Dual-Head Knowledge Distillation: Enhancing Logits Utilization with an Auxiliary Head

Penghui Yang
Nanyang Technological University
Singapore
penghui004@e.ntu.edu.sg

Chen-Chen Zong
Nanjing University of Aeronautics and Astronautics
Nanjing, China
chencz@nuaa.edu.cn

Sheng-Jun Huang
Nanjing University of Aeronautics and Astronautics
Nanjing, China
huangsj@nuaa.edu.cn

Lei Feng*
Southeast University
Nanjing, China
fenglei@seu.edu.cn

Bo An
Nanyang Technological University
Singapore
boan@ntu.edu.sg

Abstract

Traditional knowledge distillation focuses on aligning the student's predicted probabilities with both ground-truth labels and the teacher's predicted probabilities. However, the transition to predicted probabilities from logits would obscure certain indispensable information. To address this issue, it is intuitive to additionally introduce a logit-level loss function as a supplement to the widely used probability-level loss function, for exploiting the latent information of logits. Unfortunately, we empirically find that the amalgamation of the newly introduced logit-level loss and the previous probability-level loss will lead to performance degeneration, even trailing behind the performance of employing either loss in isolation. We attribute this phenomenon to the collapse of the classification head, which is verified by our theoretical analysis based on the *neural collapse* theory. Specifically, the gradients of the two loss functions exhibit contradictions in the linear classifier yet display no such conflict within the backbone. Drawing from the theoretical analysis, we propose a novel method called *dual-head knowledge distillation*, which partitions the linear classifier into two classification heads responsible for different losses, thereby preserving the beneficial effects of both losses on the backbone while eliminating adverse influences on the classification head. Extensive experiments validate that our method can effectively exploit the information inside the logits and achieve superior performance against state-of-the-art counterparts[1].

CCS Concepts

• **Computing methodologies → Neural networks**; **Supervised learning by classification**.

Keywords

Knowledge Distillation, Neural Collapse

ACM Reference Format:

Penghui Yang, Chen-Chen Zong, Sheng-Jun Huang, Lei Feng, and Bo An. 2025. Dual-Head Knowledge Distillation: Enhancing Logits Utilization with an Auxiliary Head. In *Proceedings of the 31st ACM SIGKDD Conference on Knowledge Discovery and Data Mining V.2 (KDD '25), August 3–7, 2025, Toronto, ON, Canada.* ACM, New York, NY, USA, 12 pages. https://doi.org/10.1145/3711896.3736915

1 Introduction

Despite the remarkable success of deep neural networks (DNNs) in various fields, it is a significant challenge to deploy these large models in lightweight terminals (e.g., mobile phones), particularly under the constraint of computational resources or the requirement of short inference time. To mitigate this problem, knowledge distillation (KD) [11] is widely investigated, which aims to improve the performance of a small network (*a.k.a.* the "student") by leveraging the expansive knowledge of a large network (*a.k.a.* the "teacher") to guide the training of the student network.

Traditional KD techniques focus on minimizing the disparity in the predicted probabilities between the teacher and the student, which are typically the outputs of the softmax function. Nevertheless, the transformation from logits to predictive probabilities via the softmax function may lose some underlying information. As shown in Figure 1(a), considering a 3-class classification problem, even if the teacher model outputs two different logit vectors $[2, 3, 4]$ and $[-2, -1, 0]$, the softmax function renders the same probability vector $[0.09, 0.24, 0.67]$. However, different logit vectors may carry different underlying information that would be further exploited by the student, which could be lost due to the transformation process carried out by the softmax function.

In order to properly leverage the information inside the logits, we introduce a logit-level KD loss. Specifically, we use the sigmoid function to formalize the pre-softmax output for each class into a range of $[0, 1]$ and deploy the Kullback-Leibler (KL) divergence to perform the binary classification of each class. By aligning the pre-softmax output of each class separately, this newly introduced loss (denoted by *BinaryKL*) can adequately exploit the information inside the logits. However, it is interesting to show that combining the logit-level BinaryKL loss with the probability-level cross-entropy

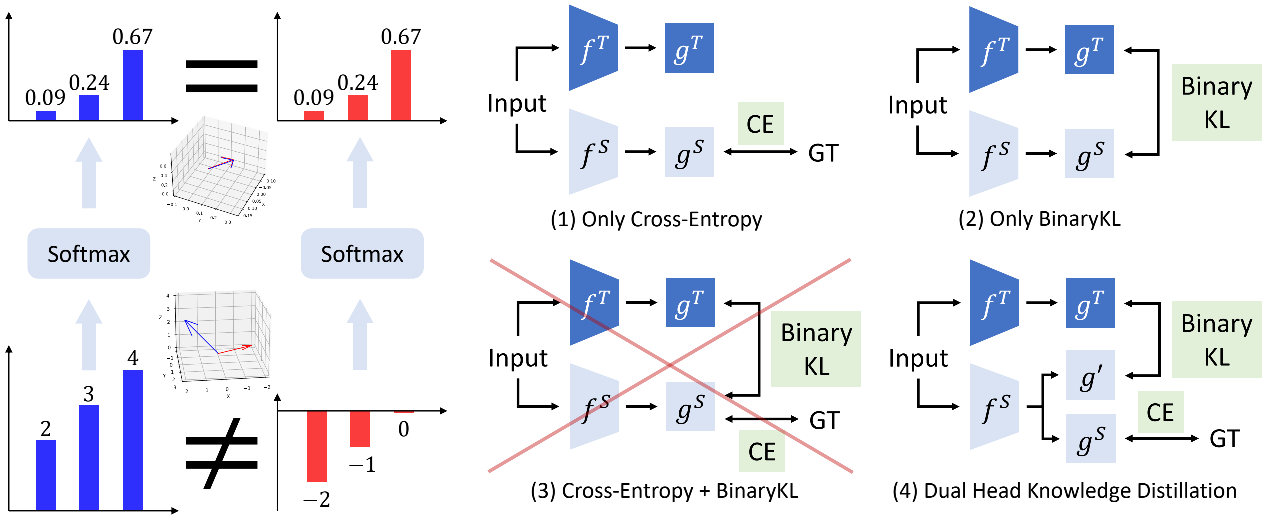
*Corresponding Author.
[1] The full technical report is available at https://arxiv.org/abs/2411.08937

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than the author(s) must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.

KDD '25, Toronto, ON, Canada

© 2025 Copyright held by the owner/author(s). Publication rights licensed to ACM.

ACM ISBN 979-8-4007-1454-2/2025/08
https://doi.org/10.1145/3711896.3736915



(1) Only Cross-Entropy

(2) Only BinaryKL

(3) Cross-Entropy + BinaryKL

(4) Dual Head Knowledge Distillation

(a) Information loss through softmax

(b) An illustration of four settings we evaluate

Figure 1: The reason for introducing the BinaryKL loss and the incompatibility of the CE loss and the BinaryKL loss. Figure 1(a) shows that two different vectors may become the same through the softmax function, which means some information would be lost during the transformation process carried out by the softmax function. Figure 1(b) shows four settings we evaluate: (1) only cross-entropy (CE) loss; (2) only the BinaryKL loss; (3) CE + BinaryKL; (4) Our proposed Dual-Head Knowledge Distillation (DHKD). The red cross over the third setting means that the student model trained under this setting will collapse. The performance sorting of four settings is (4) > (1) > (2) ≫ (3), as shown in Figure 2.

(CE) loss results in poor performance of the student model, which is even worse than the performance of employing either loss separately. As illustrated in Figure 1(b), employing either loss separately as the training loss can induce a high-performance student model, but the model's performance will be largely degraded when we combine them together during the training process.

To identify the root cause of this abnormal phenomenon, inspired by the recent research on *neural collapse* [39], we analyze the gradients of the model when using both losses. Specifically, by dividing the gradients of the model into two parts, we find that the gradients of the two loss functions contradict each other regarding the linear classifier head but display no such conflict regarding the backbone. As a consequence, while the BinaryKL loss can facilitate the backbone in learning from the teacher model more precisely, it prevents the linear classifier head from converging to a simplex equiangular tight frame (see a detailed definition in Definition 1), which is the ideal status that a well-trained linear classifier should converge to. Therefore, the combination of these two loss functions would cause the linear classifier to collapse, thereby degrading the performance of the student model. Based on our theoretical analysis that a single linear classifier is faced with the gradient contradiction when being trained by both losses simultaneously, we propose Dual-Head Knowledge Distillation (DHKD), which introduces an auxiliary classifier head apart from the original linear classifier to effectively circumvent the collapse of the linear classifier while retaining the positive effects of the BinaryKL loss on the backbone. Extensive experiments show the advantage of our proposed DHKD method.

Our main contributions can be summarized as follows:

- We disclose an interesting phenomenon in the knowledge distillation scenario: combining the probability-level CE loss and the logit-level BinaryKL loss would cause a performance drop of the student model, compared with using either loss separately.
- We provide theoretical analyses to explain the discordance between the BinaryKL loss and the CE loss. While the BinaryKL loss aids in cultivating a stronger backbone, it harms the performance of the linear classifier head.
- We propose a novel knowledge distillation method called Dual-Head Knowledge Distillation (DHKD). Apart from the linear classifier trained with the CE loss, DHKD specially introduces an auxiliary classifier trained with the BinaryKL loss.

Extensive experiments demonstrate the effectiveness of our proposed method.

2 Related Work

2.1 Knowledge Distillation

Knowledge distillation (KD) [11] aims to transfer knowledge from a large teacher network to a small student network. Existing works can be roughly divided into two groups: feature-based methods and logit-based methods.

Feature-based methods focus on distilling knowledge from intermediate feature layers. FitNet [29] is the first approach to distill knowledge from intermediate features by measuring the distance between feature maps. RKD [26] utilizes the relations among instances to guide the training process of the student model. CRD [33] incorporates contrastive learning into knowledge distillation. OFD [10] contains a new distance function to distill significant information between the teacher and student using marginal ReLU. ReviewKD [3] proposes a review mechanism that uses multiple layers in the teacher to supervise one layer in the student. Other papers [15, 17, 19, 21, 24, 27, 30, 35] enforce various criteria based on features. Most feature-based methods can attain superior performance, yet involving considerably high computational and storage costs.

Logit-based methods mainly concentrate on distilling knowledge from logits and softmax scores after logits. DML [43] introduces a mutual learning method to train both teachers and students simultaneously. DKD [44] proposes a novel logit-based method to reformulate the classical KD loss into two parts and achieves state-of-the-art performance by adjusting weights for these two parts. DIST [12] relaxes the exact matching in previous KL divergence loss with a correlation-based loss and performs better when the discrepancy between the teacher and the student is large. TTM [45] drops the temperature scaling on the student side, which causes an inherent Renyi entropy term as an extra regularization term in the loss function. Although logit-based methods require fewer computational and storage resources, they experience a performance gap compared with feature-based methods.

2.2 Neural Collapse

Neural collapse is a phenomenon observed in the late stages of training a deep neural network, which is especially evident in the classification tasks [25]. As the training process approaches its optimum, the feature representations of data points belonging to the same class tend to converge to a single point, or at least become significantly more similar to each other in the feature space. At the same time, the class mean vectors tend to be equidistant and form a simplex equiangular tight frame (see a detailed definition in Definition 1).

Although the phenomenon is intuitive, its reason has not been entirely understood, which inspires several lines of theoretical work on it. Papyan et al. [25] prove that if the features satisfy neural collapse, the optimal classifier vectors under the MSE loss will also converge to neural collapse. Some studies turn to a simplified model that only considers the last-layer features and the classifier as independent variables, and they prove that neural collapse emerges under the CE loss with proper constraints or regularization [5, 13, 22, 36, 49]. Yang et al. [39] also use such a simplified model and decompose the gradient into two parts to help build a better classifier for class-imbalanced learning, which is the primary technical reference for the theoretical analysis section of this article.

2.3 Decoupled Heads

Using multiple linear classifiers is a common strategy in various fields. In object detection, to address the conflict between classification and regression tasks, the decoupled head for classification and localization is widely used in most one-stage and two-stage detectors [6, 20, 37]. In multitask learning, it is common practice to jointly train various tasks through the shared backbone [1, 14, 46]. In long-tailed learning, the Bilateral-Branch Network takes care of both representation learning and classifier learning concurrently, where each branch performs its own duty separately [47].

3 Dual-Head Knowledge Distillation

In this section, we firstly point out the incompatibility between the CE loss and the BinaryKL loss in Section 3.1. Then we analyze its theoretical foundation in Section 3.2. Finally, based on the theoretical analysis, we introduce our Dual-Head Knowledge Distillation (DHKD) method in Section 3.3.

3.1 Incompatibility between CE and BinaryKL

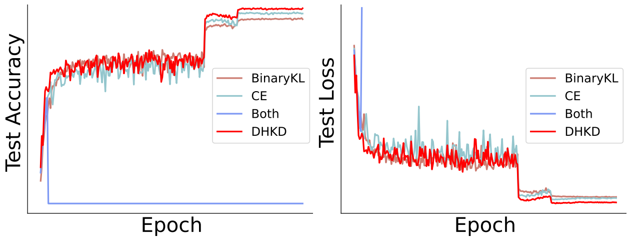
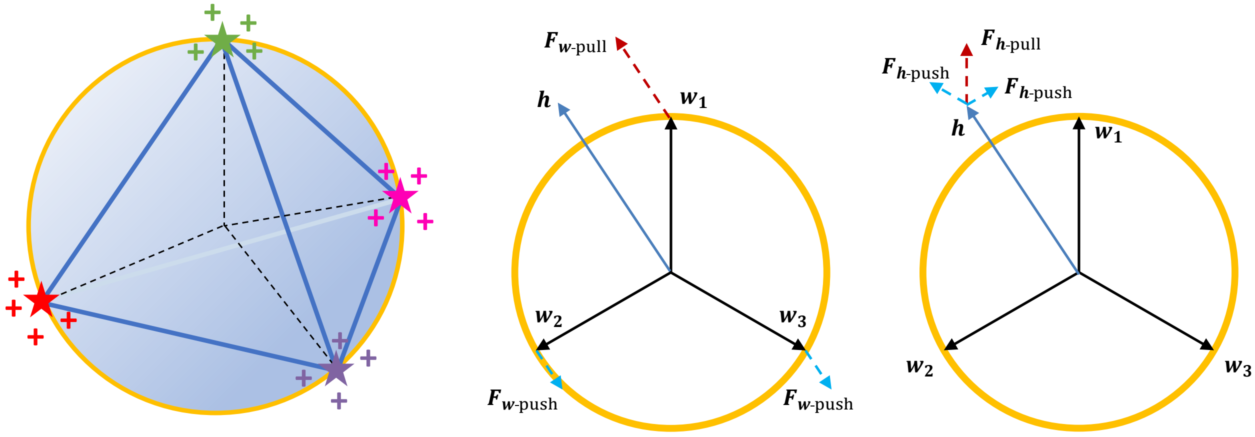


Figure 2: The test accuracy and test loss curves of the student models during the training phase. We set resnet56 as the teacher and resnet20 as the student on the CIFAR-100 dataset.

Traditional KD methods only minimize the difference in the predicted probabilities (*i.e.*, the outputs of the softmax function) between the teacher and the student. However, some underlying information may be lost when converting logits to predicted probabilities using the softmax function. For example, in a 3-class classification problem in Figure 1(a), if the teacher model outputs logit vectors $[2, 3, 4]$ and $[-2, -1, 0]$ for different instances, after being processed by the softmax function, their predicted probabilities are both $[0.09, 0.24, 0.67]$. Such differences in logits may carry different hidden information that would be further utilized by the student but could be lost because of the transformation process carried out by the softmax function.

To fully exploit the information inside the logits, we introduce a logit-level KD loss. Previous studies in various fields have delved into some logit-level loss functions, such as the mean squared error and the mean absolute error [16]. We explore a recently proposed method called BinaryKL [38], which uses the sigmoid function to formalize the pre-softmax output for each class into a range of $[0, 1]$ and deploys the Kullback-Leibler (KL) divergence as the binary classification of each class. By aligning the pre-softmax output of each class separately, the newly introduced loss can better exploit the information inside the logits.

Formally, with the logits $z^{\mathcal{S}} \in \mathbb{R}^{B\times K}$ and $z^{\mathcal{T}} \in \mathbb{R}^{B\times K}$ of the student model and the teacher model, where B and K denote batch size and the number of classes respectively, the BinaryKL loss can



(a) A simplex equiangular tight frame

(b) Gradient directions *w.r.t.* $\boldsymbol{w}$

(c) Gradient directions *w.r.t.* $\boldsymbol{h}$

Figure 3: Gradient analysis based on the *neural collapse* theory. (a) An illustration of a simplex ETF when $d = 3$ and $K = 4$. The "+" and "☆" with different colors refer to features and classifier vectors of different classes, respectively. (b) Gradient directions about a certain $\boldsymbol{h}$ (belongs to the 1-st class) *w.r.t.* all $\boldsymbol{w}_i$, $i \in 1, 2, 3$. (c) Gradient directions *w.r.t.* an $\boldsymbol{h}$ (belongs to the 1-st class).

be formulated as follows:

$$\mathcal{L}_{\text{BinaryKL}} = \tau^2 \sum_{i=1}^{B} \sum_{k=1}^{K} \mathcal{KL}\Big(\left[\sigma\left(z_{i,k}^{\mathcal{T}}/\tau\right), 1-\sigma\left(z_{i,k}^{\mathcal{T}}/\tau\right)\right] \\ \| \left[\sigma\left(z_{i,k}^{\mathcal{S}}/\tau\right), 1-\sigma\left(z_{i,k}^{\mathcal{S}}/\tau\right)\right] \Big), \tag{1}$$

where $\sigma(\cdot)$ is the sigmoid function, $[\cdot,\cdot]$ is an operator used to concatenate two scalars into a vector, and $\mathcal{KL}$ denotes the KL divergence $\mathcal{KL}(P\|Q) = \sum_{x\in\mathcal{X}} P(\boldsymbol{x}) \log\left(P(\boldsymbol{x})/Q(\boldsymbol{x})\right)$, where P and Q are two different probability distributions.

As shown in Figures 1 and 2, although the BinaryKL loss performs well alone, it is incompatible with the CE loss. The student model can achieve descent performance by training with either the CE loss or the BinaryKL loss separately, but the performance will be degraded severely when we combine them as follows:

$$\mathcal{L}_{\text{overall}} = \mathcal{L}_{\text{CE}} + \alpha\mathcal{L}_{\text{BinaryKL}}. \tag{2}$$

It seems that this problem can be addressed by decreasing the balancing parameter α, reducing the learning rate, or using clipping the gradients, but our experiments show that these three solutions either do not work or weaken the effects of the BinaryKL loss (see more details in Appendix A).

3.2 Theoretical Analysis

To theoretically analyze the abnormal phenomenon above, we conduct gradient analyses by calculating the gradients of both the CE loss and the BinaryKL loss *w.r.t.* the linear classifier and the feature. Our gradient analyses reveal that the gradient of the BinaryKL loss *w.r.t.* the linear classifier obstructs it from achieving the ideal state posited by the *neural collapse* theory. Therefore, before delving into the gradient analyses, we provide a brief introduction to the *neural collapse* theory below.

Papyan et al. [25] revealed the neural collapse phenomenon, where the last-layer features converge to their within-class means, and the within-class means together with the classifier vectors collapse to the vertices of a simplex equiangular tight frame (ETF) at the terminal phase of training. According to Papyan et al. [25], all vectors in a simplex ETF have an equal ℓ_2 norm and the same pairwise angle. An illustration of a simplex ETF is shown in Figure 3. The neural collapse phenomenon [25] can be characterized by four manifestations: (1) the variability of the last-layer features in the same class collapses to zero; (2) the class means of different classes' last-layer features converge to a simplex ETF; (3) the linear classifier vectors collapse to the class means; (4) the task of predicting for classification can be simplified into finding out the nearest class center of the last-layer feature. The detailed definition of a simplex ETF and the features of the neural collapse phenomenon can be found in Appendix B.1.

From the above analysis, we can find that a well-trained linear classifier would collapse into a simplex ETF, which implies that any factor that hinders the linear classifier from forming a simplex ETF can adversely affect the training of the linear classifier and thus detrimentally impact the overall training of the student model. It is noteworthy that although the last-layer features in the same class tend to collapse into their mean value as well, this is not the ideal status of a well-trained linear classifier, and such a phenomenon is mainly called over-fitting. Inspired by Yang et al. [39], we decompose the gradients of $\mathcal{L}_{\text{CE}}$ and $\mathcal{L}_{\text{BinaryKL}}$ to figure out their effects on the linear classifier and the feature extracted by the backbone. First, we decompose the gradients *w.r.t.* the linear classifier. Let $\boldsymbol{h}_{k,i}$ be the feature of the i-th item in the k-th class extracted by the backbone and $\boldsymbol{w}$ be the linear classifier where $\boldsymbol{w} = [\boldsymbol{w}_1, \ldots, \boldsymbol{w}_K] \in \mathbb{R}^{d\times K}$. Let n_k be the number of instances in the k-th class in a certain batch. We denote the k-th class output through the softmax function as $p_k(\cdot)$ and the k-th class output through the sigmoid function as $q_k(\cdot)$. The detailed definitions of $p_k(\cdot)$ and $q_k(\cdot)$ can be found in Appendix B.2.1.

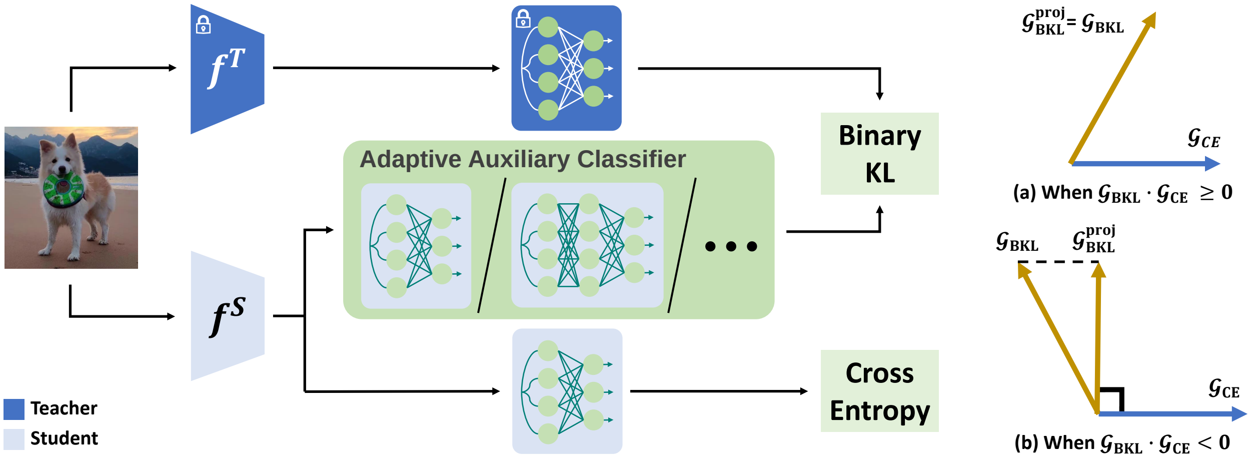

f^T
f^S
Adaptive Auxiliary Classifier
Binary KL
Cross Entropy
Teacher
Student
$\mathcal{G}^{\text{proj}}_{\text{BKL}} = \mathcal{G}_{\text{BKL}}$
$\mathcal{G}_{\text{CE}}$
(a) When $\mathcal{G}_{\text{BKL}} \cdot \mathcal{G}_{\text{CE}} \geq 0$
$\mathcal{G}_{\text{BKL}}$
$\mathcal{G}^{\text{proj}}_{\text{BKL}}$
$\mathcal{G}_{\text{CE}}$
(b) When $\mathcal{G}_{\text{BKL}} \cdot \mathcal{G}_{\text{CE}} < 0$

Figure 4: An illustration of Dual-Head Knowledge Distillation (DHKD). DHKD decouples the original linear classifier into a duo. We can evade conflicts between two losses by introducing an Adaptive Auxiliary Classifier customized according to the models' architectures. The right side shows the gradient alignment method on CIFAR-100. When the angle between two gradients is larger than 90°, we will project the gradient of the BinaryKL loss to the orthogonal direction of the gradient of the CE loss.

Proposition 1. *The gradient of $\mathcal{L}_{\text{overall}}$ w.r.t. the linear classifier can be formulated as follows:*

$$\frac{\partial \mathcal{L}_{\text{overall}}}{\partial \boldsymbol{w}_k} = -(\boldsymbol{F}^{\text{CE}}_{\boldsymbol{w}\text{-pull}} + \alpha \boldsymbol{F}^{\text{BinaryKL}}_{\boldsymbol{w}\text{-pull}}) - (\boldsymbol{F}^{\text{CE}}_{\boldsymbol{w}\text{-push}} + \alpha \boldsymbol{F}^{\text{BinaryKL}}_{\boldsymbol{w}\text{-push}}), \tag{3}$$

where

$$\begin{aligned}
\boldsymbol{F}^{\text{CE}}_{\boldsymbol{w}\text{-pull}} &= \sum\nolimits_{i=1}^{n_k} (1 - p_k(\boldsymbol{h}^{\mathcal{S}}_{k,i}))\boldsymbol{h}^{\mathcal{S}}_{k,i}, \\
\boldsymbol{F}^{\text{BinaryKL}}_{\boldsymbol{w}\text{-pull}} &= \tau \sum\nolimits_{i=1}^{n_k} (q_k(\boldsymbol{h}^{\mathcal{T}}_{k,i}) - q_k(\boldsymbol{h}^{\mathcal{S}}_{k,i}))\boldsymbol{h}^{\mathcal{S}}_{k,i}, \\
\boldsymbol{F}^{\text{CE}}_{\boldsymbol{w}\text{-push}} &= -\sum\nolimits_{k' \neq k}^{K} \sum\nolimits_{j=1}^{n_{k'}} p_k(\boldsymbol{h}^{\mathcal{S}}_{k',i})\boldsymbol{h}^{\mathcal{S}}_{k',j}, \\
\boldsymbol{F}^{\text{BinaryKL}}_{\boldsymbol{w}\text{-push}} &= -\tau \sum\nolimits_{k' \neq k}^{K} \sum\nolimits_{j=1}^{n_{k'}} (q_k(\boldsymbol{h}^{\mathcal{S}}_{k,i}) - q_k(\boldsymbol{h}^{\mathcal{T}}_{k,i}))\boldsymbol{h}^{\mathcal{S}}_{k',j}.
\end{aligned} \tag{4}$$

The proof of Proposition 1 is provided in Appendix B.2.1. The gradient *w.r.t.* $\boldsymbol{w}_k$ can be decomposed into four terms. As Figure 3(b) shows, on one hand, the "pull" terms $\boldsymbol{F}^{\text{CE}}_{\boldsymbol{w}\text{-pull}}$ and $\boldsymbol{F}^{\text{BinaryKL}}_{\boldsymbol{w}\text{-pull}}$ pull $\boldsymbol{w}_k$ towards feature directions of the same class, *i.e.*, $\boldsymbol{h}_{k,i}$; on the other hand, the "push" terms $\boldsymbol{F}^{\text{CE}}_{\boldsymbol{w}\text{-push}}$ and $\boldsymbol{F}^{\text{BinaryKL}}_{\boldsymbol{w}\text{-push}}$ push $\boldsymbol{w}_k$ away from the feature directions of the other classes, *i.e.*, $\boldsymbol{h}_{k',i}$, for all $k' \neq k$. The terms *w.r.t.* the CE loss keep the signs unchanged, which means each coefficient of $\boldsymbol{h}_{k,i}$ in $\boldsymbol{F}^{\text{CE}}_{\boldsymbol{w}\text{-pull}}$ is positive and each coefficient of $\boldsymbol{h}_{k',j}$ in $\boldsymbol{F}^{\text{CE}}_{\boldsymbol{w}\text{-push}}$ is negative.

However, this does not hold for the terms *w.r.t.* the BinaryKL loss. The coefficients of $\boldsymbol{h}_{k,i}$ in $\boldsymbol{F}^{\text{BinaryKL}}_{\boldsymbol{w}\text{-pull}}$ may be negative and the coefficients of $\boldsymbol{h}_{k',j}$ in $\boldsymbol{F}^{\text{BinaryKL}}_{\boldsymbol{w}\text{-push}}$ may be positive. Such coefficients can reduce the absolute values of the gradients in the optimization direction. In the worst-case scenario, suppose the absolute values of these coefficients are larger than those of the CE loss coefficients but have the opposite signs, these coefficients can even guide the optimization to the opposite direction. Hence, the BinaryKL loss may obstruct the linear classifier's learning process. Because a well-trained linear classifier will become a simplex ETF, the terms *w.r.t.* the BinaryKL loss will harm the training of the linear classifier when their corresponding coefficients have the contrary sign to the terms *w.r.t.* the CE loss.

Then, we decompose the gradients *w.r.t.* the feature extracted by the backbone. Suppose the instance $\boldsymbol{x}$ has a feature $\boldsymbol{h}$ and belongs to the c-th class. Similar to the gradients *w.r.t.* the linear classifier, we can calculate the gradients *w.r.t.* the features.

Proposition 2. *The gradient of $\mathcal{L}_{\text{overall}}$ w.r.t. the features can be formulated as follows:*

$$\frac{\partial \mathcal{L}_{\text{overall}}}{\partial \boldsymbol{h}} = -(\boldsymbol{F}^{\text{CE}}_{\boldsymbol{h}\text{-pull}} + \alpha \boldsymbol{F}^{\text{BinaryKL}}_{\boldsymbol{h}\text{-pull}}) - (\boldsymbol{F}^{\text{CE}}_{\boldsymbol{h}\text{-push}} + \alpha \boldsymbol{F}^{\text{BinaryKL}}_{\boldsymbol{h}\text{-push}}), \tag{5}$$

where

$$\begin{aligned}
\boldsymbol{F}^{\text{CE}}_{\boldsymbol{h}\text{-pull}} &= (1 - p_c(\boldsymbol{h}^{\mathcal{S}}))\boldsymbol{w}^{\mathcal{S}}_c, \\
\boldsymbol{F}^{\text{BinaryKL}}_{\boldsymbol{h}\text{-pull}} &= \tau(q_c(\boldsymbol{h}^{\mathcal{T}}) - q_c(\boldsymbol{h}^{\mathcal{S}}))\boldsymbol{w}^{\mathcal{S}}_c, \\
\boldsymbol{F}^{\text{CE}}_{\boldsymbol{h}\text{-push}} &= -\sum\nolimits_{k \neq c}^{K} p_k(\boldsymbol{h}^{\mathcal{S}})\boldsymbol{w}^{\mathcal{S}}_k, \\
\boldsymbol{F}^{\text{BinaryKL}}_{\boldsymbol{h}\text{-push}} &= -\tau \sum\nolimits_{k \neq c}^{K} (q_k(\boldsymbol{h}^{\mathcal{S}}) - q_k(\boldsymbol{h}^{\mathcal{T}}))\boldsymbol{w}^{\mathcal{S}}_k.
\end{aligned} \tag{6}$$

The proof of Proposition 2 is provided in Appendix B.2.2. The gradients *w.r.t.* $\boldsymbol{h}$ are decomposed into four terms. As Figure 3(c) shows, the "pull" terms $\boldsymbol{F}^{\text{CE}}_{\boldsymbol{h}\text{-pull}}$ and $\boldsymbol{F}^{\text{BinaryKL}}_{\boldsymbol{h}\text{-pull}}$ pull $\boldsymbol{h}$ towards the directions of the corresponding class vector, *i.e.*, $\boldsymbol{w}_c$, while the "push" terms $\boldsymbol{F}^{\text{CE}}_{\boldsymbol{h}\text{-push}}$ and $\boldsymbol{F}^{\text{BinaryKL}}_{\boldsymbol{h}\text{-push}}$ push $\boldsymbol{h}$ away from the directions of the other class vectors, *i.e.*, $\boldsymbol{w}_k$, for all $k \neq c$. Unlike the linear classifier, a well-trained backbone will not let the features of the same class collapse into a specific vector. The differences among the features of the same class contain essential information from the teacher model. Thus, the terms *w.r.t.* the BinaryKL loss will provide more detailed information about the knowledge learned by the teacher model. As a result, the backbone of the student model can benefit from the terms *w.r.t.* the BinaryKL loss.

3.3 Our Proposed Method

As demonstrated in our theoretical analysis, the BinaryKL loss facilitates the backbone in learning a better backbone but impedes the linear classifier from achieving an Equiangular Tight Frame status. To fully leverage the positive effects of the BinaryKL loss while mitigating its negative impact, we propose a novel method called Dual-Head Knowledge Distillation (DHKD), which restricts the effects of the BinaryKL loss to the backbone, excluding the original linear classifier.

For a given input $\boldsymbol{x}$, we can get a feature $\boldsymbol{h} = f(\boldsymbol{x})$ through the backbone $f(\cdot)$. Just as Figure 4 shows, we use two classifiers for different losses: $g(\cdot)$ for the CE loss and $g'(\cdot)$ for the BinaryKL loss. The original BinaryKL loss suffers from a problem: when the student output falls into a small neighborhood of the teacher output (which means it is close to the optimization goal), the derivative of the BinaryKL loss at the student output also depends on the value of the teacher output. As a result, the optimization progress cannot be synchronized among distinct teacher outputs with the same τ. Although using different τ in different settings can lead to state-of-the-art performance, it would introduce more hyper-parameters. To mitigate this issue, we use a variant of the BinaryKL loss. To unify the gradients at the outputs of the teacher model, we choose to narrow the distance between zero and the difference between the teacher and student. The modified loss (we call it *BinaryKL-Norm*) can be formulated as follows:

$$\mathcal{L}_{\text{BinaryKL-Norm}} = \tau^2 \sum\nolimits_{i=1}^{B} \sum\nolimits_{k=1}^{K} \mathcal{KL}\Bigg(\left[\frac{1}{2}, \frac{1}{2}\right] \Bigg\| \left[\sigma\left(\left(z^{\mathcal{S}}_{i,k} - z^{\mathcal{T}}_{i,k}\right)/\tau\right), 1-\sigma\left(\left(z^{\mathcal{S}}_{i,k} - z^{\mathcal{T}}_{i,k}\right)/\tau\right)\right]\Bigg). \quad (7)$$

It is worth noting that the modification of the loss function does not affect the correctness of our theoretical analysis. The theoretical analysis of the modified loss can be found in Appendix B.3. Given the challenges of aligning logits between models with different architectures, we relax the constraint on the auxiliary classifier $g'(\cdot)$. Instead of utilizing a linear layer, we employ a one-hidden-layer neural network, which provides additional flexibility and capacity for the model to align the logits effectively when the student model's architecture diverges from the teacher's.

To further improve the stability of the training phase for the CIFAR-100 dataset, we introduce the gradient alignment technique [40], which was widely used to handle the conflicts among diverse targets [8, 28, 48]. We denote $\frac{\partial \mathcal{L}_{\text{BinaryKL-Norm}}}{\partial \boldsymbol{w}_k}$ as $\mathcal{G}_{\text{BKL}}$ and $\frac{\partial \mathcal{L}_{\text{CE}}}{\partial \boldsymbol{w}_k}$ as $\mathcal{G}_{\text{CE}}$, respectively. The relations between $\mathcal{G}_{\text{BKL}}$ and $\mathcal{G}_{\text{CE}}$ are two-fold. (1) Their angle is smaller than 90°, which indicates that the optimization direction of the BinaryKL loss does not conflict with the CE loss. In this case, we simply set the updated gradient direction of the auxiliary classifier as $\mathcal{G}_{\text{BKL}}$; (2) Their angle is larger than 90°, indicating that the BinaryKL loss conflicts with the CE loss. In other words, optimizing the neural network following the BinaryKL loss will weaken the performance of classification. In this case, we project the $\mathcal{G}_{\text{BKL}}$ to the orthogonal direction of $\mathcal{G}_{\text{CE}}$ to optimize the model by the BinaryKL-Norm loss, which avoids increasing the loss for classification. The modified gradient is mathematically formulated as follows:

$$\mathcal{G}^{\text{proj}}_{\text{BKL}} = \begin{cases} \mathcal{G}_{\text{BKL}}, & \text{if } \mathcal{G}_{\text{BKL}} \cdot \mathcal{G}_{\text{CE}} \geq 0, \\ \mathcal{G}_{\text{BKL}} - \frac{\mathcal{G}_{\text{BKL}} \cdot \mathcal{G}_{\text{CE}}}{\|\mathcal{G}_{\text{CE}}\|^2} \mathcal{G}_{\text{CE}}, & \text{otherwise.} \end{cases} \quad (8)$$

By decoupling the linear classifier and doing gradient alignment, we can preserve the positive effects of the BinaryKL-Norm loss on the backbone and simultaneously avoid its negative impacts on the classifier head. The classifier head is only trained with the CE loss without being aligned with the teacher model because we want to induce it into an ideal simplex ETF.

4 Experiments

Information about the comparing methods and implementation details can be found in Appendices C and D. More experimental results can be found in Appendices A, E and F.

4.1 Main Results

CIFAR-100 image classification. The validation accuracy on CIFAR-100 is reported in Table 1 for cases where teachers and students share the same architecture and in Table 2 for cases where they have different architectures. It can be observed that DHKD achieves remarkable improvements over the vanilla KD on all teacher-student pairs. Compared with the SOTA logit-based methods, our method can achieve better performance in most settings. Furthermore, when combined with one of the SOTA feature-based methods ReviewKD, our method can achieve the best performance, showing that our method is compatible with feature-based methods.

ImageNet image classification. The top-1 and top-5 validation accuracy on ImageNet is reported in Table 3 for cases where teachers and students share the same architecture and in Table 4 for cases where they have different architectures. We can find that our DHKD achieves comparable or even better performance than the existing methods. The success on the large-scale dataset further proves the effectiveness of our method.

4.2 Ablation Studies

To further analyze how our proposed method improves distillation performance, Table 5 reports the results of the ablation studies on CIFAR-100. We choose one pair of models with the same architecture and another pair with different architectures. It can be observed that most of the performance improvement comes from using our DHKD. Introducing a nonlinear auxiliary classifier helps the pair with different architectures achieve better performance, but it harms the performance of the pair with the same architecture, which confirms the rationality of our selective use of a nonlinear auxiliary classifier. By incorporating these components together, the fusing method achieves the best performance and significantly outperforms the other methods. These results demonstrate that all components are of great importance to the performance of our proposed DHKD.

4.3 t-SNE Visualization of the Features

t-SNE (t-distributed Stochastic Neighbor Embedding) [34] is a famous unsupervised non-linear dimensionality reduction technique

Table 1: Results on the CIFAR-100 validation. Teachers and students are in the same architectures. The best performance of a single method is highlighted with a star *, and the best logits-based performance is highlighted in bold. The combination of DHKD and ReviewKD always achieves the best performance and is highlighted with gray backgrounds.

	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	69.21	71.06	73.50	73.58	72.24	71.02
	RKD	69.61	71.82	71.90	73.35	72.22	71.48
	CRD	71.16	73.48	75.51	75.48	74.14	73.94
	OFD	70.98	73.23	74.95	75.24	74.33	73.95
	ReviewKD	71.89	73.89	75.63	76.12	75.09	74.84
	SimKD	71.02	73.89	78.04*	75.48	75.21	74.83
	CAT-KD	71.62	73.62	76.91	75.60	74.82	74.65
logits	KD	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
	DHKD	71.19	73.92	**76.54**	**76.36***	**75.25***	**74.84***
	DHKD + ReviewKD	**73.14**	**75.21**	**78.29**	**77.97**	**76.56**	**76.27**

Table 2: Results on the CIFAR-100 validation. Teachers and students are in different architectures. The best performance of a single method is highlighted with a star *, and the best logits-based performance is highlighted in bold. The combination of DHKD and ReviewKD always achieves the best performance and is highlighted with gray backgrounds.

	Teacher	resnet32×4 79.42	WRN-40-2 75.61	VGG13 74.64	ResNet-50 79.34	resnet32×4 79.42
	Student	ShuffleNet-V1 70.50	ShuffleNet-V1 70.50	MBN-V2 64.60	MBN-V2 64.60	ShuffleNet-V2 71.82
features	FitNet	73.59	73.73	64.14	63.16	73.54
	RKD	72.28	72.21	64.52	64.43	73.21
	CRD	75.11	76.05	69.73	69.11	75.65
	OFD	75.98	75.85	69.48	69.04	76.82
	ReviewKD	77.45	77.14	70.37*	69.89	77.78
	SimKD	77.18	77.23	69.45	71.12	78.39
	CAT-KD	78.26*	77.35*	69.13	71.36*	78.41*
logits	KD	74.07	74.83	67.37	67.35	74.45
	DKD	76.45	76.70	69.71	70.35	77.07
	DHKD	**76.78**	**77.25**	**70.09**	**71.08**	**77.99**
	DHKD + ReviewKD	**78.35**	**78.22**	**71.01**	**71.51**	**78.83**

Table 3: Top-1 and top-5 accuracy (%) on the ImageNet validation. We set ResNet-34 as the teacher and ResNet-18 as the student.

distillation manner			features						logits			
Metric	Teacher	Student	AT	OFD	CRD	ReviewKD	SimKD	CAT-KD	KD	DKD	DIST	DHKD
top-1	73.31	69.75	70.69	70.81	71.17	71.61	71.59	71.26	70.66	71.70	72.07	**72.15**
top-5	91.42	89.07	90.01	89.98	90.13	90.51	90.48	90.45	89.88	90.41	90.42	**90.89**

for visualizing high-dimensional data. We use t-SNE to visualize the features (the outputs of the models’ backbones) learned by different models: vanilla student without distillation, models trained by KD, DHKD, and the teacher model.

Table 4: Top-1 and top-5 accuracy (%) on the ImageNet validation. We set ResNet-50 as the teacher and MobileNet as the student.

distillation manner			features						logits			
Metric	Teacher	Student	AT	OFD	CRD	ReviewKD	SimKD	CAT-KD	KD	DKD	DIST	DHKD
top-1	76.16	68.87	69.56	71.25	71.37	72.56	72.25	72.24	68.58	72.05	**73.24**	72.99
top-5	92.86	88.76	89.33	90.34	90.41	91.00	90.86	91.13	88.98	91.05	91.12	**91.45**

Table 5: Ablation studies on CIFAR-100. We choose one pair of models with the same architecture and another pair with different architectures. For the former pair, we set resnet32×4 as the teacher and resnet8×4 as the student; for the latter pair, we set ResNet-50 as the teacher and MobileNet-V2 as the student.

CE	BinaryKL -*Norm*	Dual Head	Gradient Alignment	Nonlinear Auxiliary Head	resnet32×4 resnet8×4	ResNet-50 MBN-V2
✓					72.50	64.60
	✓				75.15	68.52
✓	✓				N.A.	N.A.
✓	✓	✓			76.16	70.16
✓	✓	✓	✓		**76.54**	69.97
✓	✓	✓	✓	✓	74.38	**71.08**

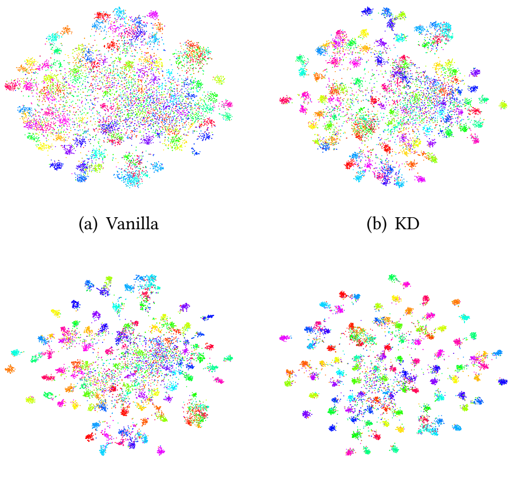

(a) Vanilla (b) KD
(c) DHKD (d) Teacher

Figure 5: t-SNE visualization of features learned by different methods. We do the visualization on the test set of CIFAR-100. We set resnet32×4 as the teacher and resnet8×4 as the student.

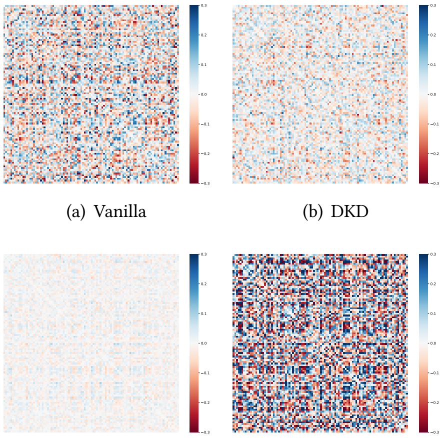

(a) Vanilla (b) DKD
(c) DHKD Aux (d) DHKD

Figure 6: The difference between the correlation matrices of the teacher's and student's logits. "DHKD Aux" is short for DHKD's auxiliary head. The test set is CIFAR-100, with resnet32×4 as the teacher and resnet8×4 as the student. The performance sorting of four classifier heads is DHKD (76.54) > DKD (76.37) > DHKD's Aux (76.20) > Vanilla (72.17).

The t-SNE results show that the features of KD are more separable than the vanilla student model, and the features of DHKD are more separable than KD but are less than the ones of the teacher model. We attribute this to the restriction of the model capacity. The t-SNE results prove that DHKD improves the discriminability of deep features.

4.4 Difference of Correlation Matrices

We compute the differences between the correlation matrices of the teacher's and student's logits for three different students: the vanilla student without distillation and the students trained by DKD and DHKD. Notably, the student model trained by DHKD has two classifier heads, thus having two sets of logits. We do the visualization for both heads. It can be found that the auxiliary head of DHKD captures the most correlation structure in the logits, as shown by the smallest differences between the teacher and the student. However, the original head of DHKD shows the biggest differences between the teacher and the student, even bigger than the vanilla student model. We attribute this phenomenon to the fact that we only align the output of the auxiliary classifier with the teacher in DHKD and the logits output by the original classifier is never aligned with the teacher.

5 Conclusion

This paper studied the problem of knowledge distillation. We provided the attempt to combine the BinaryKL loss with the CE loss, while our empirical findings indicated that merging the two losses results in degraded performance, even falling below the performance of using either loss independently. Inspired by previous research on neural collapse, we theoretically demonstrated that while the BinaryKL loss improves the efficacy of the backbone, it conflicts with the CE loss at the level of the linear classifier, thereby leading the model to a sub-optimal situation. To address this issue, we proposed a novel method called Dual-Head Knowledge Distillation, which separates the linear classifier into two distinct parts, each responsible for a specific loss. Experimental results on benchmark datasets confirmed the effectiveness of our proposed method.

6 Acknowledgement

The authors thank the anonymous reviewers for their careful reading of our manuscript and their many insightful comments and suggestions. This research is supported by the National Research Foundation, Singapore under its Industry Alignment Fund – Pre-positioning (IAF-PP) Funding Initiative. Any opinions, findings and conclusions or recommendations expressed in this material are those of the author(s) and do not reflect the views of National Research Foundation, Singapore. This research is also supported by the Joint NTU-WeBank Research Centre on Fintech, Nanyang Technological University, Singapore. Sheng-Jun Huang is supported by the NSFC(U2441285, 62222605).

References

[1] Rich Caruana. 1997. Multitask Learning. *MLJ* (1997).

[2] Defang Chen, Jian-Ping Mei, Hailin Zhang, Can Wang, Yan Feng, and Chun Chen. 2022. Knowledge distillation with the reused teacher classifier. In *CVPR*.

[3] Pengguang Chen, Shu Liu, Hengshuang Zhao, and Jiaya Jia. 2021. Distilling Knowledge via Knowledge Review. In *CVPR*.

[4] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. 2009. ImageNet: A large-scale hierarchical image database. In *CVPR*.

[5] Cong Fang, Hangfeng He, Qi Long, and Weijie J Su. 2021. Exploring deep neural networks via layer-peeled model: Minority collapse in imbalanced training. *PNAS* (2021).

[6] Zheng Ge, Songtao Liu, Feng Wang, Zeming Li, and Jian Sun. 2021. YOLOX: Exceeding YOLO Series in 2021. *arXiv preprint arXiv:2107.08430* (2021).

[7] Ziyao Guo, Haonan Yan, Hui Li, and Xiaodong Lin. 2023. Class attention transfer based knowledge distillation. In *CVPR*.

[8] Gunshi Gupta, Karmesh Yadav, and Liam Paull. 2020. Look-ahead meta learning for continual learning. In *NeurIPS*.

[9] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016. Deep Residual Learning for Image Recognition. In *CVPR*.

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

[11] Geoffrey Hinton, Oriol Vinyals, Jeff Dean, et al. 2015. Distilling the Knowledge in a Neural Network. *arXiv preprint arXiv:1503.02531* (2015).

[12] Tao Huang, Shan You, Fei Wang, Chen Qian, and Chang Xu. 2022. Knowledge distillation from a stronger teacher. In *NeurIPS*.

[13] Wenlong Ji, Yiping Lu, Yiliang Zhang, Zhun Deng, and Weijie J Su. 2022. An unconstrained layer-peeled perspective on neural collapse. In *ICLR*.

[14] Alex Kendall, Yarin Gal, and Roberto Cipolla. 2018. Multi-task Learning Using Uncertainty to Weigh Losses for Scene Geometry and Semantics. In *CVPR*.

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

[16] Taehyeon Kim, Jaehoon Oh, NakYil Kim, Sangwook Cho, and Se-Young Yun. 2021. Comparing kullback-leibler divergence and mean squared error loss in knowledge distillation. In *IJCAI*.

[17] Animesh Koratana, Daniel Kang, Peter Bailis, and Matei Zaharia. 2019. Lit: Learned intermediate representation training for model compression. In *ICML*.

[18] Alex Krizhevsky and Geoffrey Hinton. 2009. Learning multiple layers of features from tiny images. (2009).

[19] Lujun Li. 2022. Self-regulated feature learning via teacher-free feature distillation. In *ECCV*.

[20] Tsung-Yi Lin, Priya Goyal, Ross Girshick, Kaiming He, and Piotr Dollár. 2017. Focal Loss for Dense Object Detection. In *ICCV*.

[21] Xiaolong Liu, Lujun Li, Chao Li, and Anbang Yao. 2023. NORM: Knowledge Distillation via N-to-One Representation Matching. In *ICLR*.

[22] Jianfeng Lu and Stefan Steinerberger. 2022. Neural collapse under cross-entropy loss. *Applied and Computational Harmonic Analysis* (2022).

[23] Ningning Ma, Xiangyu Zhang, Hai-Tao Zheng, and Jian Sun. 2018. Shufflenet v2: Practical guidelines for efficient cnn architecture design. In *ECCV*.

[24] Roy Miles and Krystian Mikolajczyk. 2024. Understanding the role of the projector in knowledge distillation. In *AAAI*.

[25] Vardan Papyan, XY Han, and David L Donoho. 2020. Prevalence of neural collapse during the terminal phase of deep learning training. *PNAS* (2020).

[26] Wonpyo Park, Dongju Kim, Yan Lu, and Minsu Cho. 2019. Relational Knowledge Distillation. In *CVPR*.

[27] Nikolaos Passalis and Anastasios Tefas. 2018. Learning Deep Representations with Probabilistic Knowledge Transfer. In *ECCV*.

[28] Alexandre Rame, Corentin Dancette, and Matthieu Cord. 2022. Fishr: Invariant Gradient Variances for Out-of-Distribution Generalization. In *ICML*.

[29] Adriana Romero, Nicolas Ballas, Samira Ebrahimi Kahou, Antoine Chassang, Carlo Gatta, and Yoshua Bengio. 2015. Fitnets: Hints for Thin Deep Nets. In *ICLR*.

[30] Ismail Elezi Roy Miles and Jiankang Deng. 2024. VKD: Improving Knowledge Distillation using Orthogonal Projections. In *CVPR*.

[31] Mark Sandler, Andrew Howard, Menglong Zhu, Andrey Zhmoginov, and Liang-Chieh Chen. 2018. Mobilenetv2: Inverted residuals and linear bottlenecks. In *CVPR*.

[32] Karen Simonyan and Andrew Zisserman. 2015. Very deep convolutional networks for large-scale image recognition. In *ICLR*.

[33] Yonglong Tian, Dilip Krishnan, and Phillip Isola. 2019. Contrastive Representation Distillation. In *ICLR*.

[34] Laurens Van der Maaten and Geoffrey Hinton. 2008. Visualizing data using t-SNE. *JMLR* (2008).

[35] Yuzhu Wang, Lechao Cheng, Manni Duan, Yongheng Wang, Zunlei Feng, and Shu Kong. 2023. Improving knowledge distillation via regularizing feature norm and direction. *arXiv preprint arXiv:2305.17007* (2023).

[36] Stephan Wojtowytsch Weinan E. 2022. On the emergence of simplex symmetry in the final and penultimate layers of neural network classifiers. In *MSML*.

[37] Yue Wu, Yinpeng Chen, Lu Yuan, Zicheng Liu, Lijuan Wang, Hongzhi Li, and Yun Fu. 2020. Rethinking Classification and Localization for Object Detection. In *CVPR*.

[38] Penghui Yang, Ming-Kun Xie, Chen-Chen Zong, Lei Feng, Gang Niu, Masashi Sugiyama, and Sheng-Jun Huang. 2023. Multi-Label Knowledge Distillation. In *ICCV*.

[39] Yibo Yang, Shixiang Chen, Xiangtai Li, Liang Xie, Zhouchen Lin, and Dacheng Tao. 2022. Inducing Neural Collapse in Imbalanced Learning: Do We Really Need a Learnable Classifier at the End of Deep Neural Network?. In *NeurIPS*.

[40] Tianhe Yu, Saurabh Kumar, Abhishek Gupta, Sergey Levine, Karol Hausman, and Chelsea Finn. 2020. Gradient surgery for multi-task learning. In *NeurIPS*.

[41] Sergey Zagoruyko and Nikos Komodakis. 2016. Wide Residual Networks. In *BMVC*.

[42] Xiangyu Zhang, Xinyu Zhou, Mengxiao Lin, and Jian Sun. 2018. Shufflenet: An extremely efficient convolutional neural network for mobile devices. In *CVPR*.

[43] Ying Zhang, Tao Xiang, Timothy M Hospedales, and Huchuan Lu. 2018. Deep Mutual Learning. In *CVPR*.

[44] Borui Zhao, Quan Cui, Renjie Song, Yiyu Qiu, and Jiajun Liang. 2022. Decoupled Knowledge Distillation. In *CVPR*.

[45] Kaixiang Zheng and En-Hui Yang. 2024. Knowledge Distillation Based on Transformed Teacher Matching. In *ICLR*.

[46] Qingping Zheng, Jiankang Deng, Zheng Zhu, Ying Li, and Stefanos Zafeiriou. 2022. Decoupled Multi-task Learning with Cyclical Self-regulation for Face Parsing. In *CVPR*.

[47] Boyan Zhou, Quan Cui, Xiu-Shen Wei, and Zhao-Min Chen. 2020. BBN: Bilateral-branch network with cumulative learning for long-tailed visual recognition. In *CVPR*.

[48] Beier Zhu, Yulei Niu, Yucheng Han, Yue Wu, and Hanwang Zhang. 2023. Prompt-aligned gradient for prompt tuning. In *ICCV*.

[49] Zhihui Zhu, Tianyu Ding, Jinxin Zhou, Xiao Li, Chong You, Jeremias Sulam, and Qing Qu. 2021. A geometric analysis of neural collapse with unconstrained features. In *NeurIPS*.

A Dual Head is the Only Way

As mentioned in Section 3.1, we have tried many other methods before introducing the auxiliary head, including decreasing the balancing parameter α, reducing the learning rate, and using a gradient clipping method.

We cannot even give a table for decreasing the balancing parameter α because the training will collapse unless α is less than 0.1, leading to a result even worse than the vanilla KD.

More results about reducing the learning rate and using a gradient clipping method can be found in Table 6 and Table 7. Reducing the learning rate from 0.1 to $1e-2$, $5e-3$, $1e-3$, the student model still collapses during the training phase most of the time. The only student model that does not collapse has an accuracy of only 39.56%, far below the normally-trained model.

As for the gradient clipping method, even though we change the Maximum Gradient Norm Value (MGNV) in a small step size, the model still cannot achieve comparable performance with the model trained with only the CE loss.

B Supplementary Material about Neural Collapse

B.1 Detailed Definition of ETF and Neural Collapse

Definition 1 (Simplex Equiangular Tight Frame [25]). *A collection of vectors $\mathbf{m}_i \in \mathbb{R}^d$, $i = 1, 2, \cdots, K$, $d \geq K-1$, is said to be a simplex equiangular tight frame if:*

$$\mathbf{M} = \sqrt{\frac{K}{K-1}}\mathbf{U}\left(\mathbf{I}_K - \frac{1}{K}\mathbf{1}_K\mathbf{1}_K^\top\right), \tag{9}$$

where $\mathbf{M} = [\mathbf{m}_1, \cdots, \mathbf{m}_K] \in \mathbb{R}^{d\times K}$, $\mathbf{U} \in \mathbb{R}^{d\times K}$ allows a rotation and satisfies $\mathbf{U}^\top\mathbf{U} = \mathbf{I}_K$, $\mathbf{I}_K$ is the identity matrix, and $\mathbf{1}_K$ is an all-ones vector.

Theorem 1 (Papyan et al. [25]). *All vectors in a simplex Equiangular Tight Frame (ETF) have an equal ℓ_2 norm and the same pair-wise angle,* i.e.,

$$\mathbf{m}_i^\top\mathbf{m}_j = \frac{K}{K-1}\delta_{i,j} - \frac{1}{K-1}, \forall i, j \in [1, K], \tag{10}$$

where $\delta_{i,j}$ equals to 1 when $i = j$ and 0 otherwise. The pairwise angle $-\frac{1}{K-1}$ is the maximal equiangular separation of the K vectors in $\mathbb{R}^d$.

Then, according to [25], the neural collapse (NC) phenomenon can be formally described as:

(NC1) Within-class variability of the last-layer features collapse: $\Sigma_W \rightarrow \mathbf{0}$, and $\Sigma_W := \text{Avg}_{i,k}\{(\boldsymbol{h}_{k,i} - \boldsymbol{h}_k)(\boldsymbol{h}_{k,i} - \boldsymbol{h}_k)^\top\}$, where $\boldsymbol{h}_{k,i}$ is the last-layer feature of the i-th sample in the k-th class, and $\boldsymbol{h}_k = \text{Avg}_i\{\boldsymbol{h}_{k,i}\}$ is the within-class mean of the last-layer features in the k-th class;

(NC2) Convergence to a simplex ETF: $\tilde{\boldsymbol{h}}_k = (\boldsymbol{h}_k - \boldsymbol{h}_G)/||\boldsymbol{h}_k - \boldsymbol{h}_G||$, $k \in [1, K]$, satisfies Eq. (10), where $\boldsymbol{h}_G$ is the global mean of the last-layer features, *i.e.,* $\boldsymbol{h}_G = \text{Avg}_{i,k}\{\boldsymbol{h}_{k,i}\}$;

(NC3) Convergence to self duality: $\tilde{\boldsymbol{h}}_k = \boldsymbol{w}_k/||\boldsymbol{w}_k||$, where $\boldsymbol{w}_k$ is the classifier vector of the k-th class;

(NC4) Simplification to the nearest class center prediction: $\arg\max_k\langle \boldsymbol{h}, \boldsymbol{w}_k\rangle = \arg\min_k ||\boldsymbol{h} - \boldsymbol{h}_k||$, where $\boldsymbol{h}$ is the last-layer feature of a sample to predict for classification.

B.2 Proof of Propositions

B.2.1 Proof of Proposition 1. According to [39], the negative gradients of $\mathcal{L}_{CE}$ on this batch can be calculated as follows:

$$-\frac{\partial\mathcal{L}_{CE}}{\partial\boldsymbol{w}_k} = \underbrace{\sum_{i=1}^{n_k}\left(1 - p_k\left(\boldsymbol{h}_{k,i}^{\mathcal{S}}\right)\right)\boldsymbol{h}_{k,i}^{\mathcal{S}}}_{\boldsymbol{F}_{\boldsymbol{w}\text{-pull}}^{\text{CE}}} + \underbrace{\left(-\sum_{k'\neq k}^{K}\sum_{j=1}^{n_{k'}}p_k\left(\boldsymbol{h}_{k',j}^{\mathcal{S}}\right)\boldsymbol{h}_{k',j}^{\mathcal{S}}\right)}_{\boldsymbol{F}_{\boldsymbol{w}\text{-push}}^{\text{CE}}}, \tag{11}$$

where $p_k(\boldsymbol{h})$ is the predicted probability that $\boldsymbol{h}$ belongs to the k-th class. It is calculated by the softmax function and takes the following form in the CE loss:

$$p_k(\boldsymbol{h}) = \frac{\exp(\boldsymbol{h}^\top\boldsymbol{w}_k)}{\sum_{k'=1}^{K}\exp(\boldsymbol{h}^\top\boldsymbol{w}_{k'})},\ 1 \leq k \leq K. \tag{12}$$

Similarly, we can calculate the negative gradients of $\mathcal{L}_{\text{BinaryKL}}$ as follows:

$$-\frac{\partial\mathcal{L}_{\text{BinaryKL}}}{\partial\boldsymbol{w}_k} = \underbrace{\sum_{i=1}^{n_k}\tau(q_k(\boldsymbol{h}_{k,i}^{\mathcal{T}}) - q_k(\boldsymbol{h}_{k,i}^{\mathcal{S}}))\boldsymbol{h}_{k,i}^{\mathcal{S}}}_{\boldsymbol{F}_{\boldsymbol{w}\text{-pull}}^{\text{BinaryKL}}} + \tag{13}$$

$$\underbrace{\left(-\sum_{k'\neq k}^{K}\sum_{j=1}^{n_{k'}}\tau(q_k(\boldsymbol{h}_{k',j}^{\mathcal{S}}) - q_k(\boldsymbol{h}_{k',j}^{\mathcal{T}}))\boldsymbol{h}_{k',j}^{\mathcal{S}}\right)}_{\boldsymbol{F}_{\boldsymbol{w}\text{-push}}^{\text{BinaryKL}}}, \tag{14}$$

where $q_k(\boldsymbol{h})$ is the binary predicted probability that $\boldsymbol{h}$ has a positive label on the k-th class. It is calculated by the sigmoid function and takes the following form in the BinaryKL loss:

$$q_k(\boldsymbol{h}) = \frac{1}{1 + e^{-\boldsymbol{h}^\top\boldsymbol{w}_k/\tau}}. \tag{15}$$

Based upon Equation 11, 13, we can prove Proposition 1.

B.2.2 Proof of Proposition 2. The definitions of $p_k(\boldsymbol{h})$ and $q_k(\boldsymbol{h})$ are the same as Equation 12 and 15.

Then the negative gradients of $\mathcal{L}_{CE}$ *w.r.t.* the features on this batch can be calculated as follows:

$$-\frac{\partial\mathcal{L}_{CE}}{\partial\boldsymbol{h}} = \underbrace{(1 - p_c(\boldsymbol{h}^{\mathcal{S}}))\boldsymbol{w}_c^{\mathcal{S}}}_{\boldsymbol{F}_{\boldsymbol{h}\text{-pull}}^{\text{CE}}} + \underbrace{(-\sum_{k\neq c}^{K}p_k(\boldsymbol{h}^{\mathcal{S}})\boldsymbol{w}_k^{\mathcal{S}})}_{\boldsymbol{F}_{\boldsymbol{h}\text{-push}}^{\text{CE}}}. \tag{16}$$

The negative gradients of $\mathcal{L}_{\text{BinaryKL}}$ can be calculated as follows:

$$-\frac{\partial\mathcal{L}_{\text{BinaryKL}}}{\partial\boldsymbol{w}_k} = \underbrace{\tau(q_c(\boldsymbol{h}^{\mathcal{T}}) - q_c(\boldsymbol{h}^{\mathcal{S}}))\boldsymbol{w}_c^{\mathcal{S}}}_{\boldsymbol{F}_{\boldsymbol{h}\text{-pull}}^{\text{BinaryKL}}} + \tag{17}$$

$$\underbrace{\left(-\tau\sum_{k\neq c}^{K}(p_k(\boldsymbol{h}^{\mathcal{S}}) - p_k(\boldsymbol{h}^{\mathcal{T}}))\boldsymbol{w}_k^{\mathcal{S}}\right)}_{\boldsymbol{F}_{\boldsymbol{h}\text{-push}}^{\text{BinaryKL}}}. \tag{18}$$

Based upon Equation 16, 17, we can prove Proposition 2.

Table 6: Some attempts that change the learning rate. Top-1 accuracy on CIFAR-100 is given in the table. For elements that are "N.A.", we give the ordinal number of the epoch when the models collapse during the training phase. It can be observed that almost all the student models collapse. As the learning rate decreases, the time for model collapse is delayed.

	Same Architecture			Different Architectures		
teacher	resnet56	resnet110	resnet32×4	resnet32×4	VGG13	ResNet-50
student	resnet20	resnet32	resnet8×4	ShuffleNet-V1	MBN-V2	MBN-V2
lr=1e-2	N.A.(8)	N.A.(6)	N.A.(5)	N.A.(8)	N.A.(44)	N.A.(34)
lr=5e-3	N.A.(8)	N.A.(8)	N.A.(11)	N.A.(16)	N.A.(48)	N.A.(69)
lr=1e-3	N.A.(35)	N.A.(29)	N.A.(27)	N.A.(50)	N.A.(126)	39.56

Table 7: Some attempts using gradient clipping (GC) method. We change the Maximum Gradient Norm Value (MGNV) in a small step size, and the student model still cannot achieve comparable performance with the model trained with only the CE loss. We set resnet56 as the teacher and resnet20 as the student. All the experiments are done on CIFAR-100.

MGNV	0.01	0.1	0.3	0.5	0.7	1.0	Pure CE No GC
Test Accuracy	1.95	23.63	59.11	65.16	N.A.	N.A.	69.06

Table 8: The parameter sensitivity analysis on CIFAR-100. We set resnet32×4 as the teacher and resnet8×4 as the student.

α	0.2	0.5	1	2	5
Top-1 Accuracy	75.66	75.85	76.24	76.54	67.56

Table 9: Comparison between DHKD and "dual head + vanilla KD".

Teacher	Student	without distillation	vanilla KD	dual head + vanilla KD	DHKD
WRN-40-2	WRN-16-2	73.26	74.92	75.24	76.36
ResNet50	MBN-V2	64.60	67.35	68.76	71.08

B.3 Propositions for the Modified Loss

We can rewrite the $\mathcal{L}_{\text{overall}}$ in the following form:

$$\mathcal{L}_{\text{overall}} = \mathcal{L}_{CE} + \alpha\mathcal{L}_{\text{BinaryKL-Norm}}, \tag{19}$$

where

$$\mathcal{L}_{\text{BinaryKL-Norm}} = \tau^2 \sum\nolimits_{i=1}^{B} \sum\nolimits_{k=1}^{K} \mathcal{KL}\Big(\Big[\frac{1}{2}, \frac{1}{2}\Big] \Bigg\| \Bigg[\sigma\left(\frac{z^{\mathcal{S}}_{i,k} - z^{\mathcal{T}}_{i,k}}{\tau}\right), 1 - \sigma\left(\frac{z^{\mathcal{S}}_{i,k} - z^{\mathcal{T}}_{i,k}}{\tau}\right)\Bigg]\Big), \tag{20}$$

as defined in Equation 7.

Define an auxiliary function $w(\cdot,\cdot)$ as below:

$$w_k(\boldsymbol{h}_1, \boldsymbol{h}_2) = \frac{1}{1 + e^{-(\boldsymbol{h}_1^\top \boldsymbol{w}_k - \boldsymbol{h}_2^\top \boldsymbol{w}_k)/\tau}}. \tag{21}$$

Before proving Proposition 3, we need to prove a lemma:

Lemma 1. *For all $k' \in \{1, \cdots, K\}$ and $j \in \{1, \cdots, n_{k'}\}$,*

$$\Big[(\frac{1}{2} - w_k(\boldsymbol{h}^{\mathcal{S}}_{k',j}, \boldsymbol{h}^{\mathcal{T}}_{k',j}))(\boldsymbol{h}^{\mathcal{S}}_{k',j} - \boldsymbol{h}^{\mathcal{T}}_{k',j})\Big]^\top \boldsymbol{w}_k \leq 0. \tag{22}$$

Proof. If $(\boldsymbol{h}^{\mathcal{S}}_{k',j} - \boldsymbol{h}^{\mathcal{T}}_{k',j})^\top \boldsymbol{w}_k > 0$, then $w_k(\boldsymbol{h}^{\mathcal{S}}_{k',j}, \boldsymbol{h}^{\mathcal{T}}_{k',j}) > \frac{1}{2}$. If $(\boldsymbol{h}^{\mathcal{S}}_{k',j} - \boldsymbol{h}^{\mathcal{T}}_{k',j})^\top \boldsymbol{w}_k \leq 0$, then $w_k(\boldsymbol{h}^{\mathcal{S}}_{k',j}, \boldsymbol{h}^{\mathcal{T}}_{k',j}) \leq \frac{1}{2}$. So, in both conditions, we have the following inequality:

$$\begin{aligned} &\left[\left(\frac{1}{2} - w_k(\boldsymbol{h}^{\mathcal{S}}_{k',j}, \boldsymbol{h}^{\mathcal{T}}_{k',j})\right)\left(\boldsymbol{h}^{\mathcal{S}}_{k',j} - \boldsymbol{h}^{\mathcal{T}}_{k',j}\right)\right]^\top \boldsymbol{w}_k \\ &\quad = (\frac{1}{2} - w_k(\boldsymbol{h}^{\mathcal{S}}_{k',j}, \boldsymbol{h}^{\mathcal{T}}_{k',j}))\Big[(\boldsymbol{h}^{\mathcal{S}}_{k',j} - \boldsymbol{h}^{\mathcal{T}}_{k',j})^\top \boldsymbol{w}_k\Big] \leq 0. \end{aligned} \tag{23}$$

Then, we can get the propositions for the modified loss.

Proposition 3. *The gradients of $\mathcal{L}_{\text{overall}}$* w.r.t. *the linear classifier can be formulated as follows:*

$$\frac{\partial \mathcal{L}_{\text{overall}}}{\partial \boldsymbol{w}_k} = -(\boldsymbol{F}^{\text{CE}}_{\boldsymbol{w}\text{-pull}} + \boldsymbol{F}^{\text{CE}}_{\boldsymbol{w}\text{-push}}) - \alpha \boldsymbol{F}^{\text{BinaryKL-Norm}}_{\boldsymbol{w}-\text{obstacle}}, \tag{24}$$

where

$$\boldsymbol{F}^{\text{CE}}_{\boldsymbol{w}\text{-pull}} = \sum_{i=1}^{n_k}(1 - p_k(\boldsymbol{h}^{\mathcal{S}}_{k,i}))\boldsymbol{h}^{\mathcal{S}}_{k,i}, \boldsymbol{F}^{\text{CE}}_{\boldsymbol{w}\text{-push}} = -\sum_{k' \neq k}^{K}\sum_{j=1}^{n_{k'}} p_k(\boldsymbol{h}^{\mathcal{S}}_{k',i})\boldsymbol{h}^{\mathcal{S}}_{k',j},$$

$$\boldsymbol{F}^{\text{BinaryKL-Norm}}_{\boldsymbol{w}-\text{obstacle}} = \tau\sum_{k'=1}^{K}\sum_{j=1}^{n_{k'}}(\frac{1}{2} - w_k(\boldsymbol{h}^{\mathcal{S}}_{k',j}, \boldsymbol{h}^{\mathcal{T}}_{k',j}))(\boldsymbol{h}^{\mathcal{S}}_{k',j} - \boldsymbol{h}^{\mathcal{T}}_{k',j}). \tag{25}$$

Proof. The definition of $p_k(\boldsymbol{h})$ is the same as Equation 12. The negative gradients of $\mathcal{L}_{\text{BinaryKL-Norm}}$ on this batch can be calculated as follows:

$$-\frac{\partial\mathcal{L}_{\text{BinaryKL-Norm}}}{\partial\boldsymbol{w}_k} = \sum_{k'=1}^{K}\sum_{j=1}^{n_{k'}}\tau(\frac{1}{2} - w_k(\boldsymbol{h}^{\mathcal{S}}_{k',j}, \boldsymbol{h}^{\mathcal{T}}_{k',j}))(\boldsymbol{h}^{\mathcal{S}}_{k',j} - \boldsymbol{h}^{\mathcal{T}}_{k',j}) \tag{26}$$

According to Lemma 1, each term above has an opposite direction with $\boldsymbol{w}_k$, which would obstruct the learning of the linear classifier. Consequently, we can denote it as $\boldsymbol{F}^{\text{BinaryKL-Norm}}_{\boldsymbol{w}-\text{obstacle}}$.

Based upon Equation 11, 26, we can prove Proposition 3.

Remark The form of Proposition 3 differs a lot from Proposition 1, but Proposition 3 shows that the BinaryKL-Norm loss has stronger negative effects over the linear classifier the BinaryKL loss: it hinders the training of the linear classifier all the time without deploy any positive effect.

Proposition 4. *The gradients of* $\mathcal{L}_{\text{overall}}$ w.r.t. *the features can be formulated as follows:*

$$\frac{\partial\mathcal{L}_{\text{overall}}}{\partial\boldsymbol{h}} = -(\boldsymbol{F}^{\text{CE}}_{\boldsymbol{h}\text{-pull}} + \alpha\boldsymbol{F}^{\text{BinaryKL-Norm}}_{\boldsymbol{h}\text{-pull}}) - (\boldsymbol{F}^{\text{CE}}_{\boldsymbol{h}\text{-push}} + \alpha\boldsymbol{F}^{\text{BinaryKL-Norm}}_{\boldsymbol{h}\text{-push}}), \tag{27}$$

where

$$\begin{aligned}
\boldsymbol{F}^{\text{CE}}_{\boldsymbol{h}\text{-pull}} &= (1 - p_c(\boldsymbol{h}^{\mathcal{S}}))\boldsymbol{w}^{\mathcal{S}}_c, \\
\boldsymbol{F}^{\text{BinaryKL-Norm}}_{\boldsymbol{h}\text{-pull}} &= \tau(\frac{1}{2} - w_c(\boldsymbol{h}^{\mathcal{S}}, \boldsymbol{h}^{\mathcal{T}}))\boldsymbol{w}^{\mathcal{S}}_c, \\
\boldsymbol{F}^{\text{CE}}_{\boldsymbol{h}\text{-push}} &= -\sum\nolimits_{k\neq c}^{K} p_k(\boldsymbol{h}^{\mathcal{S}})\boldsymbol{w}^{\mathcal{S}}_k, \\
\boldsymbol{F}^{\text{BinaryKL-Norm}}_{\boldsymbol{h}\text{-push}} &= \tau\sum\nolimits_{k\neq c}^{K}(w_k(\boldsymbol{h}^{\mathcal{S}}, \boldsymbol{h}^{\mathcal{T}}) - \frac{1}{2})\boldsymbol{w}^{\mathcal{S}}_k.
\end{aligned} \tag{28}$$

Proof. The definition of $p_k(\boldsymbol{h})$ is the same as Equation 12. The negative gradients of $\mathcal{L}_{\text{BinaryKL-Norm}}$ can be calculated as follows:

$$-\frac{\partial\mathcal{L}_{\text{BinaryKL-Norm}}}{\partial\boldsymbol{w}_k} = \underbrace{\tau(\frac{1}{2} - w_c(\boldsymbol{h}^{\mathcal{S}}, \boldsymbol{h}^{\mathcal{T}}))\boldsymbol{w}^{\mathcal{S}}_c}_{\boldsymbol{F}^{\text{BinaryKL-Norm}}_{\boldsymbol{h}\text{-pull}}} + \underbrace{\left(-\tau\sum\nolimits_{k\neq c}^{K}(w_k(\boldsymbol{h}^{\mathcal{S}}, \boldsymbol{h}^{\mathcal{T}}) - \frac{1}{2})\boldsymbol{w}^{\mathcal{S}}_k\right)}_{\boldsymbol{F}^{\text{BinaryKL-Norm}}_{\boldsymbol{h}\text{-push}}}. \tag{29}$$

Based upon Equation 16, 29, we can prove Proposition 4.

C Comparing Methods

Comparing methods are FitNet [29], RKD [26], CRD [33], OFD [10], ReviewKD [3], DKD [44], SimKD [2] CAT-KD [7], and DIST [12].

D Implementation Details

We perform experiments on two benchmark datasets: CIFAR-100 [18] and ImageNet [4]. CIFAR-100 covers 100 categories. It contains 50,000 images in the train set and 10,000 images in the test set. ImageNet covers 1,000 categories of images. It contains 1.28 million images in the train set and 50,000 images in the test set.

CIFAR-100: Teachers and students are trained for 240 epochs with SGD, and the batch size is 64. The learning rates are 0.01 for ShuffleNet [23, 42] and MobileNet-V2 [31], and 0.05 for the other series (*e.g.* VGG[32], ResNet [9] and WRN [41]). The learning rate is divided by 10 at the 150th, 180th, and 210th epochs. The weight decay and the momentum are set to 5e-4 and 0.9. The weight for the CE loss is set to 1.0, and the temperature is set to 2 for all experiments. α is set as 0.1 for (resnet56⟶resnet20, resnet110⟶ resnet32) and is chosen from $\{0.5, 1, 2\}$ for all other experiments. We choose a linear classifier as the auxiliary classifier for students having the same architectures as their teachers. A one-hidden-layer MLP with 200 hidden neurons is deployed as the auxiliary classifier for students with different architectures from their teachers. **We do not use the gradient alignment method when combining DHKD with ReviewKD** because it would seriously slow down the training speed, and we can still achieve the SOTA performance without it. All the experiments on CIFAR-100 are conducted on GeForce RTX 3090 GPUs.

ImageNet: Our implementation for ImageNet follows the standard practice. We train the models for 100 epochs. The batch size is 256, and the learning rate is initialized to 0.1 and divided by 10 for every 30 epochs. Weight decay is 1e-4, and the weight for the CE loss is set to 1.0. We set the temperature as 2 and α as 0.1 for all experiments. We only use the BinaryKL loss in the first 50 epochs of the training phase, which means that we set $\alpha = 0$ for the last 50 epochs. **We do not use the gradient alignment method on ImageNet** because it would seriously slow down the training speed, and we can still achieve the SOTA performance without it. Strictly following [44], for distilling networks of the same architecture, the teacher is ResNet-34 model, the student is ResNet-18; for different series, the teacher is ResNet-50 model, the student is MobileNet-V1. All the experiments on ImageNet are conducted on H100 PCIe GPUs.

E Parameter Sensitivity Analysis

We study the influence of hyper-parameters in this section. In all of our experiments, the temperature parameter τ is fixed at 2. Therefore, the only adjustable hyperparameter is α. Table 8 illustrates the performance of DHKD as the value of α changes among 0.2, 0.5, 1, 2, 5 on CIFAR-100. From the table, it can be observed that the performance of DHKD is not very sensitive to the parameter α.

F The Performance of "dual head + vanilla KD"

In this section, we conduct the experiments of comparing DHKD and "dual head + vanilla KD". The results are shown in Table 9, illustrating that our dual-head strategy with KL loss on the predicted probability provides only a slight improvement. As a result, only adding a new classifier cannot bring the improvement as much as our method, which does not undermine our contribution.