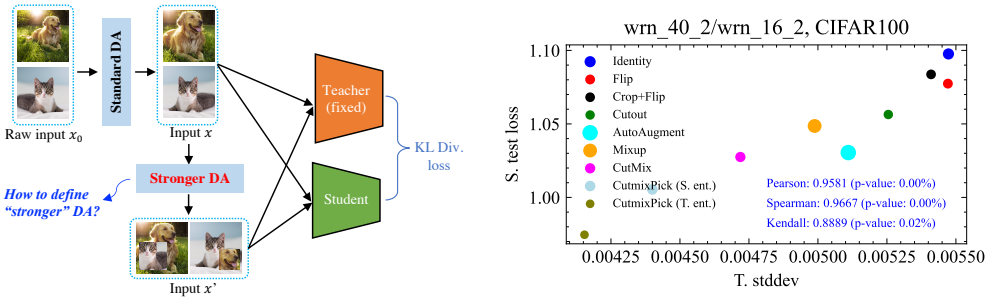
What Makes a "Good" Data Augmentation in Knowledge Distillation – A Statistical Perspective (with Appendix)

Huan Wang[1,2,†] **Suhas Lohit**[2,*] **Mike Jones**[2] **Yun Fu**[1]
[1]Northeastern University, Boston, MA [2]MERL, Cambridge, MA
Project: http://huanwang.tech/Good-DA-in-KD



(a) Apply additional stronger DA in KD (b) S. test loss *vs.* T. stddev with different DA schemes

Figure 1: **(a)** Illustration of applying a stronger data augmentation (DA) in addition to the standard DA (random crop and flip) in knowledge distillation (KD). We ask: *What makes a "good" DA when it is applied to KD in the manner of (a)?* **(b)** We present a proven proposition (Proposition 3.1) to answer this question rigorously, along with a practical metric to evaluate the "goodness" of a DA. The proposed metric is called *stddev of teacher's mean probability* (shorted as T. stddev). As seen in (b), there is a *strong* positive correlation (p-value < 5% is typically considered statistically significant) between the student's test loss (S. test loss) and T. stddev, showing that T. stddev well captures the "goodness" of different DA schemes in KD. The most striking fact from this plot may be: T. stddev is *purely* calculated with the teacher (no any student used) while it can "predict" the relative order of the *student*'s performance, implying the "goodness" of DA in KD probably is student-invariant.

Abstract

Knowledge distillation (KD) is a general neural network training approach that uses a teacher model to guide the student model. Existing works mainly study KD from the network output side (*e.g.*, trying to design a better KD loss function), while few have attempted to understand it from the input side. Especially, its interplay with data augmentation (DA) has not been well understood. In this paper, we ask: Why do some DA schemes (*e.g.*, CutMix) inherently perform much better than others in KD? What makes a "good" DA in KD? Our investigation from a statistical perspective suggests that **a good DA scheme should reduce the covariance of the teacher-student cross-entropy**. A practical metric, *the stddev of teacher's mean probability* (T. stddev), is further presented and well justified empirically. Besides the theoretical understanding, we also introduce a new entropy-based data-mixing DA scheme, *CutMixPick*, to further enhance CutMix. Extensive empirical studies

[†]This paper originates from Huan's summer internship work at MERL.
[*]Corresponding author: slohit@merl.com

36th Conference on Neural Information Processing Systems (NeurIPS 2022).

support our claims and demonstrate how we can harvest considerable performance gains simply by using a better DA scheme in knowledge distillation.

1 Introduction

Deep neural networks (DNNs) are the de facto methodology in many artificial intelligence areas nowadays [25, 37]. How to effectively train a deep network has been a central topic for decades. In the past several years, efforts have mainly focused on better architecture design (*e.g.*, batch normalization [20], residual blocks [14], dense connections [19]) and better loss functions (*e.g.*, label smoothing [43, 30], contrastive loss [18], large-margin softmax [26]) than the standard cross-entropy (CE) loss. Knowledge distillation (KD) [17] is a training method that falls into the second group. In KD, a stronger network – called teacher – is introduced to guide the learning of the original network – called student – by minimizing the discrepancy between the representations of the two networks,

$$\mathcal{L}_{KD} = (1-\alpha)\mathcal{L}_{CE}(y, \mathbf{p}^{(s)}) + \alpha\tau^2\mathcal{D}_{KL}(\mathbf{p}^{(t)}/\tau, \mathbf{p}^{(s)}/\tau), \tag{1}$$

where $\mathcal{D}_{KL}$ represents KL divergence [24]; $\alpha \in (0, 1)$ is a factor to balance the two loss terms; $\mathcal{L}_{CE}$ denotes the cross-entropy loss; y is the one-hot label and $\mathbf{p}^{(t)}, \mathbf{p}^{(s)}$ stand for the teacher's and student's output probabilities over the classes; τ is a temperature constant [17] to smooth predicted probabilities. KD allows us to train smaller, more efficient neural networks without compromising on accuracy, which facilitates deploying deep learning in resource constrained environments (*e.g.*, on mobile devices). KD has found plenty of applications in many tasks [6, 49, 13, 21, 50].

Most existing KD methods have attempted to improve it by proposing better KD loss functions applied at the network outputs [34, 31, 45]. Few works have considered KD from *the input side*. Especially, the interplay between KD and data augmentation (DA) [40] has not been well understood so far (note, by DA here, we mean the conventional DA concept: generating a new input by *transforming one or multiple inputs*. A broader scope of DA may involve the *neural network*, *e.g.*, dropout can be seen as a kind of DA [3]. We do not consider this type of DA in this paper due to the limited length).

In this work, we ask: *What makes a "good" data augmentation in knowledge distillation*? A clear answer to this question has many benefits. First, theoretically, it can help us towards a better understanding about how data augmentation plays a role in KD. Second, practically, it can bring us considerable performance gain – in Fig. 2, we show test error rates using the standard CE loss (no teacher) *vs.* using KD loss (with a teacher). As seen, a stronger DA can lower the test error rate and admit more training iterations without overfitting in KD. It is pretty obvious to see that "Flip+Crop" is stronger than "Flip" alone in Fig. 2. However, for other DA schemes, such as Mixup [54] *vs.* AutoAugment [8], which one is stronger? Not very clear. We thus desire a principled way (*e.g.*, a concrete metric) to make the vague concept "stronger" exact. Presenting such a theoretically sound metric and empirically validating its effectiveness is the goal of this paper.

Intuitively, a good DA should enrich the input data and expose more knowledge of the teacher so that the student can generalize better. We formalize this idea from a statistical learning perspective. Specifically, we will show a good DA scheme is defined by a lower variance (or covariance) of the *teacher's mean output probability* over different input samples, which ultimately leads to a lower generalization gap for the student. The proposed theory is well justified by our extensive empirical studies on CIFAR100 and Tiny ImageNet datasets with various pairs. The proposed theory well explains why CutMix is better than other alternatives (such as Mixup [54], AutoAugment [8]) in KD.

In addition to the new theoretical results, we also propose an entropy-based data picking scheme to select more informative samples for KD, which can deliver even lower variance of the

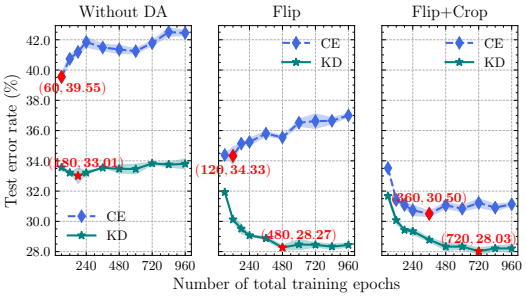


Figure 2: Test error rate of resnet20 on CIFAR100 when trained for different numbers of epochs with (KD) and without (CE) knowledge distillation (the teacher is resnet56 for KD). Each result is obtained by averaging 3 random runs (shaded area indicates the stddev). "Flip": random horizontal flip; "Crop": random crop. The optimal number of training epochs and its test loss are highlighted in red.

teacher's mean probability as well as lower generalization error of the student.

We make the following contributions in this paper:

- We present a proven proposition (Proposition 3.1) that precisely answers what defines a better DA in KD: Given a fixed teacher, a better DA is the one that gives a lower variance of the teacher's mean probability.
- The proposition is well justified empirically on standard image classification datasets with many teacher-student pairs.
- An entropy-based data picking scheme is introduced to further reduce the variance of the teacher's mean probability, which can further advance CutMix, the prior state-of-the-art DA approach among those evaluated in this paper.
- Empirically, we show how the presented theory can benefit in practice – we can simply enhance the existing KD methods by using a stronger DA and prolonged training iterations.

2 Related Work

Knowledge Distillation (KD). The general idea of knowledge distillation is to guide the training of a student model through a (stronger) teacher model (or an ensemble of models). It was pioneered by Buciluǎ *et al.* [4] and later refined by Hinton *et al.* [17], who coined the term. Since its debut, KD has seen extensive application in vision and language tasks [6, 15, 49, 21, 50]. Many variants have been proposed regarding the central question in KD, that is, how to define the *knowledge* transferred from the teacher to the student. Examples of such knowledge definitions include feature distance [35], feature map attention [53], feature distribution [32], activation boundary [16], inter-sample distance relationship [31, 34, 27, 46], and mutual information [45]. Several works [30, 39, 5] investigate the connection between label smoothing and knowledge distillation. Another line of works (*e.g.*, [44]) attempts to understand KD more theoretically. Over the past several years, the progress has been made primarily for intermediate feature maps and network outputs (*i.e.*, through a better loss function). In contrast, our goal is to improve the KD performance at the *input end* with the help of data augmentation. We will show this path is as effective and also has much potential for future research.

Data Augmentation (DA). Deep neural networks are prone to overfitting, *i.e.*, building input-target mappings using undesirable or irrelevant features (like noise) in the data. Data augmentation is a prevailing technique to curb overfitting [40]. In classification tasks, data augmentation aims to explicitly provide data with *label-invariant transformations* (such as random crop, horizontal flip, color jittering, Cutout [12]) during training so that the model can learn representations robust to those nuisance factors. Recently, more advanced data augmentation methods were proposed, which not only transform the input, but also transform the target. For example, Mixup [54] linearly mixes two images with the labels mixed by the same linear interpolation. Manifold Mixup [48] is similar to Mixup but conducts the mix operation in the feature level instead of pixel level; CutMix [51] pastes a patch cut from an image onto another image with the label decided by the area ratio of the two parts. AutoAugment [8] is a strong DA method that finds an optimal augmentation policy from a large search space via reinforcement learning. When both the input and target are transformed simultaneously, the key is to maintain a *semantic correspondence* between the new input and new target. Unlike these methods, which focus on general classification using the cross-entropy loss, our work investigates the interplay between data augmentation and knowledge distillation loss and proposes new data augmentation specifically for knowledge distillation.

Some recent KD works also involve the utilization of DA in KD, such as [2, 9]. Especially, [2] also employs Mixup and prolonged training to enhance the student performance in KD, akin to ours. Yet it is worthwhile to note that our paper is *substantially different* from theirs, in that we are seeking the *theoretical* reason explaining how to define a better DA in KD to deliver better performance; these works mainly investigate in an empirical fashion, with no theoretical results presented. Meanwhile, our work is not limited to one specific DA (see Sec. 5). We target a *general theoretical* understanding which can apply to a broad scope of DA schemes (fortunately, as our experiments show, the proposed theory and the derived DA "goodness" measure indeed capture it). The fact that [2] utilizes Mixup [54] and prolonged training to deliver 82.8% top-1 accuracy with resnet50 [14] on ImageNet [11] can be a direct proof that the principle proposed in this work can bring us very promising practical benefits.

One recent work [10] also conducts empirical studies of the impact of DA on KD. They first apply DA (*e.g.*, Mixup/CutMix) to the teacher training then conduct the KD step as usual (no extra DA in this step). Our investigation is the *exact opposite* to their setup: We train the teacher as usual (no Mixup/CutMix), then in the KD step we employ a more advanced DA (*e.g.*, Mixup/CutMix). Interestingly, they conclude that the teacher trained with Mixup/CutMix *hurts* the student's generalization ability, while we consistently see student performance boost via a stronger DA. Another recent work [42] utilizes KD to re-label mixed samples in Mixup to fix the inaccurate labelling problem of Mixup. Their work shows that KD can be used to make Mixup more generally useful, which is orthogonal to the topic of this work.

The work of [29] presents a statistical perspective to understand how the softened probabilities in KD are better than the one-hot hard labels. Our work is inspired by their *Bayes teacher* notion. This said, our work is different from theirs in that we focus on explaining how data augmentation plays a role in KD and answer what characterizes a good DA, while they attempt to answer why the knowledge distillation loss is better than the standard cross-entropy loss.

3 Theoretical Investigation

3.1 Prerequisites: Multi-Class Classification with KD

Given a training set $S = \{(x_n, y_n)\}_{n=1}^N \sim \mathcal{D}^N$, where $\mathcal{D}$ is the joint distribution for input-output random variable pair (x, y), the goal in multi-class classification is to pin down a predictor $\mathbf{f} : \mathcal{X} \to \mathbb{R}^C$ from a hypothesis class $\mathcal{H}$, where $\mathcal{X}$ is the input space and C refers to the number of classes. The predictor $\mathbf{f}$ is supposed to minimize the *true risk*

$$R_{\mathcal{D}}(\mathbf{f}) \stackrel{\text{def}}{=} \mathop{\mathrm{E}}_{(x,y)\sim\mathcal{D}} \left[L(y, \mathbf{f}(x)) \right], \tag{2}$$

where L stands for the loss objective function (*e.g.*, cross-entropy); the subscript $\mathcal{D}$ of $R_{\mathcal{D}}$ is to emphasize that the true risk is defined on the data distribution. The true risk is approximated in practice on a separate test set.

For training, the predictor aims to minimize the *empirical risk* defined on the training sequence S:

$$R_S(\mathbf{f}) \stackrel{\text{def}}{=} -\frac{1}{N} \sum_{n=1}^{N} \mathbf{e}_{y_n}^\top \log(\mathbf{f}(x_n)), \tag{3}$$

where $\mathbf{e}_y \in \{0, 1\}^C$ is a one-hot vector indicating the label $y \in [C] = \{1, 2, \cdots, C\}$. The subscript S of R_S is to emphasize the empirical risk is defined on the finite sampled data points.

In the context of KD, the one-hot hard target vector $\mathbf{e}_y$ is replaced with a probability vector $\mathbf{p}^{(t)}(x) \in \mathbb{R}_+^C$, giving us the *empirical distilled risk* of $\mathbf{f}$:

$$\hat{R}_S(\mathbf{f}) \stackrel{\text{def}}{=} -\frac{1}{N} \sum_{n=1}^{N} \mathbf{p}^{(t)}(x_n)^\top \log(\mathbf{f}(x_n)), \tag{4}$$

where the super-script t indicates the fixed *teacher* model. The theory concerning why Eq. (4) is better than Eq. (3) has been established in [29]. Interested readers may refer to their paper for more details. Next, we look into how the data augmentation plays a role in KD.

3.2 What Makes a "Good" DA in KD?

Intuitively, a better DA should provide more information, *i.e.*, expose more knowledge of the teacher so that the student can absorb more and thus generalize better. We make this idea rigorous as follows.

Proposition 3.1. *Given a bounded loss function and a fixed teacher model with the empirical distilled risk defined in Eq. (4), for any predictor* $\mathbf{f}$*, consider two sampled sequences* $S_1 \in \mathcal{D}^N$ *and* $S_2 \in \mathcal{D}^N$*, they are made up of* N *elements sampled from the same distribution* $\mathcal{D}$*,* ***while not i.i.d.*** *(especially when data augmentation is employed). If the elements in* S_1 *present a larger correlation than those in* S_2*, then the student's generalization gap trained on* S_1 *will be greater than that trained on* S_2*:*

$$\mathop{\mathrm{E}}_{S_1\sim\mathcal{D}^N} \left[(\hat{R}_{S_1}(\mathbf{f}) - R_{\mathcal{D}}(\mathbf{f}))^2 \right] > \mathop{\mathrm{E}}_{S_2\sim\mathcal{D}^N} \left[(\hat{R}_{S_2}(\mathbf{f}) - R_{\mathcal{D}}(\mathbf{f}))^2 \right]. \tag{5}$$

Proof. Let $\Delta = \hat{R}_S(\mathbf{f}) - R_{\mathcal{D}}(\mathbf{f})$. By the definition of variance, $\mathrm{E}_S[\Delta^2] = \mathrm{Var}_S[\Delta] + (\mathrm{E}_S[\Delta])^2$. To make the notation clearer, we define $q(x_i) = -\mathbf{p}^{(t)}(x_i)^\top \log(\mathbf{f}(x_i))$.

(1) Since $R_{\mathcal{D}}(\mathbf{f})$ is a constant (albeit unknown),

$$\begin{aligned}\mathrm{E}_{S\sim\mathcal{D}^N}[\Delta] &= \mathrm{E}_S[\hat{R}_S(\mathbf{f})] + \text{Const} = \mathrm{E}_S[\frac{1}{N}\sum_{i=1}^{N} q(x_i)] + \text{Const} \\ &= \frac{1}{N}\sum_{i=1}^{N}\mathrm{E}_S[q(x_i)] + \text{Const} = \frac{1}{N}\sum_{i=1}^{N}\mathrm{E}_{x_i}[q(x_i)] + \text{Const} \\ &= \frac{1}{N}\cdot N\cdot \mathrm{E}_x[q(x)] + \text{Const} = \mathrm{E}_x[q(x)] + \text{Const},\end{aligned} \tag{6}$$

where the second last equation is because each element x_i in S is drawn from the same distribution $\mathcal{D}$. From the RHS of the last equation, we can clearly see that for S_1, S_2, $\mathrm{E}_S[\Delta]$ is the same.

(2) Then we consider $\mathrm{Var}_S[\Delta]$. Again, since $R_{\mathcal{D}}(\mathbf{f})$ is a constant, we only need to consider

$$\begin{aligned}\mathrm{Var}_S[\hat{R}_S(\mathbf{f})] &= \mathrm{Var}_S[\frac{1}{N}\sum_{i=1}^{N} q(x_i)] = \frac{1}{N^2}\mathrm{Cov}_S[\sum_{j=1}^{N} q(x_j), \sum_{k=1}^{N} q(x_k)] \\ &= \frac{1}{N^2}\big(\sum_{i=1}^{N}\mathrm{Var}_{x_i}[q(x_i)] + 2\sum_{1\le j<k\le N}\mathrm{Cov}_S[q(x_j), q(x_k)]\big) \\ &= \frac{1}{N^2}\big(N\cdot\mathrm{Var}_x[q(x)] + 2\sum_{1\le j<k\le N}\mathrm{Cov}_S[q(x_j), q(x_k)]\big) \\ &= \frac{1}{N}\mathrm{Var}_x[q(x)] + \frac{2}{N^2}\sum_{1\le j<k\le N}\mathrm{Cov}_S[q(x_j), q(x_k)],\end{aligned} \tag{7}$$

where $\mathrm{Cov}[\cdot,\cdot]$ stands for covariance. From the last item in Eq. (7), we can see that the covariance part is different for different sampled S's. If the samples in S_1 present a larger correlation than those in S_2, we will have $\mathrm{Var}_{S_1}[\Delta] > \mathrm{Var}_{S_2}[\Delta]$, which further leads to increased generalization gap for the student, *i.e.*, $\mathrm{E}_{S_1}[(R_{S_1}(\mathbf{f}) - R_{\mathcal{D}}(\mathbf{f}))^2] > \mathrm{E}_{S_2}[(R_{S_2}(\mathbf{f}) - R_{\mathcal{D}}(\mathbf{f}))^2]$. The proof is finished. □

Practical Use: Stddev of Teacher's Mean Probability (T. stddev). Note in Proposition 3.1, the predictor $\mathbf{f}$ (*i.e.*, the student model) can be any one (not necessarily a converged model). Each student would have an order of different DA schemes regarding which is better "in its opinion". Presumably, different students will lead to different such orders. For practical use, we must pick a certain student as oracle to conduct the evaluation. Here, we can play a trick that can make it rather simple to use our theory – *assume* there is a student that performs *exactly the same* as the teacher (this assumption is not unpractical, since one straightforward example is to use the teacher as student). Then we have

$$\begin{aligned}\mathrm{Cov}[q(x_j), q(x_k)] &= \mathrm{Cov}[\mathbf{p}^{(t)}(x_j)^\top \log(\mathbf{f}(x_j)), \mathbf{p}^{(t)}(x_k)^\top \log(\mathbf{f}(x_k))] \\ &= \mathrm{Cov}[\mathbf{p}^{(t)}(x_j)^\top \log(\mathbf{p}^{(t)}(x_j)), \mathbf{p}^{(t)}(x_k)^\top \log(\mathbf{p}^{(t)}(x_k)).\end{aligned} \tag{8}$$

In this case, we only need the covariance of the *teacher's* probability to measure the "goodness" of a certain DA technique, *no need for the student*. Despite not using any information of the student, the proposed metric turns out to correlate surprisingly well with the student's performance (see Fig. 4).

Based on Eq. (8), when using S_1 *vs.* S_2 as training sequence, the fundamental factor answering for the variance gap of $\mathrm{Var}_S[\Delta]$ boils down to the covariance in $\{\mathbf{p}^{(t)}(x_i)\}_{i=1}^N$. That is, a larger covariance in $\{\mathbf{p}^{(t)}(x_i)\}_{i=1}^N$ leads to a larger covariance in $\{\mathbf{p}^{(t)}(x_i)^\top \log(\mathbf{p}^{(t)}(x_i))\}_{i=1}^N$ in Eq. (8), which ultimately leads to a higher $\mathrm{Var}_S[\Delta]$ in Eq. (7).

The next step is to find a feasible way to estimate the covariance in $\{\mathbf{p}^{(t)}(x_i)\}_{i=1}^N$. Consider the *average* variable (denoted as $\mathbf{u}$ here) of several random variables $\{\mathbf{p}^{(t)}(x_k)\}_{k=1}^K$,

$$\mathbf{u} = \frac{1}{K}\sum_{x_k\in S^*}\mathbf{p}^{(t)}(x_k), \mathbf{u}\in\mathbb{R}_+^C. \tag{9}$$

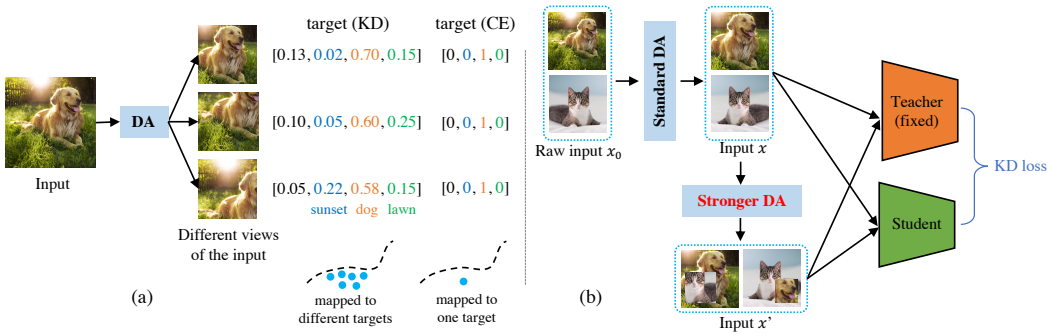


Figure 3: **(a)** Illustration of the difference of supervised target between the KD loss and cross-entropy (CE) loss. An input is transformed to different versions owing to data augmentation. KD loss can provide extra information to the student by mapping these views to different targets, while the CE loss cannot. This work attempts to answer what characterizes a "good" data augmentation scheme in distillation. **(b)** Illustration of adapting an existing DA approach to KD. The standard DA consists of random crop and horizontal flip. This training framework is employed to empirically verify our proposed metric to define a "stronger" data augmentation.

Its variance (or equivalently, stddev) inherently takes into account the covariance among its addends. Therefore, we can use the variance of $\mathbf{u}$ as a proxy for the covariance in $\{\mathbf{p}^{(t)}(x_k)\}_{k=1}^K$:

$$\mathbf{m} = \mathrm{Var}_{S^*}(\mathbf{u}), \mathbf{m} \in \mathbb{R}_+^C, \quad \bar{m} = \frac{1}{C}\sum_{i\in[C]} (\mathbf{m}_i)^{\frac{1}{2}}, \bar{m} \in \mathbb{R}_+, \tag{10}$$

where K is a pre-defined number of samples of set S^* to realize the averaging effect (*e.g.*, $K = 640$ in our experiments for CIFAR100 and Tiny ImageNet – see Appendix Sec. A.3 for a concrete calculation example). Note, $\mathbf{u}$ can be interpreted as the *teacher's mean probability* over K input samples.

If there is a large covariance among $\{\mathbf{p}^{(t)}(x_k)\}_{k=1}^K$, $\mathbf{m}$ will be large (in an element-wise sense). Then the average of $\mathbf{m}$ over classes, *i.e.*, the $\bar{m}$ (we do this averaging simply because we desire a *scalar* metric), is a good indicator to capture such covariance. *We thus formally introduce $\bar{m}$, the (averaged) T. stddev, as the proposed metric to measure the quality of a data augmentation scheme*. A lower $\bar{m}$ implies a better DA by our definition. In the experiments, we will show this metric defined purely using the *teacher* can accurately characterize the generalization error of the *students* after distillation.

4 Evaluated Algorithms

4.1 Extend Existing DA Approaches for KD

Given an existing DA approach, this section explains how we adapt it properly to the case of KD.

Specifically, let x_0 denote the raw data, x denote the transformed data by the standard augmentation (random crop and flip). Illustrated in Fig. 3(b), we will add the DA following x to obtain x'. Unlike the common data augmentation where *only* the transformed input is fed into the network, we keep *both the input x and x'* for the training (as such, the number of input examples is doubled). The consideration of keeping both inputs is to maintain the information path for the original input x so that we can easily see how the added information path of x' leads to a difference.

For x, its loss is still the original KD loss, consisting of the cross-entropy loss and the KL divergence (Eq. (1)). Of special note is that, for x', its loss is *only* the KL divergence, *i.e.*, *we do not use the labels assigned by the DA algorithm* (for example, in the original Mixup and CutMix, they assign a linearly interpolated label to the augmented sample) because these labels can actually be *wrong* (see Appendix Sec. A.4 for concrete examples on ImageNet). In fact, not using the hard label has another bonus. A dataset augmentation scheme which employs CE loss has to provide corresponding labels as supervisory information. In order to maintain the semantic correspondence, it cannot admit very extreme transformations for data augmentation. In contrast, in the Mixup/CutMix+KD setting described above, the data augmentation scheme need not worry about the labels as they are assigned *by the teacher* – the recent work [2] refers this kind of utilization of DA as *function matching* of the teacher. As a result, it can admit a *broader* set of transformations to expose the teacher's knowledge more completely. This reflects that data augmentation in KD has more freedom than in CE.

Among all the data augmentation techniques evaluated in this paper, we will show *CutMix works best*. The fundamental reason for its success, as we will show, is that it achieves a much lower variance of the teacher's mean probability, implying it produces more diverse data than its counterparts.

4.2 CutMixPick: Enhancing CutMix with Entropy-Based Data Picking

In this section, we propose a data picking scheme to further reduce the variance of the teacher's mean probability. The idea is partly inspired by *active learning* [38]. In active learning, the learner enjoys the freedom to query the data instances to be labeled for training by an oracle (*i.e.*, the teacher in our case) [38]. Since the augmented data can vary in their quality, we can introduce a certain criterion to pick the more valuable data for the student.

Intuitively, we regard a sample with more *information* is of higher quality. Therefore, we take *Shannon's entropy* of the teacher's output probability as a natural measure to select samples,

$$H(\mathbf{p}^{(t)}(x)) = -\mathbf{p}^{(t)}(x)^\top \log(\mathbf{p}^{(t)}(x)). \tag{11}$$

Note this formula is in exactly the same form of Eq. (8). Empirically, we will also show the data selected by this formula indeed results in lower $\bar{m}$ and better test loss for the student.

Concretely, given a batch of data, we first apply CutMix to obtain a bunch of augmented samples. Then sort all the augmented samples by Eq. (11) in ascending order and keep the top r (a pre-defined percentage constant, $r = 0.5$ in our experiments) samples.

This simple technique can be rather effective according to our empirical study. Another seemingly potential alternative is to use the *student's* entropy as the picking metric. The intuition behind this is that a high-entropy sample in the view of the student can be regarded as a hard example, too. Learning with these hard examples may expand the student's knowledge by squeezing its blind spots. Despite this intuitively plausible explanation, in practice, we will show this scheme actually under-performs the former student-agnostic scheme in Eq. (11), which is a bit surprising.

5 Experimental Results

Datasets and Networks. We evaluate our method primarily on the CIFAR100 [23] and Tiny ImageNet* datasets. CIFAR100 has 100 object classes (32×32 RGB images). Each class has 500 images for training and 100 images for testing. Tiny ImageNet is a small version of ImageNet [11] with 200 classes (64×64 RGB images). Each class has 500 images for training, 50 for validation and 50 for testing. To thoroughly evaluate our methods, we benchmark them on various standard network architectures: vgg [41], resnet [14], wrn [52], MobileNetV2 [36], ShuffleV2 [28]. We will also include results on ImageNet100 (a randomly drawn 100-class subset of ImageNet) and ImageNet.

Benchmark Methods. In addition to the standard cross-entropy training and the original KD method [17], we also compare with the state-of-the-art distillation approach, *Contrastive Representation Distillation* (CRD) [45]. It is important to note that our method focuses on improving KD by using better *inputs*, while CRD improves KD at the *output* end (*i.e.*, a better loss function). Therefore, they are orthogonal and we will show they can be combined together to deliver even better results.

Hyper-Parameter Settings. The temperature τ of knowledge distillation is set to 4 following CRD [45]. Loss weight $\alpha = 0.9$ (Eq. (1)). For CIFAR100 and Tiny ImageNet, training batch size is 64; the original number of total training epochs is 240, with learning rate (LR) decayed at epoch 150, 180, and 210 by multiplier 0.1. The initial LR is 0.05. All these settings are *the same* as CRD [45] for fair comparison. Note, in our experiments we will present the results of more training iterations. If the number of total epochs is scaled by a factor k, the epochs after which learning rate is decayed is also be scaled by k. For example, if we train a network for CIFAR100 for 480 epochs ($k = 2$) in total, the learning rate will be decayed at epoch 300, 360, and 420. We use PyTorch [33] to conduct all our experiments. For CIFAR100, we adopt the pretrained teacher models from CRD† for fair comparison. For Tiny ImageNet and ImageNet100, we train our own teacher models. For ImageNet, we use torchvision models following CRD [45].

Data Augmentation Schemes. We investigate the following popular DA schemes:

*https://tiny-imagenet.herokuapp.com/
†https://github.com/HobbitLong/RepDistiller

- **Identity**: This augmentation scheme simply makes a copy of each batch data during training, which should be the *lower bound* of all DA schemes discussed here since it adds no new information.
- **Flip**: Random horizontal flip.
- **Flip+Crop**: Random horizontal flip and random crop. This is the standard DA extensively used in 2D image recognition task (such as on CIFAR and ImageNet datasets).
- **Cutout** [12]: Cutout occludes a small random patch of an image.
- **AutoAugment** [8]: AutoAugment is an ensemble of a collected DA schemes. The DA policy is automatically selected by reinforcement learning instead of manually.
- **Mixup** [54]: Mixup applies linear interpolation between two inputs and applies the same linear interpolation to their labels to make the new label for the augmented sample.
- **CutMix** [51]: CutMix cuts a small patch from a source image and pastes it to another source image. The resulted image is considered as a new input. The target for the new input is a linear interpolation from the two source labels.

5.1 Empirical Verification of Our Proposition

Before presenting results, one point worth mentioning is that, in this section we use *test loss* instead of accuracy as the measure of student's performance in KD, because **(1)** all the formulas in Sec. 3 are derived using numerical loss instead of accuracy; **(2)** more importantly, it is observed that accuracy can mismatch with loss – a model may achieve a better accuracy, meanwhile higher loss too [29], which we also observe several times in our experiments. Using accuracy as measure would prevent us from seeing the correlation between T. stddev and student's performance (see Appendix Fig. 6 for an example). Potential extension of our theory from loss to accuracy is left for future work.

In Fig. 4, we plot the scatters of S. test loss and T. stddev. The x/y-axis value of each data point is averaged by at least three random runs (see Tabs 3/4 and Tabs 5/6 for detailed numbers).

(1) In terms of T. stddev, there is a rough trend (*e.g.*, on the vgg13/vgg8 pair on CIFAR100): Identity $<$ Flip $<$ Flip+Crop $<$ Cutout $<$ AutoAugment $<$ Mixup $<$ CutMix. These inequalities are well-aligned with our intuition. *E.g.*, AutoAugment [8] includes Cutout [12] in its transformation pool, thus should be stronger than Cutout. This is faithfully reflected by the T. stddev on many pairs.

(2) Obviously, S. test loss poses a *clear positive correlation* with T. stddev. Per our theory, lower T. stddev should lead to better generalization risk for the student. This is generally well verified in these plots. We do see some minor counterexamples. Possible reasons are: *1)* The test loss is obtained on the test set with finite samples, which is only an approximation of the true risk defined on distribution; *2)* The teacher's mean probability is also evaluated on finite data. Despite them, the general picture from Fig. 4 still confirms the positive correlation between S. test loss and T. stddev. The correlation is actually *very significant* as the p-values indicate.

(3) Importantly, note we only need the teacher to define the quality of a certain DA in KD, *no need for the student*. This is more clear if we examine the results in Tabs. 3/4 and Tabs. 5/6. Taking vgg13/vgg8 and vgg13/MobileNetV2 as an example, the students are starkly different but both the student's performance correlates well with T. stddev. This observation implies that *the "goodness" of DA in KD is probably student-invariant*. This, notably, is a great advantage in practice since we can decide which DA should be used for the best performance simply using a formula (Eq. (10)) with a few network forwards, without having to train the students physically.

5.2 Boosting KD with Stronger DA

In this section, we present more results to show how the proposed theory can benefit in practice – we can harvest considerable performance gain simply by using a stronger DA in KD.

Prolonged Training. Notably, a stronger DA produces more diverse data, implying more information. Intuitively, it should take a student model *more training iterations* (if the batch size does not change) to fully absorb the excessive information. That is, a stronger DA conceivably takes *more* training iterations to fully exhibit its potential. This intuition is confirmed in Fig. 2 (and also two more pairs wrn_40_2/wrn_16_2 and vgg13/vgg8 in the Appendix). Thus, in our experiments, we will also report results with prolonged training iterations for maximized performance.

Results on CIFAR100. The results on CIFAR100 dataset are shown in Tab. 1.

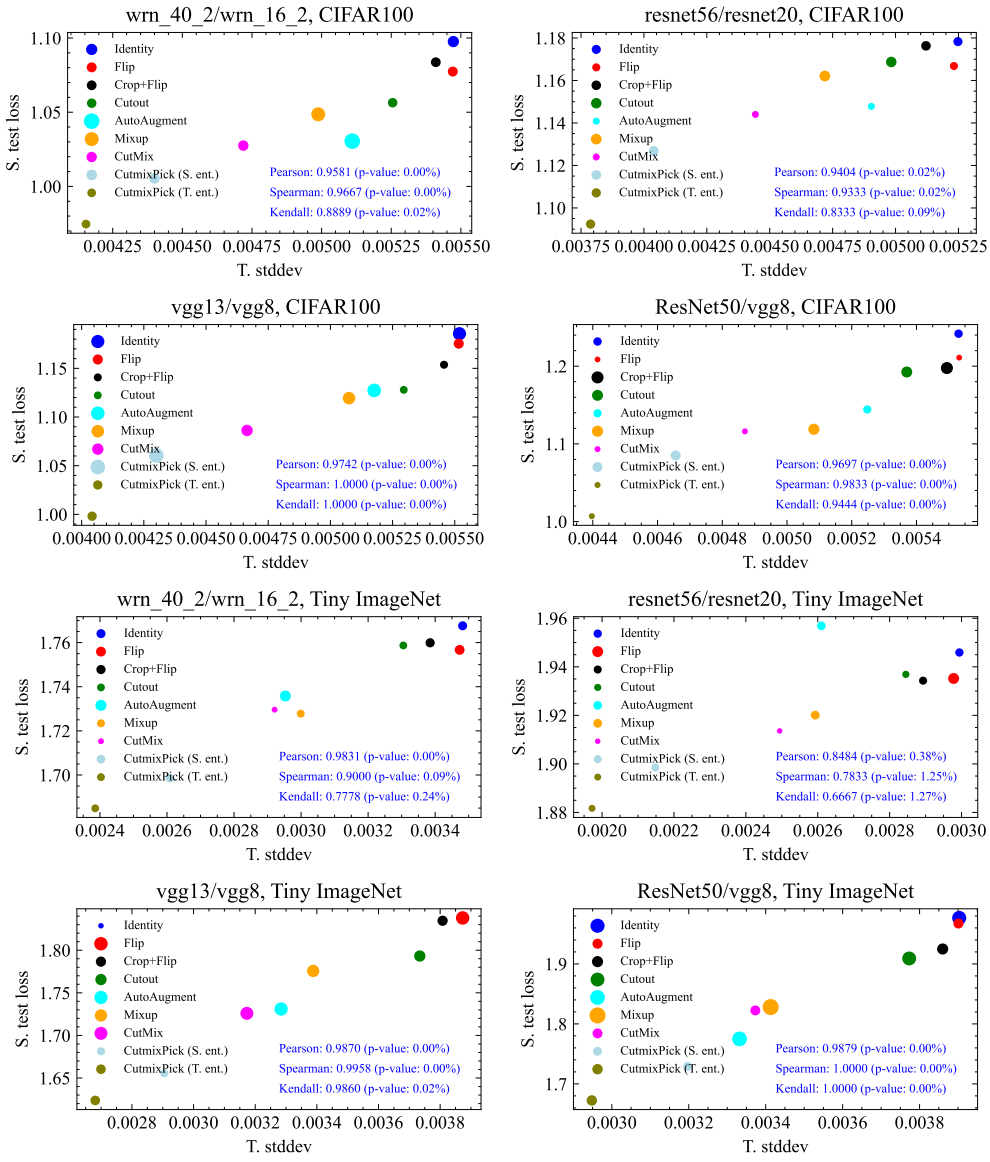


Figure 4: Scatter plots of T. stddev *vs.* S. test loss of different pairs on CIFAR100 and Tiny ImageNet. The detailed numbers are deferred to Appendix (Tab. 3, Tab. 4, Tab. 5, Tab. 6).

(1) Comparing the row "KD+CutMix" to "KD", we see CutMix improves the student accuracies on *all* the pairs. On the pair resnet32x4/ShuffleV2, the improvement is very significant (more than 1 percentage point). **(2)** Comparing the row "KD+CutMixPick" to "KD+CutMix", we see 6/7 pairs are improved further, showing the proposed data picking scheme works in most cases. **(3)** Finally, "KD+CutMixPick" scheme can be combined with more training iterations (960 epochs), which delivers even higher accuracies. **(4)** If comparing our best results (KD+CutMixPick$_{960}$) with those of CRD (though this is not an apples-to-apples comparison since the two methods focus on different aspects to improve KD), we can see our approach outperforms CRD on 6/7 pairs.

In the last two rows of Tab. 1, when CRD [45] is armed with our proposed "CutMixPick" and more training iterations, its results can be further advanced *consistently*. This demonstrates that our method is general and can readily work with those methods focusing on better KD loss functions.

Results on Tiny ImageNet. We also evaluate CutMix and CutMixPick on a more challenging dataset, Tiny ImageNet. Similar to the case on CIFAR100, we have results on different teacher-student pairs, shown in Tab. 2. For prolonged training, we train for 480 epochs instead of 960 to save time. Most claims on the CIFAR100 dataset are also validated here: **(1)** "KD+CutMix" is better than KD, which is verified on *all* pairs. **(2)** "KD+CutMixPick" is better than "KD+CutMix", verified on 6/7 pairs.

Table 1: Student test accuracy on **CIFAR100**. Each result is obtained by 3 random runs, mean (std) accuracy reported. The best results are in **bold** and second best underlined. The subscript 960 means the total number of training epochs (default: 240).

Teacher Student	wrn_40_2 wrn_16_2	resnet56 resnet20	resnet32x4 resnet8x4	vgg13 vgg8	vgg13 MobileNetV2	ResNet50 vgg8	resnet32x4 ShuffleV2
Teacher Acc.	75.61	72.34	79.42	74.64	74.64	79.34	79.42
Student Acc.	73.26	69.06	72.50	70.36	64.60	70.36	71.82
KD [17]	74.92 (0.28)	70.66 (0.24)	73.33 (0.25)	72.98 (0.19)	67.37 (0.32)	73.81 (0.13)	74.45 (0.27)
KD+CutMix	75.34 (0.19)	70.77 (0.17)	74.91 (0.20)	74.16 (0.18)	68.79 (0.35)	74.85 (0.23)	76.61 (0.18)
KD+CutMixPick	75.59 (0.22)	70.99 (0.20)	74.78 (0.35)	74.43 (0.20)	69.49 (0.32)	74.95 (0.18)	76.90 (0.25)
KD_{960} [17]	75.68 (0.12)	**71.79** (0.29)	73.14 (0.06)	74.00 (0.34)	68.77 (0.05)	74.04 (0.25)	74.64 (0.30)
KD+CutMixPick$_{960}$	**76.41** (0.10)	71.66 (0.15)	**75.12** (0.18)	**75.00** (0.17)	**70.47** (0.12)	**76.13** (0.16)	**77.90** (0.30)
CRD [45]	75.64 (0.21)	71.63 (0.15)	75.46 (0.25)	74.29 (0.12)	69.94 (0.05)	74.58 (0.27)	76.05 (0.09)
CRD+**CutMixPick**	75.96 (0.27)	71.41 (0.26)	**76.11** (0.53)	74.65 (0.12)	69.95 (0.22)	75.35 (0.22)	76.93 (0.11)
CRD+**CutMixPick**$_{960}$	**76.61** (0.01)	**72.40** (0.20)	75.96 (0.29)	**75.41** (0.10)	**70.84** (0.05)	**76.20** (0.22)	**78.51** (0.27)

Table 2: Student test accuracy on **Tiny ImageNet**. The subscript 480 means the total number of training epochs (default: 240).

Teacher Student	wrn_40_2 wrn_16_2	resnet56 resnet20	resnet32x4 resnet8x4	vgg13 vgg8	vgg13 MobileNetV2	ResNet50 vgg8	resnet32x4 ShuffleV2
Teacher Acc.	61.28	58.37	64.41	62.59	62.59	68.20	64.41
Student Acc.	58.23	52.53	55.41	56.67	58.20	56.67	62.07
KD [17]	58.65 (0.09)	53.58 (0.18)	55.67 (0.09)	61.48 (0.36)	59.28 (0.13)	60.39 (0.16)	66.34 (0.11)
KD+CutMix	59.06 (0.18)	53.77 (0.33)	56.41 (0.04)	62.17 (0.11)	60.48 (0.30)	61.12 (0.18)	67.01 (0.30)
KD+CutMixPick	59.22 (0.05)	53.66 (0.05)	56.82 (0.23)	62.32 (0.18)	60.53 (0.18)	61.40 (0.26)	67.08 (0.13)
KD_{480} [17]	59.20 (0.30)	54.23 (0.24)	55.49 (0.11)	61.72 (0.10)	59.27 (0.08)	60.10 (0.30)	65.81 (0.11)
KD+CutMixPick$_{480}$	**60.07** (0.04)	**54.25** (0.07)	**57.54** (0.23)	**62.60** (0.25)	**60.66** (0.15)	**61.95** (0.14)	**67.35** (0.21)
CRD [45]	60.79 (0.24)	55.34 (0.02)	59.28 (0.13)	62.92 (0.31)	62.38 (0.19)	62.03 (0.16)	67.33 (0.13)
CRD+**CutMixPick**	60.72 (0.09)	54.99 (0.16)	59.65 (0.24)	63.39 (0.10)	62.54 (0.22)	**62.85** (0.18)	67.64 (0.18)
CRD+**CutMixPick**$_{480}$	**60.99** (0.33)	**55.68** (0.22)	**60.13** (0.13)	**63.60** (0.20)	**62.79** (0.03)	62.60 (0.17)	**67.70** (0.35)

The exception pair is resnet56/resnet20, where adding data picking decreases the accuracy slightly by 0.11%. **(3)** When "KD+CutMixPick" is trained for prolonged iterations, the students perform best.

We further evaluate our DA methods equipped with CRD [45], shown in the last two rows of Tab. 2. Our "CutMixPick" method further advances the prior SOTA on 5 pairs. When CRD+CutMixPick is trained for 480 epochs (instead of 240), further improvement can be observed on 6 of 7 pairs.

Apply DA to More KD Methods. Notably, we achieve the above performance boosting simply using the original KD loss [17], *with no bells and whistles*. This justifies one of our motivations in this paper, *i.e.*, existing KD methods [34, 31, 45] mainly improve KD at the network *output* side via better loss functions, while we propose to improve KD at the *input* side and show this path is just as promising. Actually, this performance boosting effect is *generic* – we also applied CutMix to another 5 top-performing KD methods on CIFAR100: AT [53], CC [34], SP [46], PKT [32], and VID [1]. *All* the pairs see accuracy gains; half of them are even improved by more than 1% point.

Results on ImageNet100 and ImageNet. These results are deferred to Appendix A.1. In general, we observe that the correlation between T. stddev and S. test loss become weaker on ImageNet100 and ImageNet. This is because these two datasets are inherently harder than CIFAR100 and Tiny ImageNet. Nevertheless, the p-value of the correlation is still below 5% on ImageNet100, *i.e.*, still statistically significant, suggesting our theory can generalize to large-resolution (224×224) datasets.

6 Conclusion

In this paper, we attempt to precisely answer what makes a good data augmentation in knowledge distillation. By analyzing the generalization gap of the student under different sampling schemes, we reach the conclusion that a good data augmentation scheme should reduce the variance of the cross-entropy (*i.e.*, the distilled risk) between the teacher and student. Based on this, we propose a new metric, the stddev of the teacher's mean probability (T. stddev), as a feasible measure of the quality of data augmentation techniques. Empirical studies with various teacher-student pairs confirm the efficacy of the proposed metric for data augmentation quality in KD. Besides the theoretical understanding, we also develop an entropy-based data picking scheme to further enhance the prior

best augmentation scheme (CutMix) in KD. Finally, we show how we can obtain considerable KD performance gains simply by using a stronger DA guided by the proposed theory.

Acknowledgments and Disclosure of Funding

We thank the anonymous NeurIPS reviewers for giving us very helpful suggestions to improve this paper!

This work originates from Huan's internship at MERL, and is finished eventually after he returns to Northeastern University as a research assistant. This work is thus fully supported by MERL and Northeastern University. There is no third-party funding or support in any form. There are no competing interests to disclose.

References

[1] Sungsoo Ahn, Shell Xu Hu, Andreas Damianou, Neil D Lawrence, and Zhenwen Dai. Variational information distillation for knowledge transfer. In *CVPR*, 2019. 10

[2] Lucas Beyer, Xiaohua Zhai, Amélie Royer, Larisa Markeeva, Rohan Anil, and Alexander Kolesnikov. Knowledge distillation: A good teacher is patient and consistent. In *CVPR*, 2022. 3, 6

[3] Xavier Bouthillier, Kishore Konda, Pascal Vincent, and Roland Memisevic. Dropout as data augmentation. *arXiv preprint arXiv:1506.08700*, 2015. 2

[4] Cristian Buciluǎ, Rich Caruana, and Alexandru Niculescu-Mizil. Model compression. In *SIGKDD*, 2006. 3

[5] Keshigeyan Chandrasegaran, Ngoc-Trung Tran, Yunqing Zhao, and Ngai-Man Cheung. Revisiting label smoothing and knowledge distillation compatibility: What was missing? In *ICML*, 2022. 3

[6] Guobin Chen, Wongun Choi, Xiang Yu, Tony Han, and Manmohan Chandraker. Learning efficient object detection models with knowledge distillation. In *NeurIPS*, 2017. 2, 3

[7] Jang Hyun Cho and Bharath Hariharan. On the efficacy of knowledge distillation. In *ICCV*, 2019. 15

[8] Ekin D Cubuk, Barret Zoph, Dandelion Mane, Vijay Vasudevan, and Quoc V Le. Autoaugment: Learning augmentation policies from data. In *CVPR*, 2019. 2, 3, 8

[9] Wanyun Cui and Sen Yan. Isotonic data augmentation for knowledge distillation. *arXiv preprint arXiv:2107.01412*, 2021. 3

[10] Deepan Das, Haley Massa, Abhimanyu Kulkarni, and Theodoros Rekatsinas. An empirical analysis of the impact of data augmentation on knowledge distillation. *arXiv preprint arXiv:2006.03810*, 2020. 4

[11] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *CVPR*, 2009. 3, 7

[12] Terrance DeVries and Graham W Taylor. Improved regularization of convolutional neural networks with cutout. *arXiv preprint arXiv:1708.04552*, 2017. 3, 8

[13] Yushu Feng, Huan Wang, Haoji Hu, and Daniel Yi. Triplet distillation for deep face recognition. In *ICML Workshop*, 2019. 2

[14] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *CVPR*, 2016. 2, 3, 7

[15] Byeongho Heo, Jeesoo Kim, Sangdoo Yun, Hyojin Park, Nojun Kwak, and Jin Young Choi. A comprehensive overhaul of feature distillation. In *ICCV*, 2019. 3

[16] Byeongho Heo, Minsik Lee, Sangdoo Yun, and Jin Young Choi. Knowledge transfer via distillation of activation boundaries formed by hidden neurons. In *AAAI*, 2019. 3

[17] Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. Distilling the knowledge in a neural network. In *NeurIPS Workshop*, 2014. 2, 3, 7, 10

[18] Geoffrey E Hinton. Training products of experts by minimizing contrastive divergence. *Neural Computation*, 14(8):1771–1800, 2002. 2

[19] Gao Huang, Zhuang Liu, Laurens Van Der Maaten, and Kilian Q Weinberger. Densely connected convolutional networks. In *CVPR*, 2017. 2

[20] Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *ICML*, 2015. 2

[21] Xiaoqi Jiao, Yichun Yin, Lifeng Shang, Xin Jiang, Xiao Chen, Linlin Li, Fang Wang, and Qun Liu. Tinybert: Distilling bert for natural language understanding. *arXiv preprint arXiv:1909.10351*, 2019. 2, 3

[22] Michael J Kearns, Umesh Virkumar Vazirani, and Umesh Vazirani. *An introduction to computational learning theory*. MIT Press, 1994. 15, 16

[23] Alex Krizhevsky. Learning multiple layers of features from tiny images. Technical report, Citeseer, 2009. 7

[24] Solomon Kullback. *Information theory and statistics*. Courier Corporation, 1997. 2

[25] Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. Deep learning. *Nature*, 521(7553):436, 2015. 2

[26] Weiyang Liu, Yandong Wen, Zhiding Yu, and Meng Yang. Large-margin softmax loss for convolutional neural networks. In *ICML*, 2016. 2

[27] Yufan Liu, Jiajiong Cao, Bing Li, Chunfeng Yuan, Weiming Hu, Yangxi Li, and Yunqiang Duan. Knowledge distillation via instance relationship graph. In *CVPR*, 2019. 3

[28] Ningning Ma, Xiangyu Zhang, Hai-Tao Zheng, and Jian Sun. Shufflenet v2: Practical guidelines for efficient cnn architecture design. In *ECCV*, 2018. 7

[29] Aditya K Menon, Ankit Singh Rawat, Sashank Reddi, Seungyeon Kim, and Sanjiv Kumar. A statistical perspective on distillation. In *ICML*, 2021. 4, 8, 16

[30] Rafael Müller, Simon Kornblith, and Geoffrey E Hinton. When does label smoothing help? In *NeurIPS*, 2019. 2, 3

[31] Wonpyo Park, Dongju Kim, Yan Lu, and Minsu Cho. Relational knowledge distillation. In *CVPR*, 2019. 2, 3, 10

[32] Nikolaos Passalis and Anastasios Tefas. Learning deep representations with probabilistic knowledge transfer. In *ECCV*, 2018. 3, 10

[33] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. Pytorch: An imperative style, high-performance deep learning library. In *NeurIPS*, 2019. 7

[34] Baoyun Peng, Xiao Jin, Jiaheng Liu, Dongsheng Li, Yichao Wu, Yu Liu, Shunfeng Zhou, and Zhaoning Zhang. Correlation congruence for knowledge distillation. In *ICCV*, 2019. 2, 3, 10

[35] Adriana Romero, Nicolas Ballas, Samira Ebrahimi Kahou, Antoine Chassang, Carlo Gatta, and Yoshua Bengio. Fitnets: Hints for thin deep nets. In *ICLR*, 2015. 3

[36] Mark Sandler, Andrew Howard, Menglong Zhu, Andrey Zhmoginov, and Liang-Chieh Chen. Mobilenetv2: Inverted residuals and linear bottlenecks. In *CVPR*, 2018. 7

[37] Jürgen Schmidhuber. Deep learning in neural networks: An overview. *Neural networks*, 61:85–117, 2015. 2

[38] Burr Settles. From theories to queries: Active learning in practice. In *AISTATS Workshop on Active Learning and Experimental Design*, 2011. 7

[39] Zhiqiang Shen, Zechun Liu, Dejia Xu, Zitian Chen, Kwang-Ting Cheng, and Marios Savvides. Is label smoothing truly incompatible with knowledge distillation: An empirical study. In *ICLR*, 2021. 3

[40] Connor Shorten and Taghi M Khoshgoftaar. A survey on image data augmentation for deep learning. *Journal of Big Data*, 6(1):60, 2019. 2, 3

[41] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. In *ICLR*, 2015. 7

[42] Jy-yong Sohn, Liang Shang, Hongxu Chen, Jaekyun Moon, Dimitris Papailiopoulos, and Kangwook Lee. Genlabel: Mixup relabeling using generative models. In *ICML*, 2022. 4

[43] Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jon Shlens, and Zbigniew Wojna. Rethinking the inception architecture for computer vision. In *CVPR*, 2016. 2

[44] Jiaxi Tang, Rakesh Shivanna, Zhe Zhao, Dong Lin, Anima Singh, Ed H Chi, and Sagar Jain. Understanding and improving knowledge distillation. *arXiv preprint arXiv:2002.03532*, 2020. 3

[45] Yonglong Tian, Dilip Krishnan, and Phillip Isola. Contrastive representation distillation. In *ICLR*, 2020. 2, 3, 7, 9, 10, 15

[46] Frederick Tung and Greg Mori. Similarity-preserving knowledge distillation. In *CVPR*, 2019. 3, 10

[47] Vladimir Vapnik. *The nature of statistical learning theory*. Springer Science & Business Media, 2013. 15, 16

[48] Vikas Verma, Alex Lamb, Christopher Beckham, Amir Najafi, Ioannis Mitliagkas, David Lopez-Paz, and Yoshua Bengio. Manifold mixup: Better representations by interpolating hidden states. In *ICML*, 2019. 3

[49] Huan Wang, Yijun Li, Yuehai Wang, Haoji Hu, and Ming-Hsuan Yang. Collaborative distillation for ultra-resolution universal style transfer. In *CVPR*, 2020. 2, 3

[50] Lin Wang and Kuk-Jin Yoon. Knowledge distillation and student-teacher learning for visual intelligence: A review and new outlooks. *TPAMI*, 2021. 2, 3

[51] Sangdoo Yun, Dongyoon Han, Seong Joon Oh, Sanghyuk Chun, Junsuk Choe, and Youngjoon Yoo. Cutmix: Regularization strategy to train strong classifiers with localizable features. In *ICCV*, 2019. 3, 8

[52] Sergey Zagoruyko and Nikos Komodakis. Wide residual networks. In *BMVC*, 2016. 7

[53] Sergey Zagoruyko and Nikos Komodakis. Paying more attention to attention: Improving the performance of convolutional neural networks via attention transfer. In *ICLR*, 2017. 3, 10

[54] Hongyi Zhang, Moustapha Cisse, Yann N Dauphin, and David Lopez-Paz. mixup: Beyond empirical risk minimization. In *ICLR*, 2018. 2, 3, 8

Checklist

1. For all authors...
   (a) Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope? [Yes]
   (b) Did you describe the limitations of your work? [Yes] See our Appendix.
   (c) Did you discuss any potential negative societal impacts of your work? [Yes] See our Appendix.
   (d) Have you read the ethics review guidelines and ensured that your paper conforms to them? [Yes]
2. If you are including theoretical results...
   (a) Did you state the full set of assumptions of all theoretical results? [Yes] See Sec. 3.
   (b) Did you include complete proofs of all theoretical results? [Yes] See Sec. 3.
3. If you ran experiments...
   (a) Did you include the code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL)? [Yes] See our Appendix.
   (b) Did you specify all the training details (e.g., data splits, hyperparameters, how they were chosen)? [Yes] See our Appendix.
   (c) Did you report error bars (e.g., with respect to the random seed after running experiments multiple times)? [Yes] *All* of our CIFAR100/Tiny ImageNet/ImageNet100 results are averaged by at least three random runs, mean and stddev reported.

(d) Did you include the total amount of compute and the type of resources used (e.g., type of GPUs, internal cluster, or cloud provider)? [Yes] See our Appendix.

4. If you are using existing assets (e.g., code, data, models) or curating/releasing new assets...
   (a) If your work uses existing assets, did you cite the creators? [Yes]
   (b) Did you mention the license of the assets? [Yes] See our Appendix.
   (c) Did you include any new assets either in the supplemental material or as a URL? [Yes] We include the code link.
   (d) Did you discuss whether and how consent was obtained from people whose data you're using/curating? [Yes] The data we use are all publicly available.
   (e) Did you discuss whether the data you are using/curating contains personally identifiable information or offensive content? [Yes] The data we use contains *no* personally identifiable information or offensive content.
5. If you used crowdsourcing or conducted research with human subjects...
   (a) Did you include the full text of instructions given to participants and screenshots, if applicable? [No] No crowdsourcing in this work.
   (b) Did you describe any potential participant risks, with links to Institutional Review Board (IRB) approvals, if applicable? [No] No crowdsourcing in this work.
   (c) Did you include the estimated hourly wage paid to participants and the total amount spent on participant compensation? [No] No crowdsourcing in this work.

A Appendix

A.1 More Results

Detailed numerical results of S. test loss and T. stddev on CIFAR100 and Tiny ImageNet. See Tab. 3, Tab. 4, Tab. 5, Tab. 6. These tables are the numerical results that we use to plot Fig. 4.

Table 3: Student test loss (S. test loss) and the stddev of teacher's mean probability (T. stddev, $\times 10^{-3}$) comparison on **CIFAR100** when using different DA schemes.

Teacher Student Metric	wrn_40_2 / T. stddev	wrn_40_2 wrn_16_2 S. test loss	wrn_40_2 vgg8 S. test loss	resnet56 / T. stddev	resnet56 resnet20 S. test loss	resnet56 ShuffleV2 S. test loss
KD+Identity	$5.473_{\pm 0.002}$	$1.0976_{\pm 0.0136}$	$1.1830_{\pm 0.0065}$	$5.248_{\pm 0.004}$	$1.1783_{\pm 0.0081}$	$0.9785_{\pm 0.0137}$
KD+Flip	$5.471_{\pm 0.004}$	$1.0774_{\pm 0.0101}$	$1.1673_{\pm 0.0060}$	$5.232_{\pm 0.003}$	$1.1668_{\pm 0.0060}$	$0.9961_{\pm 0.0072}$
KD+Crop+Flip	$5.410_{\pm 0.009}$	$1.0837_{\pm 0.0097}$	$1.1446_{\pm 0.0182}$	$5.121_{\pm 0.006}$	$1.1763_{\pm 0.0092}$	$0.9736_{\pm 0.0062}$
KD+Cutout	$5.255_{\pm 0.009}$	$1.0564_{\pm 0.0090}$	$1.1306_{\pm 0.0156}$	$4.983_{\pm 0.012}$	$1.1687_{\pm 0.0115}$	$0.9541_{\pm 0.0088}$
KD+AutoAugment	$5.110_{\pm 0.004}$	$1.0305_{\pm 0.0290}$	$1.1102_{\pm 0.0016}$	$4.904_{\pm 0.007}$	$1.1478_{\pm 0.0041}$	$0.9355_{\pm 0.0099}$
KD+Mixup	$4.988_{\pm 0.012}$	$1.0486_{\pm 0.0225}$	$1.0917_{\pm 0.0044}$	$4.719_{\pm 0.002}$	$1.1621_{\pm 0.0121}$	$0.9703_{\pm 0.0060}$
KD+CutMix	$4.719_{\pm 0.005}$	$1.0275_{\pm 0.0112}$	$1.0657_{\pm 0.0145}$	$4.443_{\pm 0.002}$	$1.1440_{\pm 0.0040}$	$0.9348_{\pm 0.0107}$
KD+CutMixPick (S. ent.)	$4.400_{\pm 0.007}$	$1.0054_{\pm 0.0118}$	$1.0471_{\pm 0.0111}$	$4.039_{\pm 0.004}$	$1.1269_{\pm 0.0095}$	$0.9339_{\pm 0.0070}$
KD+CutMixPick (T. ent.)	$4.154_{\pm 0.005}$	$0.9746_{\pm 0.0073}$	$0.9928_{\pm 0.0061}$	$3.788_{\pm 0.004}$	$1.0924_{\pm 0.0085}$	$0.9038_{\pm 0.0148}$

Table 4: **Continued**: Student test loss (S. test loss) and the stddev of teacher's mean probability (T. stddev, $\times 10^{-3}$) comparison on **CIFAR100** when using different DA schemes.

Teacher Student Metric	vgg13 / T. stddev	vgg13 vgg8 S. test loss	vgg13 MobileNetV2 S. test loss	resnet32x4 / T. stddev	resnet32x4 resnet8x4 S. test loss	resnet32x4 ShuffleV2 S. test loss	ResNet50 / T. stddev	ResNet50 vgg8 S. test loss
KD+Identity	$5.519_{\pm 0.006}$	$1.1856_{\pm 0.0196}$	$1.6021_{\pm 0.0366}$	$5.524_{\pm 0.015}$	$1.0973_{\pm 0.0134}$	$1.3381_{\pm 0.0072}$	$5.530_{\pm 0.005}$	$1.2418_{\pm 0.0067}$
KD+Flip	$5.516_{\pm 0.002}$	$1.1754_{\pm 0.0105}$	$1.5779_{\pm 0.0076}$	$5.530_{\pm 0.009}$	$1.0967_{\pm 0.0038}$	$1.3208_{\pm 0.0172}$	$5.532_{\pm 0.002}$	$1.2110_{\pm 0.0024}$
KD+Crop+Flip	$5.457_{\pm 0.010}$	$1.1537_{\pm 0.0059}$	$1.5244_{\pm 0.0149}$	$5.498_{\pm 0.004}$	$1.0685_{\pm 0.0031}$	$1.3091_{\pm 0.0283}$	$5.494_{\pm 0.013}$	$1.1976_{\pm 0.0160}$
KD+Cutout	$5.295_{\pm 0.012}$	$1.1279_{\pm 0.0055}$	$1.4963_{\pm 0.0104}$	$5.356_{\pm 0.014}$	$1.0627_{\pm 0.0063}$	$1.2528_{\pm 0.0127}$	$5.370_{\pm 0.004}$	$1.1925_{\pm 0.0128}$
KD+AutoAugment	$5.176_{\pm 0.012}$	$1.1273_{\pm 0.0213}$	$1.4137_{\pm 0.0440}$	$5.198_{\pm 0.015}$	$1.0289_{\pm 0.0078}$	$1.1353_{\pm 0.0164}$	$5.248_{\pm 0.011}$	$1.1443_{\pm 0.0064}$
KD+Mixup	$5.075_{\pm 0.001}$	$1.1194_{\pm 0.0176}$	$1.4009_{\pm 0.0044}$	$5.082_{\pm 0.004}$	$1.0182_{\pm 0.0128}$	$1.0840_{\pm 0.0229}$	$5.083_{\pm 0.006}$	$1.1188_{\pm 0.0133}$
KD+CutMix	$4.665_{\pm 0.009}$	$1.0862_{\pm 0.0140}$	$1.2846_{\pm 0.0178}$	$4.851_{\pm 0.016}$	$1.0166_{\pm 0.0096}$	$1.0544_{\pm 0.0095}$	$4.870_{\pm 0.002}$	$1.1161_{\pm 0.0026}$
KD+CutMixPick (S. ent.)	$4.300_{\pm 0.016}$	$1.0605_{\pm 0.0249}$	$1.2739_{\pm 0.0064}$	$4.590_{\pm 0.006}$	$0.9888_{\pm 0.0015}$	$1.0270_{\pm 0.0047}$	$4.656_{\pm 0.012}$	$1.0851_{\pm 0.0101}$
KD+CutMixPick (T. ent.)	$4.042_{\pm 0.007}$	$0.9981_{\pm 0.0087}$	$1.1876_{\pm 0.0233}$	$4.323_{\pm 0.007}$	$0.9485_{\pm 0.0116}$	$0.9570_{\pm 0.0122}$	$4.397_{\pm 0.006}$	$1.0069_{\pm 0.0027}$

ImageNet100 and ImageNet Results. See Fig. 5, Tab. 7 and Tab. 8 for the results. In general, we find it is harder to verify the proposed proposition on these two more challenging datasets – As seen,

Table 5: Student test loss (S. test loss) and the stddev of teacher's mean probability (T. stddev, $\times 10^{-3}$) comparison on **Tiny ImageNet** when using different DA schemes.

Teacher Student Metric	wrn_40_2 / T. stddev	wrn_40_2 wrn_16_2 S. test loss	wrn_40_2 vgg8 S. test loss	resnet56 / T. stddev	resnet56 resnet20 S. test loss	resnet56 ShuffleV2 S. test loss
KD+Identity	$3.482_{\pm 0.002}$	$1.7676_{\pm 0.0079}$	$1.7374_{\pm 0.0111}$	$2.994_{\pm 0.001}$	$1.9459_{\pm 0.0065}$	$1.5600_{\pm 0.0077}$
KD+Flip	$3.473_{\pm 0.003}$	$1.7567_{\pm 0.0099}$	$1.7322_{\pm 0.0107}$	$2.978_{\pm 0.001}$	$1.9352_{\pm 0.0127}$	$1.5629_{\pm 0.0034}$
KD+Crop+Flip	$3.385_{\pm 0.004}$	$1.7599_{\pm 0.0083}$	$1.7283_{\pm 0.0083}$	$2.893_{\pm 0.000}$	$1.9343_{\pm 0.0060}$	$1.5593_{\pm 0.0048}$
KD+Cutout	$3.305_{\pm 0.002}$	$1.7587_{\pm 0.0055}$	$1.7086_{\pm 0.0041}$	$2.845_{\pm 0.003}$	$1.9369_{\pm 0.0044}$	$1.5636_{\pm 0.0052}$
KD+AutoAugment	$2.953_{\pm 0.002}$	$1.7358_{\pm 0.0131}$	$1.7086_{\pm 0.0386}$	$2.610_{\pm 0.002}$	$1.9569_{\pm 0.0071}$	$1.5782_{\pm 0.0085}$
KD+Mixup	$2.999_{\pm 0.002}$	$1.7278_{\pm 0.0057}$	$1.7024_{\pm 0.0066}$	$2.593_{\pm 0.004}$	$1.9201_{\pm 0.0075}$	$1.5765_{\pm 0.0036}$
KD+CutMix	$2.921_{\pm 0.005}$	$1.7296_{\pm 0.0025}$	$1.6905_{\pm 0.0033}$	$2.494_{\pm 0.005}$	$1.9136_{\pm 0.0020}$	$1.5605_{\pm 0.0024}$
KD+CutMixPick (S. ent.)	$2.609_{\pm 0.002}$	$1.6986_{\pm 0.0071}$	$1.6524_{\pm 0.0058}$	$2.148_{\pm 0.001}$	$1.8985_{\pm 0.0059}$	$1.5580_{\pm 0.0056}$
KD+CutMixPick (T. ent.)	$2.386_{\pm 0.002}$	$1.6849_{\pm 0.0055}$	$1.6349_{\pm 0.0060}$	$1.972_{\pm 0.003}$	$1.8817_{\pm 0.0040}$	$1.5429_{\pm 0.0060}$

Table 6: **Continued**: Student test loss (S. test loss) and the stddev of teacher's mean probability (T. stddev, $\times 10^{-3}$) comparison on **Tiny ImageNet** when using different DA schemes.

Teacher Student Metric	vgg13 / T. stddev	vgg13 vgg8 S. test loss	vgg13 MobileNetV2 S. test loss	resnet32x4 / T. stddev	resnet32x4 resnet8x4 S. test loss	resnet32x4 ShuffleV2 S. test loss	ResNet50 / T. stddev	ResNet50 vgg8 S. test loss
KD+Identity	$3.873_{\pm 0.001}$	$1.8377_{\pm 0.0021}$	$2.0258_{\pm 0.0211}$	$3.847_{\pm 0.004}$	$1.9859_{\pm 0.0108}$	$1.7404_{\pm 0.2064}$	$3.903_{\pm 0.001}$	$1.9767_{\pm 0.0230}$
KD+Flip	$3.873_{\pm 0.002}$	$1.8379_{\pm 0.0207}$	$1.9946_{\pm 0.0180}$	$3.842_{\pm 0.005}$	$1.9531_{\pm 0.0049}$	$1.6092_{\pm 0.0077}$	$3.901_{\pm 0.001}$	$1.9674_{\pm 0.0104}$
KD+Crop+Flip	$3.808_{\pm 0.004}$	$1.8346_{\pm 0.0108}$	$1.9707_{\pm 0.0089}$	$3.784_{\pm 0.002}$	$1.9725_{\pm 0.0092}$	$1.6009_{\pm 0.0111}$	$3.860_{\pm 0.003}$	$1.9249_{\pm 0.0132}$
KD+Cutout	$3.734_{\pm 0.066}$	$1.7932_{\pm 0.0137}$	$1.9544_{\pm 0.0101}$	$3.688_{\pm 0.001}$	$1.9872_{\pm 0.0047}$	$1.5900_{\pm 0.0092}$	$3.773_{\pm 0.003}$	$1.9091_{\pm 0.0223}$
KD+AutoAugment	$3.284_{\pm 0.003}$	$1.7310_{\pm 0.0200}$	$1.7970_{\pm 0.0445}$	$3.269_{\pm 0.003}$	$1.9037_{\pm 0.0248}$	$1.4969_{\pm 0.0159}$	$3.332_{\pm 0.005}$	$1.7750_{\pm 0.0255}$
KD+Mixup	$3.388_{\pm 0.002}$	$1.7757_{\pm 0.0168}$	$1.8587_{\pm 0.0141}$	$3.326_{\pm 0.002}$	$1.9191_{\pm 0.0136}$	$1.5368_{\pm 0.0108}$	$3.413_{\pm 0.001}$	$1.8281_{\pm 0.0309}$
KD+CutMix	$3.173_{\pm 0.001}$	$1.7259_{\pm 0.0189}$	$1.8269_{\pm 0.0423}$	$3.275_{\pm 0.003}$	$1.9420_{\pm 0.0082}$	$1.5350_{\pm 0.0083}$	$3.373_{\pm 0.004}$	$1.8225_{\pm 0.0101}$
KD+CutMixPick (S. ent.)	$2.905_{\pm 0.000}$	$1.6556_{\pm 0.0061}$	$1.7509_{\pm 0.0080}$	$3.086_{\pm 0.001}$	$1.8355_{\pm 0.0144}$	$1.5419_{\pm 0.0376}$	$3.197_{\pm 0.001}$	$1.7290_{\pm 0.0077}$
KD+CutMixPick (T. ent.)	$2.681_{\pm 0.005}$	$1.6237_{\pm 0.0094}$	$1.6839_{\pm 0.0159}$	$2.812_{\pm 0.002}$	$1.8368_{\pm 0.0060}$	$1.4317_{\pm 0.0096}$	$2.948_{\pm 0.004}$	$1.6724_{\pm 0.0113}$

the coefficients turn smaller while p-values larger than the cases on CIFAR100 and Tiny ImageNet. On ImageNet100, the p-values are below (or very close to) 5%, suggesting the correlation is still strong. On ImageNet, however, the results show a *weak* correlation between T. stddev and S. test loss, against our theory. Currently, we are not very sure why this happens sorely on ImageNet. After all, ImageNet100 is a subset of ImageNet and the proposed theory works fairly well on ImageNet100. One possible reason may be – full ImageNet historically was shown especially hard to make KD work, *e.g.*, in [7], they mentioned "*(Page 4) ...it is still a mystery why no teacher improves accuracy on ImageNet. Despite multiple recent papers in knowledge distillation, experiments on ImageNet are rarely reported. The few that do report find that standard setting of knowledge distillation fails on ImageNet [26] or perform an experiment with a small portion of ImageNet [21]*". Besides, we may notice the authors of CRD [45] mentioned in their GitHub issue[‡]: "*I have been struggling a bit to get KD work as well on ImageNet with ResNet-18 as the student network*". We conceive that the weak correlation on ImageNet may be related to this KD underperformance on ImageNet and may be related to the number of classes of the dataset. We shall continue to investigate this in our future work.

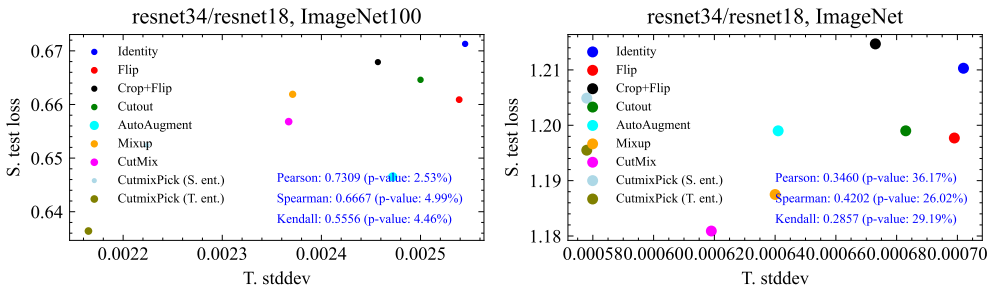


Figure 5: Scatter plots of T. stddev *vs.* S. test loss of different pairs on **ImageNet100** and **ImageNet**.

Discussion: This paper discovers that "more correlation in the data, worse generalization ability". Is this an already-known fact in statistical learning [22, 47]? Our work is *starkly different* from the established theory in [22, 47] in that they study the dependency *in the input data*,

[‡]https://github.com/HobbitLong/RepDistiller/issues/10#issuecomment-563078837

Table 7: Student test loss (S. test loss) and the stddev of teacher's mean probability (T. stddev, $\times 10^{-3}$) comparison on **ImageNet100** (a subset with 100 classes randomly drawn from ImageNet) when using different data augmentation schemes.

Teacher Student Metric	resnet34 / T. stddev	resnet34 resnet18 S. test loss
KD+Identity	$2.545_{\pm 0.000}$	$0.6713_{\pm 0.0029}$
KD+Flip	$2.539_{\pm 0.004}$	$0.6609_{\pm 0.0035}$
KD+Crop+Flip	$2.457_{\pm 0.008}$	$0.6679_{\pm 0.0027}$
KD+Cutout	$2.500_{\pm 0.005}$	$0.6646_{\pm 0.0030}$
KD+AutoAugment	$2.472_{\pm 0.007}$	$0.6465_{\pm 0.0078}$
KD+Mixup	$2.371_{\pm 0.005}$	$0.6619_{\pm 0.0043}$
KD+CutMix	$2.367_{\pm 0.003}$	$0.6568_{\pm 0.0048}$
KD+CutMixPick (S. ent.)	$2.224_{\pm 0.006}$	$0.6524_{\pm 0.0015}$
KD+CutMixPick (T. ent.)	$2.165_{\pm 0.000}$	$0.6364_{\pm 0.0049}$

Table 8: Student test loss (S. test loss) and the stddev of teacher's mean probability (T. stddev, $\times 10^{-3}$) comparison on **ImageNet** when using different data augmentation schemes. *The S. test losses are averaged by the last 5 epochs to mitigate the random variation.*

Teacher Student Metric	resnet34 / T. stddev	resnet34 resnet18 S. test loss
KD+Identity	$0.702_{\pm 0.000}$	1.2103
KD+Flip	$0.699_{\pm 0.000}$	1.1977
KD+Crop+Flip	$0.673_{\pm 0.000}$	1.2147
KD+Cutout	$0.683_{\pm 0.000}$	1.1990
KD+AutoAugment	$0.641_{\pm 0.000}$	1.1990
KD+Mixup	$0.640_{\pm 0.001}$	1.1875
KD+CutMix	$0.619_{\pm 0.001}$	1.1809
KD+CutMixPick (S. ent.)	$0.578_{\pm 0.000}$	1.2049
KD+CutMixPick (T. ent.)	$0.578_{\pm 0.000}$	1.1955

while an important point in our theory is that we consider the *teacher's output* of the input, *not* the input per se. Namely, the proposed theory must be discussed *in the context of KD*. Existing works [22, 47] clearly are not in this scope.

Use test accuracy *vs.* test loss as the measure of student's performance. In the paper, we mentioned we use test loss instead of test accuracy as the measure of student's performance. The primary consideration, as discussed in the main paper, is that accuracy may be misaligned with loss sometimes (*i.e.*, ideally we expect *higher accuracy* coincides with *lower loss*; while in practice, we observe several counterexamples). Here we show an example of calculating the correlation between T. stddev and S. test *accuracy* in Fig. 6. As seen, if accuracy used, we observe no positive correlation between T. stddev and S. performance (which is the S. test accuracy here), while the correlation is strong if we use test loss. This shows the importance of using the *correct* measure for student's performance in a theoretical investigation of KD problems, as also noted by [29].

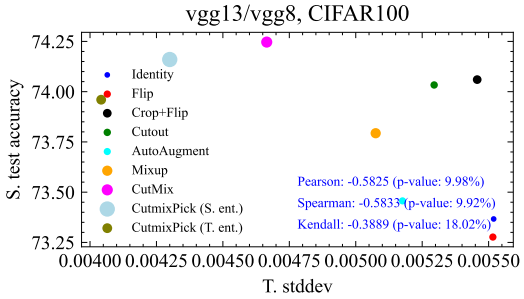
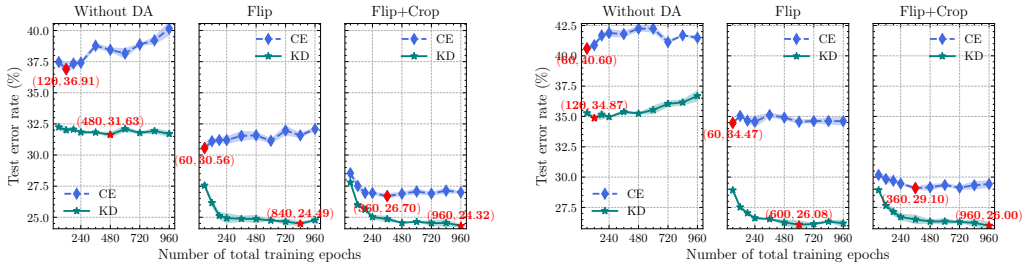


Figure 6: Showcase of using test *accuracy* instead of test *loss* as the measure of student's performance to conduct the T. stddev *vs.* S. performance correlation analysis in our paper.

KD *vs.* CE loss on wrn_40_2/wrn_16_2 and vgg13/vgg8. See Fig. 7. These plots show more examples that we can harvest considerable performance gain simply by using a strong DA with prolonged training.



(a) wrn_16_2 (teacher: wrn_40_2) (b) vgg8 (teacher: vgg13)

Figure 7: Test error rate of wrn_16_2 and vgg8 on CIFAR100 when trained for different numbers of epochs, using KD or cross-entropy (CE) loss, with or without data augmentation (DA). Every error rate is averaged by 3 random runs (shaded area indicates the stddevdev). Consistent with Fig. 2 in the main text, when DA is used, the optimal number of epochs is postponed and postponed more for KD than CE. When a stronger DA is used, the optimal number of epochs is postponed even more with smaller optimal test loss.

A.2 More Explanations from Eq. (7) to Eq. (8)

In Proposition 3.1, the main idea is to have lower covariance among samples (in terms of the teacher's outputs), as suggested by Eq. (7). Then, it seems that the most straightforward idea is to use the covariance (or possibly the normalized covariance, *i.e.*, correlation) of the teacher's output as metric to capture the dependency among samples. Why does the paper not use this idea?

We first show this idea does not work as well as our proposed metric. Then we explain why.

Given a batch of input images, the teacher's output probability is $P \in \mathbb{R}^{N\times C}$, where N is the batch size, C is the number of classes. Then we consider the covariance and correlation of P and try to construct a new metric from them:

$$\begin{aligned} \mathrm{Cov}(P,P) \in \mathbb{R}^{N\times N} &\Rightarrow v = \mathrm{Cov}(P,P).\mathrm{mean}() \in \mathbb{R}, \\ \mathrm{Cor}(P,P) \in \mathbb{R}^{N\times N} &\Rightarrow r = \mathrm{Cor}(P,P).\mathrm{mean}() \in \mathbb{R}. \end{aligned} \tag{12}$$

We collect this (mean) covariance v (a scalar) and correlation r (a scalar) over many batches and then take the average of all the collected v and r,

$$\begin{aligned} \bar{v} &= \frac{1}{K}\sum_{k\in[K]} v_k, \\ \bar{r} &= \frac{1}{K}\sum_{k\in[K]} r_k. \end{aligned} \tag{13}$$

where K is the total number of batches (10 epochs in our experiments, $K = 7,818$).

Then, we replace the proposed "T. stddev" with $\bar{v}$ and $\bar{r}$ in Tab. 3, resulting in two new tables: Tab. 9 and Tab. 10.

As seen, the foremost impression of Tab. 9 and Tab. 10 is that they have many red uparrows, which implies the metric goes *against* the supposed order, undermining its efficacy. *E.g.*, AutoAugment is stronger than Cutout (in that Autoaugment includes Cutout as a part; also AutoAugment performs better than Cutout as the test losses indicate). However, by $\bar{v}$ (see Tab. 9), AutoAugment is ranked "*weaker*" than Cutout with teacher resnet56 and vgg13.

As for Tab. 10, the correlation metric $\bar{r}$ is even worse than $\bar{v}$ in the sense that it incurs even more red uparrows than $\bar{v}$. Especially, CutMixPick (T. ent.) is the best DA (delivering the lowest test loss) while by $\bar{r}$ it "*underperforms*" all the other DA schemes except Identity and Autoaugment, with the teacher resnet56.

Table 9: Student test loss (S. test loss) and the **covariance of the teacher's output** ($\bar{v}$) comparison on CIFAR100 when using different DA schemes. This table is a replica of Tab. 3 just replacing the "T. stddev" with $\bar{v}$ here. The up (or down) arrow indicates the result is higher (or lower) than the one *right above it*. If "S. test loss" change is at a different direction from the "$\bar{v}$" change, we highlight the arrow in red.

Teacher	wrn_40_2	wrn_40_2	wrn_40_2	resnet56	resnet56	resnet56	vgg13	vgg13	vgg13
Student	/	wrn_16_2	vgg8	/	resnet20	ShuffleV2	/	vgg8	MobileNetV2
Metric	$\bar{v}$	S. test loss	S. test loss	$\bar{v}$	S. test loss	S. test loss	$\bar{v}$	S. test loss	S. test loss
KD+Identity	0.00015322	1.0976	1.1830	0.00014041	1.1783	0.9785	0.00015447	1.1856	1.6021
KD+Flip	0.00015258↓	1.0774↓	1.1673↓	0.00013956↓	1.1668↓	0.9961↑	0.00015522↑	1.1754↓	1.5779↓
KD+Flip+Crop	0.00014922↓	1.0837↑	1.1446↓	0.00013380↓	1.1763↑	0.9736↓	0.00015133↓	1.1537↓	1.5244↓
KD+Cutout	0.00014114↓	1.0564↓	1.1306↓	0.00012680↓	1.1687↓	0.9541↓	0.00014300↓	1.1279↓	1.4963↓
KD+AutoAugment	0.00013504↓	1.0305↓	1.1102↓	0.00013046↑	1.1478↓	0.9355↓	0.00014306↑	1.1273↓	1.4137↓
KD+Mixup	0.00012844↓	1.0507↑	1.0917↓	0.00011760↓	1.1621↑	0.9703↑	0.00013265↓	1.1194↓	1.4009↓
KD+CutMix	0.00011725↓	1.0310↓	1.0657↓	0.00010520↓	1.1424↓	0.9348↓	0.00011377↓	1.0728↓	1.2846↓
KD+CutMixPick (S. ent.)	0.00010184↓	1.0054↓	1.0471↓	0.00008972↓	1.1269↓	0.9339↓	0.00009727↓	1.0605↓	1.2739↓
KD+CutMixPick (T. ent.)	0.00009197↓	0.9753↓	0.9928↓	0.00007949↓	1.0853↓	0.9038↓	0.00008676↓	0.9884↓	1.1876↓

Table 10: Student test loss (S. test loss) and the **correlation coefficient of the teacher's output** ($\bar{r}$) comparison on CIFAR100 when using different DA schemes. This table is a replica of Tab. 3 just replacing the "T. stddev" with $\bar{r}$ here. The up (or down) arrow indicates the result is higher (or lower) than the one *right above it*. If "S. test loss" change is at a different direction from the "$\bar{r}$" change, we highlight the arrow in red.

Teacher	wrn_40_2	wrn_40_2	wrn_40_2	resnet56	resnet56	resnet56	vgg13	vgg13	vgg13
Student	/	wrn_16_2	vgg8	/	resnet20	ShuffleV2	/	vgg8	MobileNetV2
Metric	$\bar{r}$	S. test loss	S. test loss	$\bar{r}$	S. test loss	S. test loss	$\bar{r}$	S. test loss	S. test loss
KD+Identity	0.01570871	1.0976	1.1830	0.01570028	1.1783	0.9785	0.01562722	1.1856	1.6021
KD+Flip	0.01564524↓	1.0774↓	1.1673↓	0.01559539↓	1.1668↓	0.9961↑	0.01570224↑	1.1754↓	1.5779↓
KD+Flip+Crop	0.01559558↓	1.0837↑	1.1446↓	0.01537652↓	1.1763↑	0.9736↓	0.01559079↓	1.1537↓	1.5244↓
KD+Cutout	0.01530729↓	1.0564↓	1.1306↓	0.01510221↓	1.1687↓	0.9541↓	0.01531730↓	1.1279↓	1.4963↓
KD+AutoAugment	0.01856617↑	1.0305↓	1.1102↓	0.01971744↑	1.1478↓	0.9355↓	0.01947365↑	1.1273↓	1.4137↓
KD+Mixup	0.01497638↓	1.0507↑	1.0917↓	0.01512188↓	1.1621↑	0.9703↑	0.01507644↓	1.1194↓	1.4009↓
KD+CutMix	0.01498818↑	1.0310↓	1.0657↓	0.01502888↓	1.1424↓	0.9348↓	0.01473911↓	1.0728↓	1.2846↓
KD+CutMixPick (S. ent.)	0.01508117↑	1.0054↓	1.0471↓	0.01536088↑	1.1269↓	0.9339↓	0.01462629↓	1.0605↓	1.2739↓
KD+CutMixPick (T. ent.)	0.01525488↑	0.9753↓	0.9928↓	0.01564995↑	1.0853↓	0.9038↓	0.01487075↑	0.9884↓	1.1876↓

Why does the covariance metric $\bar{v}$ not work as well as our proposed metric? The covariance metric arises directly from our proposition, if the proposition is correct, why does it not work well? Fundamentally, the reason is that the covariance among samples is *hard* to estimate accurately. When we calculate the covariance matrix $\mathrm{Cov}(P, P)$, P is of shape $N \times C$. Note, each row of P is an *attribute (or random variable, RV)* and each column of P is an *observation*[§]. The number of observations of P is the number of classes, *which is a constant* for a given dataset (in the current case, it is 100 for the CIFAR100 dataset). Intuitively, given two random variables, in order to estimate the covariance between them, merely 100 observations is not very abundant. Namely, the limited observations render the direct estimation of the covariance among samples *inaccurate*. This can explain the counterexamples between $\bar{v}$ and test loss in Tab. 9.

(If the covariance metric is accurately estimated, it would not be surprising that the correlation in Tab. 10 is inaccurate, either. Especially, the normalization itself may not be desired. We put the results of using correlation as metric simply for a reference.)

Then, how does the proposed metric resolve this "limited observation" problem? The "smart thing" of the proposed metric is that it considers *the sum of RVs* instead of the RV itself. In statistics, it is well-known that, when multiple RVs are added together, the variance (or equivalently, stddev) of the sum RV will take into account the covariance among its members, as shown below:

$$\mathrm{Var}(\sum_{i=1}^{N} X_i) = \sum_{i=1}^{N} \mathrm{Var}(X_i) + 2 \sum_{1 \le i < j \le N} \mathrm{Cov}(X_i, X_j), \tag{14}$$

where X_i (with a little abuse of notation here) is the teacher's output probability of the i-th example.

[§]Interested readers are encouraged to check out this numpy covariance function, which we use to implement $\bar{v}$: https://numpy.org/doc/stable/reference/generated/numpy.cov.html

Essentially, we want to estimate the covariance term in RHS (*i.e.*, $\sum_{1\leq i<j\leq N}\mathrm{Cov}(X_i, X_j)$). Now, what we do in the paper is to use the LHS (*i.e.*, $\mathrm{Var}(\sum_{i=1}^{N} X_i)$) as a *proxy* of the covariance term. Clearly, there is a *hidden assumption* here to allow us to use such a proxy: the first term of RHS (*i.e.*, $\sum_{i=1}^{N}\mathrm{Var}(X_i)$) stays (nearly) the same for different samples (*i.e.*, different batches) of $\{X_i\}$. In our work, there are two conditions to make this hidden assumption hold in practice. **(1)** The augmented images are based on the original images. We can still consider them in the same domain (in our Proposition 3.1, X_i is actually assumed to be drawn from the same distribution), so $\mathrm{Var}(X_i)$ is close for different X_i. **(2)** We have abundant samples ($N = 640$ in our experiments for CIFAR100 and Tiny ImageNet). The sum effect of so many samples makes $\sum_{i=1}^{N}\mathrm{Var}(X_i)$ stabilize to a constant.

We can actually come up with counterexamples that *intentionally* break the hidden assumption. *E.g.*, consider a "bad" DA which turns all input images to a *constant* image. Then the input batch will become a repetition of just one image. When we input such a batch to the teacher, the teacher's output probability will be the same. Then the $\mathrm{Var}(\sum_{i=1}^{N} X_i)$ term will become zero. It cannot be a legitimate proxy for the covariance term anymore as the first term of RHS now is zero.

We are aware of the existence of such counterexamples. Notably, they do *not* affect the validity of the proposed metric. Here are the reasons – For one thing, such bad DA rarely appears in practice. The strong correlation per se already demonstrates the efficacy of the proposed metric in practical cases. For another thing, there is an important design in our training setup to avoid such counterexamples – Note we *add* the augmented images to the original images instead of replacing them (see Fig. 3). Therefore, in a batch, it is *unlikely* that all the images degenerate to one image. Namely, the counterexample we just mentioned never appear in practice.

Then how to understand such counterexample? Why it *appears* to go against our theory? Fundamentally, the counterexample we just mentioned breaks one of the key assumptions of our proposition in the first place – each individual sample (including the original image *and the augmented image*) obeys the same true (unknown) data distribution $\mathcal{D}$. The bad DA transforms all input images to a constant image, which already means the individual sample is not drawn from the true data distribution since randomly drawing from the true data distribution will not give us the same image.

To further confirm this, we implement the "bad" DA and re-check the validity of the proposed metric $\bar{m}$ with wrn_40_2/wrn_16_2 on CIFAR100. When using the "bad" DA, we have the metric value $\bar{m} = 0.022439$. Compare this to Tab. 3, we will notice it is ranked the *highest* (*i.e.*, the *worst*). This agrees with our expectation.

In short, the counterexamples above do not affect the efficacy of our proposed metric in practice.

A.3 Example of Calculating T. Stddev

Take CIFAR100 for an example. It has 50,000 training samples. With batch size 64, one epoch amounts to 782 batches. During training, the K in Eq. (9) is set to 10. During the iteration of training data loader, we collect the teacher's outputs. Every $K/2$ training batches (the division of 2 is because we *append* the augmented images to the original images), we obtain a matrix of shape $[K \times \mathrm{batch\_size}, \mathrm{num\_classes}]$. Then we average this matrix along the 1st axis to give us a vector, *i.e.*, the $\mathbf{u}$ in Eq. (9). In brief, every $K/2$ batches, we have one sample of $\mathbf{u}$. In total, we run the data loader for *10 epochs* (10 is empirically set; more epochs should give us more accurate estimation but we just find 10 is good enough), which gives us 7820/5=1654 $\mathbf{u}$'s, *i.e.*, a matrix of shape $[1654, \mathrm{num\_classes}]$. Then we calculate the variance along the 1st axis, giving us a vector of shape $[\mathrm{num\_classes}]$, *i.e.*, the $\mathbf{m}$ in Eq. (10). Finally, we average the stddev elements in $\mathbf{m}$ to obtain the proposed *scalar* metric.

For more details, please check our code at: http://github.com/MingSun-Tse/Good-DA-in-KD.

A.4 CutMix Sample Analysis on ImageNet

CutMix sample analysis and why KD is naturally suited to exploit CutMix. During the KD training of resnet34/resnet18 on ImageNet, we recorded the CutMix samples on which the teacher *disagrees* with the CutMix scheme on the label. We call this *label disagreement issue*.

As show in Fig. 8. there exist cases where the image cut from one image covers the salient object in the other. For example, the cab in (a) completely covers the ground beetle. In this case, using

the label by CutMix does not make sense anymore. A similar problem appears on (b). Note that these misleading labels by CutMix are rectified when the teacher is employed to guide the student. The teacher assigns the correct label "cab" to (a) and "Yorkshire terrier" to (b) (which is still not the true label "Tibetan terrier" but it is clearly more relevant and "Tibetan terrier" is also in the top-5 predictions). For (c) and (d), they pose a problem more than occlusion: the foreground cut in (c) is labeled as "acoustic guitar", however, the cut is too small for us to make it out without knowing the label. Meanwhile, the background object "Arabian camel" is occluded. Then the grids in the picture turn out to be the most salient part. If we look at the predictions of the teacher, "shopping cart" and "shopping basket" clearly make more sense than either of the original two labels. A similar issue happens on (d), where the "Indian elephant" is largely occluded. The foreground cut is labeled "quill" but the bottle in the middle is more salient. Thus the teacher predicted it as "coffeepot", "milk can", *etc*.

In order to see how severe the label disagreement issue is, we counted the number of these synthetic samples and found that on **more than half of the samples (52.1%)** produced by CutMix, the teacher model and CutMix hold a different view regarding the label. Many of these suffer from the problem shown in Fig. 8. The KD loss can rectify these label mistakes. This further shows the interplay between KD and DA: KD thrives on DA and *in turn, some DA schemes are more reasonable for KD* (than CE) where a teacher can supply more relevant labels.

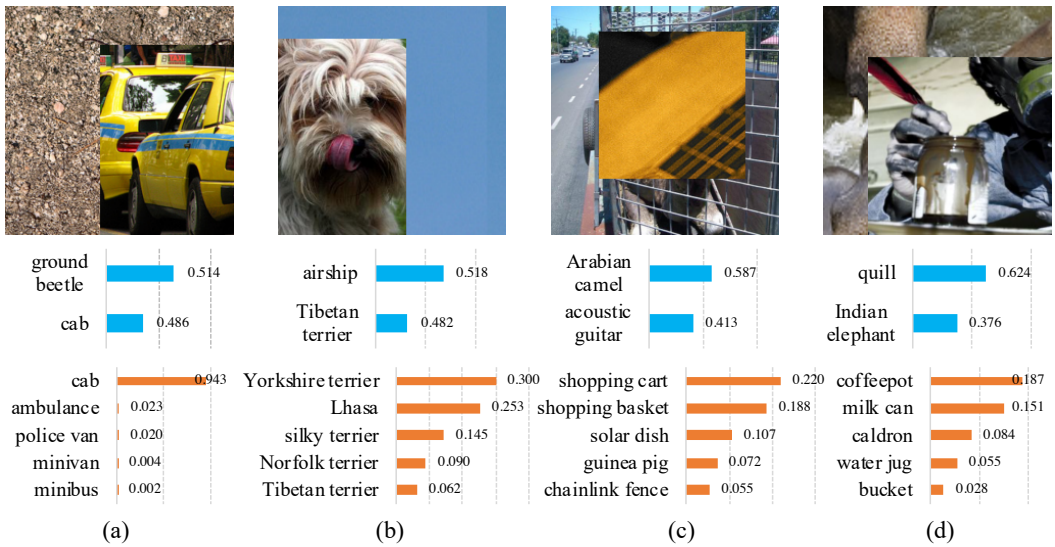


Figure 8: ImageNet CutMix samples where the main object in one of the images is no longer visible after CutMix augmentation. Below each sample, the first is the target probability assigned by CutMix and the second is the top-5 predicted probabilities by the teacher. These examples can be misleading when cross-entropy loss is used, but not for KD, as explained in the text.

A.5 Dataset License and Hardware Condition

The four datasets used in this paper (CIFAR100, Tiny ImageNet, ImageNet100, ImageNet) are all publicly available. One KD experiment with one NIVIDA 2080Ti GPU on CIFAR100 takes around 2 to 6 hrs; while an experiment on Tiny ImageNet, ImageNet100, or ImageNet may take up to 24 hrs on the 2080Ti GPU. In general, *the experimenting is not compute intensive*. For details, please refer to our GitHub code: http://github.com/MingSun-Tse/Good-DA-in-KD.

A.6 Limitations and Potential Negative Societal Impacts

Limitations. The proposed metric is calculated with finite training samples. The precision of the metric values will necessarily depend on the number of training samples used. In this regard, we have tried our best to avoid the influence of statistical variations (using abundant samples). The presented results in the paper are shown stable. Especially, the correlation in Tabs. 1 and 2 is strong. This fact itself is a proof that the limitation of our paper in this regard is small.

Potential Negative Societal Impacts. Simply put, this work proposes theories and algorithms that can boost the performance of a network with the aid of a larger network (*i.e.*, the teacher-student distillation), that is, make it smaller, faster, and possibly consume less energy in practical applications. We focus on the classification task, which is generally the foundation of the many up-stream computer vision tasks like detection and segmentation. Therefore, this work potentially has a broad application especially in the computer vision areas.

The algorithm itself has few negative societal issues, but when it makes many AI-driven technologies applicable in practice, the impact really depends on how humans use them. This actually falls into the general ethical discussion on whether AI is good or not. Beyond this scope, this work does not have specific negative societal impacts brought by its potential application, to our best knowledge.