What Makes a "Good" Data Augmentation in Knowledge Distillation – A Statistical Perspective

Anonymous Author(s)
Affiliation
Address
email

Abstract

Knowledge distillation (KD) is a general neural network training approach that uses a teacher to guide a student. Existing works mainly study KD from the network output side (*e.g.*, how 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 characterizes a "good" DA in KD? Our investigation from a statistical perspective suggests that a good DA scheme should reduce the variance of the teacher's mean probability, which will eventually lead to a lower generalization error gap for the student. Besides the theoretical understanding, we also propose a new entropy-based data-mixing DA scheme to enhance CutMix. Extensive empirical studies support our claims and demonstrate how we can harvest considerable performance gains simply by using a better DA scheme in distillation.

1 Introduction

Deep neural networks (DNNs) are the top performing method in many artificial intelligence areas [15, 27]. How to effectively train a deep network for classification has been a central topic for decades. In the past several years, efforts have mainly focused on better architecture design (*e.g.*, batch normalization [11], residual blocks [6], dense connections [10]) and better loss functions (*e.g.*, label smoothing [31, 20], contrastive loss [9], large-margin softmax [16]) than the standard cross-entropy (CE) loss. Knowledge distillation (KD) [8] is a training method that falls in 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 [14]; $\alpha \in (0, 1)$ 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 [8] 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 [2, 35, 12, 36].

Most existing KD methods have attempted to improve it by proposing better KD loss functions applied at the network outputs [24, 21, 32]. Few works have considered KD from the input side. Especially, the interplay between KD and data augmentation (DA) [29] has not been well understood so far. In this work, we ask: *What defines a "better" 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

Submitted to 36th Conference on Neural Information Processing Systems (NeurIPS 2022). Do not distribute.

us considerable performance gain – in Fig. 1, we empirically show that a stronger DA can lower the test error rate and admit more training iterations without overfitting. The figure shows test error rates using standard cross entropy loss (no teacher) as well as using knowledge distillation loss (with a teacher). It is pretty obvious to see that "Flip+Crop" is stronger than "Flip" alone in Fig. 1. However, for other DA schemes, such as Mixup [40] vs. AutoAugment [3], which one is stronger? We desire a principled way (for example, 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 better 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 CIFAR-100 and Tiny ImageNet datasets with various pairs. The proposed theory well explains why CutMix is better than other alternatives (such as Mixup [40], AutoAugment [3]) in KD.

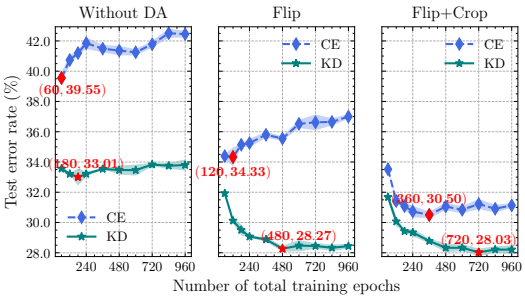

Figure 1: Test error rate of ResNet20 on CIFAR-100 when trained for different numbers of epochs with (KD) and without (CE) knowledge distillation (the teacher is ResNet56 for KD). Each result is obtained by averaging 3 random runs (shaded area indicates the std). "Flip": horizontal flip; "Crop": random crop. The optimal number of training epochs and its test error are highlighted in red.

Besides 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 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 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 output 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. The general idea of knowledge distillation is to guide the training of a student model through a stronger pretrained teacher model (or an ensemble of models). It was pioneered by Buciluă *et al.* [1] and later refined by Hinton *et al.* [8], who coined the term. Since its debut, knowledge distillation has seen extensive application in vision and language tasks [2, 35, 12, 36]. Many variants have been proposed regarding the central question in knowledge distillation, that is, how to define the *knowledge* that is supposed to be transferred from the teacher to the student. Examples of such knowledge definitions include feature distance [25], feature map attention [39], feature distribution [22], activation boundary [7], inter-sample distance relationship [21, 24, 17, 33], and mutual information [32]. 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 in this paper 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.

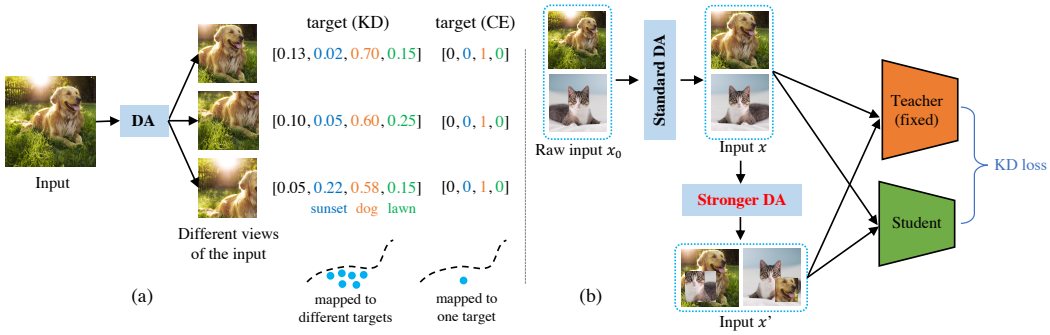

Figure 2: **(a)** Illustration of the difference of supervised target between the KD loss and cross-entropy (CE) loss. An input is transformed to different versions owing to data augmentation. KD loss can provide extra information to the student by mapping these views to different targets, while the CE loss cannot. This work attempts to answer what characterize 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.

Data Augmentation. 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 [29]. In classification tasks, data augmentation aims to explicitly provide data with label-invariant transformations (such as random crop, horizontal flip, color jittering, Cutout [5]) 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 [40] linearly mixes two images with the labels mixed by the same linear interpolation. Manifold Mixup [34] is similar to Mixup but conducts the mix operation in the feature level instead of pixel level; CutMix [37] pastes a patch cut from an image onto another image with the label decided by the area ratio of the two parts. AutoAugment [3] is a strong DA method that finds an optimal augmentation policy from a large search space that minimizes a task loss. When both the input and target are transformed simultaneously, the key is to maintain a *semantic correspondence* between the new input and new target. Although these methods have been proven effective, one lingering concern is the reasoning behind them. Specifically, it is easy to come up with examples where the semantic correspondence is poorly kept (see our supplementary material for examples of CutMix on ImageNet). 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 the proposed new data augmentation is specifically for knowledge distillation.

One recent work [4] also conducts empirical study about the impact of data augmentation on knowledge distillation. They first apply data augmentation (*e.g.*, mixup/CutMix) to the teacher then conduct the distillation step as usual (no extra data augmentation in this step); our investigation is the *exact opposite* to their setup: we train the teacher as usual (not applying mixup/CutMix), then in the distillation step we employ a more advanced data augmentation (*e.g.*, mixup/CutMix). Interestingly, they conclude that the teacher trained with mixup/CutMix *hurts* the student's generalization ability, while we consistently see a student performance boost via a stronger DA in the KD step.

The work of [19] presents a statistical perspective to understand how the softened probabilities in KD is 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 KD 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 hypothesis class $\mathcal{H}$ (C: the number of classes), which 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 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 [19]. 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 Distillation?

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 generalizes better. We make this idea rigorous as follows.

Proposition 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 elements sampled from the same distribution* $\mathcal{D}$*,* ***while not i.i.d.*** *(especially when we use data augmentation). 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),

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

The second last equation above is due to the fact that each element x_i in S is drawn the same underlying distribution $\mathcal{D}$. From the above derivation, 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,

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

where $\mathrm{Var}[\cdot,\cdot]$ stands for covariance. From the last item in Eq. (7), we can see that the covariance part is different for different sampled S. 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, $\mathrm{E}_{S_1}[(R_{S_1}(\mathbf{f}) - R_{\mathcal{D}}(\mathbf{f}))^2] > \mathrm{E}_{S_2}[(R_{S_2}(\mathbf{f}) - R_{\mathcal{D}}(\mathbf{f}))^2]$. □

Practical use: Std of the teacher's mean probability. Note in Proposition 1, the predictor $\mathbf{f}$ (*i.e.*, the student model) can be any one (not necessarily a converged model). Clearly, we want to characterize the student model *after KD training*. Then, for a well-trained student $\mathbf{f}$ with enough capacity (which is the case in practice), $\mathbf{f}(x)$ (after softmax to make a valid probability distribution) is actually pretty close to $\mathbf{p}^{(t)}(x)$. Thus, we can further reduce the covariance term in Eq. (7):

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

This formula implies an important fact that we only need the covariance of the *teacher's* probability to measure the "goodness" of a certain DA technique, no need for the student. This fact will be empirically justified in our experiments with different networks as the student (see Tabs. 1 and 2).

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) (for a *converged* student in consideration), 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 $\{\mathbf{p}^{(t)}(x_k)\}_{k=1}^K$,

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

Its variance inherently takes into account the covariance among its addends. Therefore, we can use the variance (or equivalently, its std) 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 ($K = 640$ in our experiments[1]). Note, $\mathbf{u}$ can be interpreted as the *teacher's mean probability* over K input samples.

[1] We randomly sample 10 batches during training. For CIFAR100 and Tiny ImageNet, the batchs size is 64, hence $K = 640$.

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 (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) std of the teacher's mean probability, 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 merely 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. 2(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 increased, typically 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 our supplementary material for concrete examples on ImageNet).

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: Enhance 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* [28]. 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) [28]. Since the augmented data can vary in their quality, we can introduce 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 exactly in the same form of Eq. (8) where a converged student is considered. Empirically, we will also show the data selected by this formula indeed results in lower $\bar{m}$ and better test error of 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 (see Tabs. 1 and 2). 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. Learning on these hard examples may expand the student's knowledge by reducing its blind spots. Despite this intuitively plausible explanation, in practice, we will show this scheme actually under-performs the former student-agnostic one in Eq. (11).

Table 1: KD test error ("Test err.") and the standard deviation of the teacher's mean probability ("Std of mean prob.") comparison on CIFAR-100 when using different data augmentation schemes. *All the test errors are averaged by 3 random runs*. The up (or down) arrow indicates the result is higher (or lower) than the one *right above it*. If "Test err." change is at a different direction from the "Mean prob. std" change, we highlight the arrow in red.

Teacher	WRN-40-2			ResNet56			VGG13		
Student	WRN-16-2/VGG8			ResNet20/ShuffleNetV2			VGG8/MobileNetV2		
	WRN-40-2	WRN-16-2	VGG8	ResNet56	ResNet20	ShuffleNetV2	VGG13	VGG8	MobileNetV2
	Std of mean prob.	Test err.	Test err.	Std of mean prob.	Test err.	Test err.	Std of mean prob.	Test err.	Test err.
KD+Identity	0.005474	1.0976	1.1830	0.005249	1.1783	0.9785	0.005520	1.1856	1.6021
KD+Flip	0.005473↓	1.0774↓	1.1673↓	0.005233↓	1.1668↓	0.9961↑	0.005515↓	1.1754↓	1.5779↓
KD+Flip+Crop	0.005399↓	1.0837↑	1.1446↓	0.005130↓	1.1763↑	0.9736↓	0.005443↓	1.1537↓	1.5244↓
KD+Cutout	0.005268↓	1.0564↓	1.1306↓	0.004994↓	1.1687↓	0.9541↓	0.005280↓	1.1279↓	1.4963↓
KD+AutoAugment	0.005105↓	1.0305↓	1.1102↓	0.004895↓	1.1478↓	0.9355↓	0.005159↓	1.1273↓	1.4137↓
KD+Mixup	0.004998↓	1.0507↑	1.0917↓	0.004722↓	1.1621↑	0.9703↑	0.005092↓	1.1194↓	1.4009↓
KD+CutMix	0.004720↓	1.0310↓	1.0657↓	0.004442↓	1.1424↓	0.9348↓	0.004676↓	1.0728↓	1.2846↓
KD+CutMixPick (S ent.)	0.004406↓	1.0054↓	1.0471↓	0.004035↓	1.1269↓	0.9339↓	0.004313↓	1.0605↓	1.2739↓
KD+CutMixPick (T ent.)	0.004161↓	0.9753↓	0.9928↓	0.003787↓	1.0853↓	0.9038↓	0.004043↓	0.9884↓	1.1876↓

5 Experimental Results

Datasets and Networks. We evaluate our method on the CIFAR-100 [13] and Tiny ImageNet[2] datasets. CIFAR-100 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 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 [30], ResNet [6], WRN (Wide-ResNet) [38], MobileNetV2 [26], ShuffleNetV2 [18].

Benchmark Methods. In addition to the standard cross-entropy training and the original KD method [8], we also compare with the state-of-the-art distillation approach Contrastive Representation Distillation (CRD) [32]. 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. Loss weight $\alpha = 0.9$ (Eq. (1)). For CIFAR-100 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 [32] for fair comparison. Note, in our experiments we will show 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 CIFAR-100 for 480 epochs ($k = 2$) in total, the epochs after which the learning rate is decayed will be 300, 360, and 420.

We use PyTorch [23] to conduct all our experiments. For CIFAR-100, we adopt the pretrained teacher models from CRD[3] for fair comparison with it. For Tiny ImageNet, we train our own teacher models. *Our code and trained models will be made publicly available*.

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

- **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**: Horizontal flip.
- **Flip+Crop**: 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** [5]: Cutout occludes a small random patch of an image
- **AutoAugment** [3]: AutoAugment is an ensemble of a collected DA schemes. The DA policy is automatically selected by reinforcement learning instead of manually.
- **Mixup** [40]: 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.

[2] https://tiny-imagenet.herokuapp.com/

[3] https://github.com/HobbitLong/RepDistiller

- **CutMix** [37]: CutMix cuts a small patch from one source image and pastes it to another source image, the resulted image 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 Theory

Before presenting results, one thing worth mention is that, in this section we do not use accuracy as the measure of student's performance in KD, because (1) all the formulas in Sec. 3 are derived using numerical error instead of accuracy; (2) more importantly, it is well-known that accuracy can mismatch with error – a model may achieve a better accuracy, meanwhile higher error too [19], which we also observed many times in our experiments.

In Tab. 1 (CIFAR-100) and Tab. 2 (Tiny ImageNet), we present the std of the teacher's mean probability (Eq. (10)) and the student's test error after distillation. We have the following observations.

(1) In Tab. 1, foremost, we see in terms of the std of the teacher's mean probability, Identity $<$ Flip $<$ Flip+Crop $<$ Cutout $<$ AutoAugment $<$ Mixup $<$ CutMix. These inequalities are well-aligned with our intuition. For example, AutoAugment [3] includes Cutout [5] in its transformation pool, thus should be stronger than Cutout, which is consistently reflected by the std of the teacher's mean probability on different pairs.

(2) Per our theory, lower std of the teacher's mean probability should lead to better generalization risk for the student. This is also generally well verified in Tab. 1. We see only six counterexamples (highlighted by red arrow in Tab. 1). Possible reasons are: 1) The test error is obtained on the test set with finite samples, which is only an approximation of the true error defined on distribution; 2) The teacher's mean probability is also evaluated on finite data. Despite them, the general picture from Tab. 1 still confirms the positive correlation between the std of teacher's mean probability and student's test error. The correlation is actually *very significant* as the p-values indicate (see our supplementary material for detailed p-values).

(3) The coherence between std of the teacher's mean probability and student's test error is robust. We also see it on the more challenging Tiny ImageNet, as shown in Tab. 2. Of particular note in Tab. 2 is that, Mixup is shown *weaker* than AutoAugment here (in the sense of a higher "Std of mean prob."), contrary to the CIFAR-100 case (see Tab. 1, VGG13 columns) where Mixup is stronger. Interestingly, the student's test errors also well resonate with this change: The test error of VGG8 and MobileNetV2 using Mixup is higher than that using AutoAugment. This demonstrates the efficacy of the proposed metric in Eq. (10) underpinned by Proposition 1.

Table 2: KD test error ("Test err.") and the standard deviation of the teacher's mean probability ("Std of mean prob.") comparison on Tiny ImageNet when using different data augmentation schemes. *All the results are averaged by 3 random runs*. The up (or down) arrow indicates the result is higher (or lower) than the one *right above it*. If "Test err." change is at a different direction from the "Mean prob. std" change, we highlight the arrow in red.

Teacher	VGG13		
Student	VGG8/MobileNetV2		
	VGG13	VGG8	MobileNetV2
	Std of mean prob.	Test err.	Test err.
KD+Identity	0.003873	1.8377	2.0258
KD+Flip	0.003870$\downarrow$	1.8379$\uparrow$	1.9946$\downarrow$
KD+Flip+Crop	0.003805$\downarrow$	1.8346$\downarrow$	1.9707$\downarrow$
KD+Cutout	0.003682$\downarrow$	1.8028$\downarrow$	1.9422$\downarrow$
KD+AutoAugment	0.003282$\downarrow$	1.7310$\downarrow$	1.8013$\downarrow$
KD+Mixup	0.003389$\uparrow$	1.7757$\uparrow$	1.8587$\uparrow$
KD+CutMix	0.003172$\downarrow$	1.7259$\downarrow$	1.8269$\downarrow$
KD+CutMixPick (S ent.)	0.002905$\downarrow$	1.6556$\downarrow$	1.7509$\downarrow$
KD+CutMixPick (T ent.)	0.002687$\downarrow$	1.6237$\downarrow$	1.6839$\downarrow$

(4) Importantly, note we only need the teacher to define the quality of a certain DA in KD, *no need for the student*. This is verified in Tabs. 1 and 2, as when different students are used, the coherence between std of the teacher's mean probability and student's test error is steadily maintained. Notably, this is a great advantage in practice since we can decide which DA should be used for the best performance simply using formulas (Eq. (10)) with a few network forwards, without having to fully train the students.

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)

Table 3: Student test accuracy on CIFAR-100. 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). *Since the "CutMix" and "CutMixPick" methods keep the original data, which virtually doubles the batch size, for fair comparison, here we also double the batch size for KD to keep *the same* total number of seen examples.

Teacher Student	WRN-40-2 WRN-16-2	ResNet56 ResNet20	ResNet32x4 ResNet8x4	VGG13 VGG8	VGG13 MobileNetV2	ResNet50 VGG8	ResNet32x4 ShuffleNetV2
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* [8]	75.11 (0.20)	70.78 (0.18)	73.76 (0.21)	73.56 (0.10)	68.14 (0.29)	74.23 (0.11)	74.89 (0.14)
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} [8]	75.66 (0.21)	**71.80** (0.15)	73.23 (0.10)	74.23 (0.21)	68.80 (0.11)	74.24 (0.08)	74.80 (0.23)
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 [32]	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 4: Student test accuracy on Tiny ImageNet. The subscript 480 means the total number of training epochs (default: 240). *Batch size is doubled for fair comparison akin to Tab. 3.

Teacher Student	WRN-40-2 WRN-16-2	ResNet56 ResNet20	ResNet32x4 ResNet8x4	VGG13 VGG8	VGG13 MobileNetV2	ResNet50 VGG8	ResNet32x4 ShuffleNetV2
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* [8]	58.71 (0.10)	53.60 (0.21)	55.72 (0.18)	61.62 (0.25)	59.45 (0.09)	60.42 (0.22)	66.56 (0.12)
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} [8]	59.15 (0.19)	54.10 (0.27)	55.48 (0.13)	61.78 (0.11)	59.32 (0.10)	60.16 (0.10)	65.78 (0.13)
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 [32]	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)

to fully absorb the excessive information. That is, a stronger DA may take *more* training iterations to fully exhibit its potential. This intuition is confirmed in Fig. 1 (and also two more pairs WRN-40-2/WRN-16-2 and VGG13/VGG8 in the supplementary material). Thus, in our experiments, we will also report results with prolonged training iterations for maximized performance.

Results on CIFAR-100. The results on CIFAR-100 dataset are shown in Tab. 3.

(1) Comparing the row "KD+CutMix" to "KD", we see the proposed CutMix scheme improves the accuracies of 6/7 teacher-student pairs. On the pair ResNet32x4/ShuffleNetV2, 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, 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. 3, when CRD [32], one of the SOTA KD algorithms, is armed with our proposed "CutMixPick" and more training iterations, its results can be further advanced *consistently*. This demonstrates that the proposed schemes are 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 CIFAR-100, we have results on different teacher-student pairs, shown in Tab. 4. For prolonged training, we train for 480 epochs instead of 960 to save time. Most claims on the CIFAR-100 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 pairs. 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 achieve the best performances.

We further evaluate our DA methods equipped with CRD [32], shown in the last two rows of Tab. 4. 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.

The performance boosting is simple and generic. Notably, we achieve the above performance boosting simply using the original KD loss [8], *with no bells and whistles*. This justifies one of our motivations in this paper, *i.e.*, existing KD methods [24, 21, 32] mainly improve KD at the network *output* side via better loss functions, while we propose to improve KD at the *input* end and show this path is *just as promising*. Actually, this performance boosting effect is *generic*, not limited to CRD. In the supplementary material, we present more results of applying CutMix to another 5 KD methods. *All* of the evaluated pairs see accuracy gains; half of them are even improved by more than 1% point.

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 empirical distilled risk between the teacher and student. Based on this, we propose a new metric, std of the teacher's mean probability, 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.

References

[1] Cristian Buciluă, Rich Caruana, and Alexandru Niculescu-Mizil. Model compression. In *SIGKDD*, 2006. 2

[2] Guobin Chen, Wongun Choi, Xiang Yu, Tony Han, and Manmohan Chandraker. Learning efficient object detection models with knowledge distillation. In *NeurIPS*, 2017. 1, 2

[3] Ekin D Cubuk, Barret Zoph, Dandelion Mane, Vijay Vasudevan, and Quoc V Le. Autoaugment: Learning augmentation policies from data. In *CVPR*, 2019. 2, 3, 7, 8

[4] 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. 3

[5] Terrance DeVries and Graham W Taylor. Improved regularization of convolutional neural networks with cutout. *arXiv preprint arXiv:1708.04552*, 2017. 3, 7, 8

[6] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *CVPR*, 2016. 1, 7

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

[8] Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. Distilling the knowledge in a neural network. In *NeurIPS Workshop*, 2014. 1, 2, 7, 9, 10

[9] Geoffrey E Hinton. Training products of experts by minimizing contrastive divergence. *Neural Computation*, 14(8):1771–1800, 2002. 1

[10] Gao Huang, Zhuang Liu, Laurens Van Der Maaten, and Kilian Q Weinberger. Densely connected convolutional networks. In *CVPR*, 2017. 1

[11] Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *ICML*, 2015. 1

[12] 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. 1, 2

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

[14] Solomon Kullback. *Information theory and statistics*. Courier Corporation, 1997. 1

[15] Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. Deep learning. *Nature*, 521(7553):436, 2015. 1

[16] Weiyang Liu, Yandong Wen, Zhiding Yu, and Meng Yang. Large-margin softmax loss for convolutional neural networks. In *ICML*, 2016. 1

[17] Yufan Liu, Jiajiong Cao, Bing Li, Chunfeng Yuan, Weiming Hu, Yangxi Li, and Yunqiang Duan. Knowledge distillation via instance relationship graph. In *CVPR*, 2019. 2

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

[19] Aditya K Menon, Ankit Singh Rawat, Sashank Reddi, Seungyeon Kim, and Sanjiv Kumar. A statistical perspective on distillation. In *ICML*, 2021. 3, 4, 8

[20] Rafael Müller, Simon Kornblith, and Geoffrey E Hinton. When does label smoothing help? In *NeurIPS*, 2019. 1

[21] Wonpyo Park, Dongju Kim, Yan Lu, and Minsu Cho. Relational knowledge distillation. In *CVPR*, 2019. 1, 2, 10

[22] Nikolaos Passalis and Anastasios Tefas. Learning deep representations with probabilistic knowledge transfer. In *ECCV*, 2018. 2

[23] 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

[24] 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. 1, 2, 10

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

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

[27] Jürgen Schmidhuber. Deep learning in neural networks: An overview. *Neural networks*, 61:85–117, 2015. 1

[28] Burr Settles. From theories to queries: Active learning in practice. In *AISTATS Workshop on Active Learning and Experimental Design*, 2011. 6

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

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

[31] Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jon Shlens, and Zbigniew Wojna. Rethinking the inception architecture for computer vision. In *CVPR*, 2016. 1

[32] Yonglong Tian, Dilip Krishnan, and Phillip Isola. Contrastive representation distillation. In *ICLR*, 2020. 1, 2, 7, 9, 10

[33] Frederick Tung and Greg Mori. Similarity-preserving knowledge distillation. In *CVPR*, 2019. 2

[34] 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

[35] Huan Wang, Yijun Li, Yuehai Wang, Haoji Hu, and Ming-Hsuan Yang. Collaborative distillation for ultra-resolution universal style transfer. In *CVPR*, 2020. 1, 2

[36] Lin Wang and Kuk-Jin Yoon. Knowledge distillation and student-teacher learning for visual intelligence: A review and new outlooks. *TPAMI*, 2021. 1, 2

[37] 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

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

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

[40] Hongyi Zhang, Moustapha Cisse, Yann N Dauphin, and David Lopez-Paz. mixup: Beyond empirical risk minimization. In *ICLR*, 2018. 2, 3, 7

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 supplementary material.
   (c) Did you discuss any potential negative societal impacts of your work? [Yes] See supplementary material.
   (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 supplementary material.
   (b) Did you specify all the training details (e.g., data splits, hyperparameters, how they were chosen)? [Yes] See supplementary material.
   (c) Did you report error bars (e.g., with respect to the random seed after running experiments multiple times)? [Yes] See Tab. 3, Tab. 4, and Fig. 1.
   (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 supplementary material.
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 supplementary material.
   (c) Did you include any new assets either in the supplemental material or as a URL? [No]
   (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.