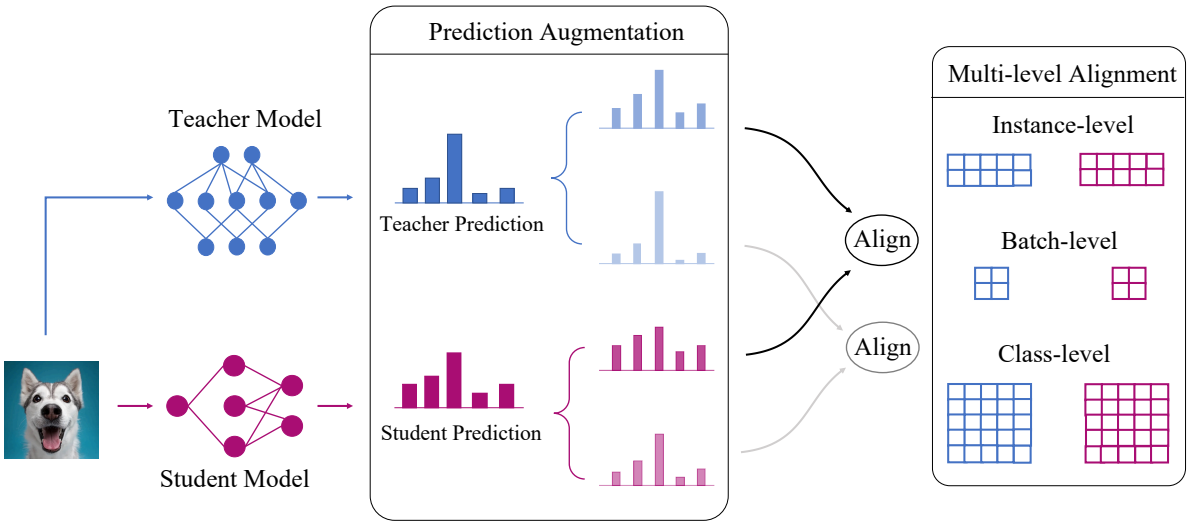


Figure 2. **Method Overview.** In our multi-level logit distillation framework, after obtaining the teacher and student predictions, we conduct **prediction augmentation**, converting them to multiple outputs with different temperatures respectively. The augmented predictions are matched respectively through **multi-level alignment**, which consists of instance-level, batch-level, and class-level alignment. We take batch size $B = 2$ and class number $C = 5$ as an example to demonstrate our multi-level alignment. *(Best viewed in color)*

where G^k is a $B \times B$ matrix, and p_k indicates the predictions obtained via T_k. We can derive that G^k_{ab} models the probability that the a-th and the b-th inputs are classified in the same category, which indicates the relationship between them.

Then we compute the input correlation matrix G_k according to different T_k, with the teacher and student predictions respectively. Our objective is to mitigate the divergence between them, thus the corresponding loss can be

$$L_{batch} = \frac{1}{B}\sum_{k=1}^{K} ||G^k_{tea} - G^k_{stu}||_2^2, \tag{6}$$

where L_{batch} serves as the batch-level alignment loss, G^k_{tea} and G^k_{stu} are the input correlation matrix computed by teacher and student predictions with temperature T_k, respectively. Similarly, with instance-level alignment, we take all augmented predictions and conduct alignment accordingly.

Class-level Alignment The last part of our method lies in class-level alignment. We state that the model predictions can depict the relationship between categories, *i.e.*, if one class is very similar to the ground-truth class, the model is prone to be reluctant between them, forming two high peaks in predictions. Such a category correlation can be modeled by predictions of a batch of data as follows,

$$M^k = p_k^T p_k, M^k_{ab} = \sum_{i=1}^{N} p_{i,a,k} \cdot p_{i,b,k}, \tag{7}$$

where M^k is a $C \times C$ matrix, p_k indicates the predictions obtained via T_k, and M^k_{ab} presents the probability that the inputs in this batch are classified to the a-th category and the b-th category simultaneously, which quantifies the relationship between the two classes.

After quantifying the category correlation, we can enforce the student model to absorb this part of knowledge from the teacher model by the following loss,

$$L_{class} = \frac{1}{C}\sum_{k=1}^{K} ||M^k_{tea} - M^k_{stu}||_2^2 \tag{8}$$

where L_{class} serves as the class-level alignment loss, M^k_{tea} and M^k_{stu} are the category correlation matrix computed by teacher and student predictions with temperature hyper-parameter T_k. Augmented predictions with multiple temperatures alleviate the over-confidence phenomenon in neural networks, which is crucial in modeling the category correlation.

Multi-level Alignemt Now we have designed the mechanism for instance-level, batch-level, and class-level alignment, and our multi-level alignment loss is presented as

$$L_{total} = L_{ins} + L_{batch} + L_{class}. \tag{9}$$

By integrating three parts of loss together, our method enforces the student model to imitate the teacher model not only in instance-level predictions but in batch-level input correlation and class-level category correlation as well. We provide the pseudo-code in Algorithm 1.

Algorithm 1: Pseudo-code in a PyTorch-like style.

```
# z_stu, z_tea: student, teacher logit outputs
# T = [T_1, T_2, ..., T_K]:
#     one set of K different temperatures
# l_ins, l_batch, l_class:
#     three parts of alignment loss
# l_total: total loss
l_total = 0
for t in T do
    p_stu = F.softmax(z_stu / t) # B x C
    p_tea = F.softmax(z_tea / t) # B x C
    l_ins = F.kl_div(p_tea, p_stu)
    G_stu = torch.mm(p_stu, p_stu.t()) # B x B
    G_tea = torch.mm(p_tea, p_tea.t()) # B x B
    l_batch = ((G_stu - G_tea) ** 2).sum() / B
    M_stu = torch.mm(p_stu.t(), p_stu) # C x C
    M_tea = torch.mm(p_tea.t(), p_tea) # C x C
    l_class = ((M_stu - M_tea) ** 2).sum() / C
    l_total += (l_ins + l_batch + l_class)
end
```

4. Experiments

4.1. Datasets and Settings

In our experiments, we evaluate the performance of our method on image classification and objection detection respectively.

Datasets We take three widely researched datasets, 1) CIFAR-100 [13], with 60,000 images in total (50,000 for training and 10,000 for validation) from 100 categories, the resolution of images is 32×32, 2) ImageNet [23], one of the most important benchmark datasets for image classification, with nearly 1.3 million training images and 50,000 images for validation. The images come from 1,000 categories and are in high resolution. 3) MS-COCO [16], a mainstream dataset for object detection, with 118,000 training images and 5,000 validation images from 80 categories.

Settings We focus on knowledge distillation with two different settings in our experiment section. 1) Homogenous architecture where the teacher and student model are in the same type of architecture (*e.g.* ResNet56 and ResNet20), and 2) Heterogeneous architecture where the two models are different in architecture (*e.g.* ResNet32x4 and ShuffleNet-V1). We include various neural network architectures in our experiment, including ResNet [7], WRN [32], VGG [26], ShuffleNet-V1 [34]/V2 [17] and MobileNetV2 [24].

Implementation Details For CIFAR-100, we set the batch size as 64 and the base learning rate as 0.05. For ImageNet, we set the batch size as 128, and the base learning rate as 0.1. For MS-COCO, we set the batch size as 8 and the base learning rate as 0.01. We take 1 GPU to train the model on CIFAR-100 and 8 GPUs on ImageNet and MS-COCO. For prediction augmentation, we take $K = 5$ temperatures $[2.0, 3.0, 4.0, 5.0, 6.0]$ and denote the median of them as T ($T = 4.0$ here). We run each experiment five times and report the average results.

4.2. Experimental Results

In our experiments, we evaluate the performance of our method and compare it with previous logit distillation methods. We also report the performance of other feature distillation methods [1, 8, 19, 22, 27]. We note that such a comparison is somewhat unfair since our method takes merely the output logits to conduct distillation.

CIFAR-100 We evaluate our method on CIFAR-100 and compare it with previous methods. For knowledge distillation where the teacher and student model are in homogenous architecture, as shown in Table 2, our method performs best among logit distillation methods, showing obvious improvements over the original student model and the vanilla KD [10] method. Moreover, our accuracy is slightly better than the feature distillation methods. We note that it validates the strong effectiveness of our method, since it only takes output predictions to surpass all the methods that absorb abundant knowledge from intermediate features.

When it comes to the situation where the teacher and student model are heterogeneous in architecture, the results in Table 3 demonstrate that our method shows a remarkable advantage over the previous logit distillation methods, enhancing the lightweight student model effectively. In addition, our method also shows competitive performance over the feature distillation methods.

ImageNet We plug our method into knowledge distillation on ImageNet, with the teacher and student models in homogenous or heterogeneous architecture. We compare our method with previous logit distillation methods, as well as present the performance of previous feature distillation methods. We report both Top-1 and Top-5 accuracy in Table 4.

The results demonstrate that no matter whether the teacher and student models are homogenous (ResNet34 and ResNet18) or heterogeneous (ResNet50 and MobileNet-V2), our method consistently outperforms previous KD methods. In addition, our method still shows competitive performance over feature distillation methods on such a large-scale and complicated dataset.

MS-COCO We extend our experiments to objection detection, another fundamental computer vision task. We take Faster-RCNN [21]-FPN [15] as the backbone, and AP,

Table 2. **Results on CIFAR-100, Homogenous Architecture.** Top-1 accuracy is adopted as the evaluation metric. The teacher model and student model are in homogenous architecture and their original performance is reported respectively.

Method		ResNet56	ResNet110	ResNet32×4	WRN-40-2	WRN-40-2	VGG13	Avg
	Teacher	72.34	74.31	79.42	75.61	75.61	74.64	
	Student	ResNet20	ResNet32	ResNet8×4	WRN-16-2	WRN-40-1	VGG8	
		69.06	71.14	72.50	73.26	71.98	70.36	
Feature	FitNet [22]	69.21	71.06	73.50	73.58	72.24	71.02	71.77
	RKD [19]	69.61	71.82	71.90	73.35	72.22	71.48	71.73
	CRD [27]	71.16	73.48	75.51	75.48	74.14	73.94	73.95
	OFD [8]	70.98	73.23	74.95	75.24	74.33	73.95	73.78
	ReviewKD [1]	71.89	73.89	75.63	76.12	75.09	74.84	74.58
Logit	KD [10]	70.66	73.08	73.33	74.92	73.54	72.98	73.09
	DML [35]	69.52	72.03	72.12	73.58	72.68	71.79	71.95
	TAKD [18]	70.83	73.37	73.81	75.12	73.78	73.23	73.36
	Ours	**72.19**	**74.11**	**77.08**	**76.63**	**75.35**	**75.18**	**75.09**

Table 3. **Results on CIFAR-100, Heterogeneous Architecture.** Top-1 accuracy is adopted as the evaluation metric. The teacher model and student model are in heterogeneous architecture and their original performance is reported respectively.

Method		ResNet32×4	WRN-40-2	VGG13	ResNet50	ResNet32×4	Avg
	Teacher	79.42	75.61	74.64	79.34	79.42	
	Student	ShuffleNet-V1	ShuffleNet-V1	MobileNet-V2	MobileNet-V2	ShuffleNet-V2	
		70.50	70.50	64.60	64.60	71.82	
Feature	FitNet [22]	73.59	73.73	64.14	63.16	73.54	69.63
	RKD [19]	72.28	72.21	64.52	64.43	73.21	69.33
	CRD [27]	75.11	76.05	69.73	69.11	75.65	73.13
	OFD [8]	75.98	75.85	69.48	69.04	76.82	73.43
	ReviewKD [1]	**77.45**	77.14	70.37	69.89	77.78	74.53
Logit	KD [10]	74.07	74.83	67.37	67.35	74.45	71.60
	DML [35]	72.89	72.76	65.63	65.71	73.45	70.09
	TAKD [18]	74.53	75.34	67.91	68.02	74.82	72.12
	Ours	77.18	**77.44**	**70.57**	**71.04**	**78.44**	**74.93**

Table 4. **Results on ImageNet.** Top-1 and Top-5 accuracy is adopted as the evaluation metric. The original accuracies of the teacher and student model are also reported.

		Top-1	Top-5	Top-1	Top-5
Method	Teacher	ResNet34		ResNet50	
		73.31	91.42	76.16	92.86
	Student	ResNet18		MobileNet-V2	
		69.75	89.07	68.87	88.76
Feature	AT [33]	70.69	90.01	69.56	89.33
	OFD [8]	70.81	89.98	71.25	90.34
	CRD [27]	71.17	90.13	71.37	90.41
	ReviewKD [1]	71.61	90.51	72.56	91.00
Logit	KD [10]	70.66	89.88	68.58	88.98
	DML [35]	70.82	90.02	71.35	90.31
	TAKD [18]	70.78	90.16	70.82	90.01
	DKD [36]	71.70	90.41	72.05	91.05
	Ours	**71.90**	**90.55**	**73.01**	**91.42**

AP_{50}, and AP_{75} as the evaluation metric. The results in Table 5 validate that our method is steadily superior to mainstream KD methods and enjoys strong performance over previous feature distillation methods.

4.3. Analyses

The experiment results above validate the effectiveness of our method in both image classification and object detection. Then we conduct more analysis on our method.

Ablation study We investigate the contributions of each component in our method, instance-level alignment, batch-level alignment, class-level alignment, and prediction augmentation. In Table 6, when merely instance-level alignment is adopted, the method shrinks to the original KD [10] method, while with all the four components, our method performs better than all the other variants, proving that each part of our method is indispensable.

Table 5. **Results on MS-COCO.** We take Faster-RCNN [21]-FPN [15] as the backbone, and AP, AP_{50}, and AP_{75} as the evaluation metric. The original accuracies of the teacher and student model are also reported.

		AP	AP_{50}	AP_{75}	AP	AP_{50}	AP_{75}	AP	AP_{50}	AP_{75}
Method	Teacher	ResNet101			ResNet101			ResNet50		
		42.04	62.48	45.88	42.04	62.48	45.88	40.22	61.02	43.81
	Student	ResNet18			ResNet50			MobileNetV2		
		33.26	53.61	35.26	37.93	58.84	41.05	29.47	48.87	30.90
Feature	FitNet [22]	34.13	54.16	36.71	38.76	59.62	41.80	30.20	49.80	31.69
	FGFI [29]	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
Logit	KD [10]	33.97	54.66	36.62	38.35	59.41	41.71	30.13	50.28	31.35
	TAKD [18]	34.59	55.35	37.12	39.01	60.32	43.10	31.26	51.03	33.46
	DKD [36]	35.05	56.60	37.54	39.25	60.90	42.73	32.34	53.77	34.01
	Ours	**36.03**	**57.28**	**38.51**	40.15	**61.67**	**44.57**	**33.83**	**54.01**	**35.22**

Table 6. **Ablation Study.** The experients are conducted on CIFAR-100, with ResNet32x4 as the teacher model, ResNet8x4 as the student model, and Top-1 accuracy as the evaluation metric.

Instance-level Alignment	Batch-level Alignment	Class-level Alignment	Prediction Augmentation	Acc
✓	✗	✗	✗	73.33
✓	✓	✗	✗	74.58
✓	✓	✓	✗	76.26
✓	✓	✓	✓	**77.08**

Comparison with Teacher Model It is interesting to explore the performance gap between the student and teacher model in our method. In Table 7, we calculate the gap between the accuracies of the student and teacher model, we note that the gap is negative when the student model outperforms the teacher model. We can observe that with our carefully designed method, the student model performance is highly close to the teacher model, with an average accuracy gap of 0.23. Another surprising phenomenon is that sometimes the student model even shows slightly stronger performance than the teacher model. We conjecture the cause of it may be that our knowledge distillation method cooperates well with pure supervised learning.

Combination with Feature Distillation Methods Besides serving as a logit distillation method, our method can also be combined with existing feature distillation methods. Since our method does not introduce any external modules, we can easily plug it into a variety of feature distillation methods. In Table 8, we integrate our method with existing feature distillation methods and show the corresponding results. Our method brings about obvious improvements to RKD [19] (71.73 to 75.32) and steadily pushes ReviewKD [1], a readily strong method, to a higher level.

Correlation Matrices In our method, we enforce the student model to absorb the input correlation as well as the category correlation knowledge from the teacher model. Here, we visualize the distance between the input correlation matrices and category correlation matrices of the teacher and student models, respectively. The diagonal values are removed for a clearer demonstration. We take ResNet32x4 as the teacher model and ResNet8x4 as the student model and train them on CIFAR-100 [13]. We calculate the distance on a batch of data with a batch size of 64. As shown in Figure 3, in our method, the student model learns input correlation and category correlation knowledge from the teacher model better, which is consistent with our motivation and experiment results.

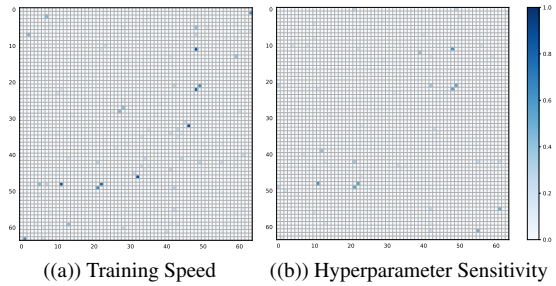
Training Time We compare the training time of various KD methods which enjoy competitive performance by assessing the training time of each batch of data, on CIFAR-100, with the teacher and student models in homogenous architecture. We can observe from Figure 4(a) that our method takes the shortest training time among previous methods. We conjecture that the reason is that our method takes merely the logit outputs to conduct knowledge distillation, while previous methods need more time and compu-

Table 7. **Performance gap between teacher and student model.** Experiments are implemented on CIFAR-100, with teacher and student in homogenous architecture. Top-1 accuracy as the evaluation metric. Note that when the student model outperforms the teacher model, the gap is negative. The settings are the same as Table 2.

Teacher	72.34	74.31	79.42	75.61	75.61	74.64	75.32 (Avg)
Student (Ours)	72.19	74.11	77.08	76.63	75.35	75.18	75.09 (Avg)
Gap	0.15	0.20	2.34	-1.02	0.26	- 0.54	0.23 (Avg)

Table 8. **Combination with Feature Distillation Methods.** Experiments are implemented on CIFAR-100, with teacher and student in homogenous architecture. Top-1 accuracy as the evaluation metric. The settings are the same as Table 2.

RKD [19]	69.61	71.82	71.90	73.35	72.22	71.48	71.73 (Avg)
+ Ours	**72.34**	**74.01**	**77.38**	**76.89**	**75.30**	**75.90**	**75.32** (Avg)
ReviewKD [1]	71.89	73.89	75.63	76.12	75.09	74.84	74.58 (Avg)
+ Ours	**72.83**	**74.52**	**78.01**	**77.54**	**76.21**	**75.69**	**75.80** (Avg)



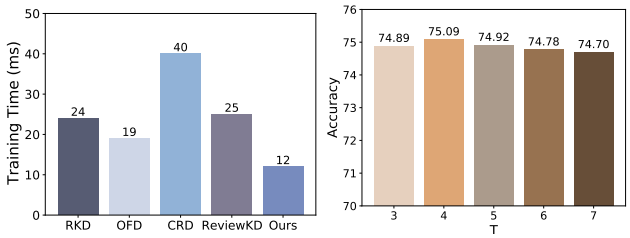
((a)) Training Speed ((b)) Hyperparameter Sensitivity

((c)) Training Speed ((d)) Hyperparameter Sensitivity

Figure 3. **Distance of the correlation matrix** between the teacher and student model. **(a), (b):** Distance of input correlation matrices. **(c), (d):** Distance of category correlation matrices. We take ResNet32x4 as the teacher model and ResNet8x4 as the student model and train them on CIFAR-100. We calculate the input correlation matrix on a batch of data with a batch size of 64. (Clearer figures are included in Appendix.)

tational costs to distill the knowledge in intermediate layers.

Hyperparameter Sensitivity In our experiments, we set the median of temperatures as $T = 4.0$. Here, we conduct hyperparameter sensitivity on T. Following our experiment settings, we take $K = 5$ temperatures with median $T = [3.0, 4.0, 5.0, 6.0, 7.0]$ and evaluate the model on CIFAR-100 with the teacher and student models in homogeneous architecture respectively. The results are shown in



((a)) Training Time ((b)) Hyperparameter Sensitivity

Figure 4. **Training Time and Hyperparameter Sensitivity. (a)** Training time for each batch of data of competitive KD methods. Our method takes the shortest training time among them. **(b)** Our method performs steadily over different T hyperparameter. Both experiments are conducted on CIFAR-100, with the teacher and student models in homogenous architecture.

Figure 4(b). Our method performs stably under different T hyperparameters.

5. Conclusion

In this paper, we propose multi-level logit distillation, a novel approach to make better utilization of logit outputs for knowledge distillation. Concretely, we introduce multi-level alignment, which consists of instance-level, batch-level, and class-level alignment. A prediction augmentation mechanism is proposed to boost the performance. Extensive experiment results prove the effectiveness of our method.

Acknowledgements. This project is funded in part by Shanghai AI Laboratory, CUHK Interdisciplinary AI Research Institute, the Centre for Perceptual and Interactive Intelligence (CPIl) Ltd under the Innovation and Technology Commission (ITC)'s InnoHK, and Hong Kong RGC Theme-based Research Scheme 2020/21 (No. T41-603/20-R).

References

[1] Pengguang Chen, Shu Liu, Hengshuang Zhao, and Jiaya Jia. Distilling knowledge via knowledge review. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2021. 1, 5, 6, 7, 8

[2] Jang Hyun Cho and Bharath Hariharan. On the efficacy of knowledge distillation. In *IEEE International Conference on Computer Vision*, 2019. 2

[3] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. In *International Conference on Learning Representations*, 2021. 1

[4] Tommaso Furlanello, Zachary Lipton, Michael Tschannen, Laurent Itti, and Anima Anandkumar. Born again neural networks. In *International Conference on Machine Learning*, 2018. 2

[5] Ian J Goodfellow, Jonathon Shlens, and Christian Szegedy. Explaining and harnessing adversarial examples. *arXiv preprint arXiv:1412.6572*, 2014. 2

[6] Chuan Guo, Geoff Pleiss, Yu Sun, and Kilian Q Weinberger. On calibration of modern neural networks. In *International Conference on Machine Learning*, pages 1321–1330. PMLR, 2017. 2, 3

[7] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2016. 1, 5

[8] Byeongho Heo, Jeesoo Kim, Sangdoo Yun, Hyojin Park, Nojun Kwak, and Jin Young Choi. A comprehensive overhaul of feature distillation. In *IEEE International Conference on Computer Vision*, 2019. 1, 2, 5, 6

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

[10] Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. Distilling the knowledge in a neural network. In *arXiv:1503.02531*, 2015. 1, 2, 3, 5, 6, 7

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

[12] Jangho Kim, SeongUk Park, and Nojun Kwak. Paraphrasing complex network: Network compression via factor transfer. In *Advances in Neural Information Processing Systems*, 2018. 2

[13] Alex Krizhevsky, Geoffrey Hinton, et al. Learning multiple layers of features from tiny images. 2009. 5, 7

[14] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E. Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in Neural Information Processing Systems*, 2012. 1

[15] Tsung-Yi Lin, Piotr Dollár, Ross Girshick, Kaiming He, Bharath Hariharan, and Serge Belongie. Feature pyramid networks for object detection. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2017. 5, 7

[16] 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 *European Conference on Computer Vision*, 2014. 5

[17] Ningning Ma, Xiangyu Zhang, Hai-Tao Zheng, and Jian Sun. Shufflenet V2: Practical guidelines for efficient cnn architecture design. In *European Conference on Computer Vision*, 2018. 5

[18] Seyed Iman Mirzadeh, Mehrdad Farajtabar, Ang Li, Nir Levine, Akihiro Matsukawa, and Hassan Ghasemzadeh. Improved knowledge distillation via teacher assistant. In *AAAI Conference on Artificial Intelligence*, 2020. 2, 6, 7

[19] Wonpyo Park, Dongju Kim, Yan Lu, and Minsu Cho. Relational knowledge distillation. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2019. 1, 2, 5, 6, 7, 8

[20] Baoyun Peng, Xiao Jin, Jiaheng Liu, Dongsheng Li, Yichao Wu, Yu Liu, Shunfeng Zhou, and Zhaoning Zhang. Correlation congruence for knowledge distillation. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2019. 2

[21] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. In *Advances in Neural Information Processing Systems*, 2015. 1, 5, 7

[22] Adriana Romero, Nicolas Ballas, Samira Ebrahimi Kahou, Antoine Chassang, Carlo Gatta, and Yoshua Bengio. Fitnets: Hints for thin deep nets. *International Conference on Learning Representations*, 2015. 1, 2, 5, 6, 7

[23] 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. *International Journal of Computer Vision*, 2015. 5

[24] Mark Sandler, Andrew Howard, Menglong Zhu, Andrey Zhmoginov, and Liang-Chieh Chen. MobilenetV2: Inverted residuals and linear bottlenecks. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2018. 5

[25] Evan Shelhamer, Jonathan Long, and Trevor Darrell. Fully convolutional networks for semantic segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2016. 1

[26] K. Simonyan and A Zisserman. Very deep convolutional networks for large-scale image recognition. In *International Conference on Learning Representations*, May 2015. 1, 5

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

[28] Frederick Tung and Greg Mori. Similarity-preserving knowledge distillation. In *IEEE International Conference on Computer Vision*, 2019. 2

[29] Tao Wang, Li Yuan, Xiaopeng Zhang, and Jiashi Feng. Distilling object detectors with fine-grained feature imitation. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2019. 7

[30] Chenglin Yang, Lingxi Xie, Chi Su, and Alan L Yuille. Snapshot distillation: Teacher-student optimization in one gener-

ation. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2019. 2

[31] Junho Yim, Donggyu Joo, Jihoon Bae, and Junmo Kim. A gift from knowledge distillation: Fast optimization, network minimization and transfer learning. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2017. 2

[32] Sergey Zagoruyko and Nikos Komodakis. Wide residual networks. In *British Machine Vision Conference*, 2016. 5

[33] Sergey Zagoruyko and Nikos Komodakis. Paying more attention to attention: Improving the performance of convolutional neural networks via attention transfer. *International Conference on Learning Representations*, 2017. 2, 6

[34] Xiangyu Zhang, Xinyu Zhou, Mengxiao Lin, and Jian Sun. Shufflenet: An extremely efficient convolutional neural network for mobile devices. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 6848–6856, 2018. 5

[35] Ying Zhang, Tao Xiang, Timothy M Hospedales, and Huchuan Lu. Deep mutual learning. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2018. 2, 6

[36] Borui Zhao, Quan Cui, Renjie Song, Yiyu Qiu, and Jiajun Liang. Decoupled knowledge distillation. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 11953–11962, 2022. 6, 7

[37] Hengshuang Zhao, Jianping Shi, Xiaojuan Qi, Xiaogang Wang, and Jiaya Jia. Pyramid scene parsing network. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2017. 1