

---

# On the Generalization Effects of Linear Transformations in Data Augmentation

---

Sen Wu<sup>\*1</sup> Hongyang R. Zhang<sup>\*2</sup> Gregory Valiant<sup>1</sup> Christopher Ré<sup>1</sup>

## Abstract

Data augmentation is a powerful technique to improve performance in applications such as image and text classification tasks. Yet, there is little rigorous understanding of why and how various augmentations work. In this work, we consider a family of linear transformations and study their effects on the ridge estimator in an over-parametrized linear regression setting. First, we show that transformations which preserve the labels of the data can improve estimation by enlarging the span of the training data. Second, we show that transformations which mix data can improve estimation by playing a regularization effect. Finally, we validate our theoretical insights on MNIST. Based on the insights, we propose an augmentation scheme that searches over the space of transformations by how *uncertain* the model is about the transformed data. We validate our proposed scheme on image and text datasets. For example, our method outperforms RandAugment by 1.24% on CIFAR-100 using Wide-ResNet-28-10. Furthermore, we achieve comparable accuracy to the SoTA Adversarial AutoAugment on CIFAR-10, CIFAR-100, SVHN, and ImageNet datasets.

## 1. Introduction

Data augmentation refers to the technique of enlarging the training dataset through pre-defined transformation functions. By searching over a (possibly large) space of transformations through reinforcement learning based techniques, augmentation schemes including AutoAugment (Cubuk et al., 2018) and TANDA (Ratner et al., 2017) have shown remarkable gains over various models on image classification tasks. Recent work has proposed random sampling (Cubuk et al., 2019) and Bayesian optimization (Hataya et al., 2019)

to reduce the search cost, since RL-based techniques are computationally expensive. Despite the rapid progress of these transformation search methods, precisely understanding their benefits remains a mystery because of a lack of analytic tools. In this work, we study when and why applying a family of linear transformations helps from a theoretical perspective. Building on the theory, we develop methods to improve the efficiency of transformation search procedures.

A major challenge to understand the theory behind data augmentation is how to model the large variety of transformations used in practice in an analytically tractable framework. The folklore wisdom behind data augmentation is that adding more labeled data improves generalization, i.e. the performance of the trained model on test data (Shorten & Khoshgoftaar, 2019). Clearly, the generalization effect of an augmentation scheme depends on how it transforms the data. Previous work has analyzed the effect of Gaussian augmentation (Rajput et al., 2019) and feature averaging effect of adding transformed data (Dao et al., 2019; Chen et al., 2019). However, there is still a large gap between the transformations studied in these works and the ones commonly used in augmentation schemes.

In this work, we consider linear transformations which represent a large family of image transformations. We consider three categories: (i) Label-invariant (base) transformations such as rotation and horizontal flip; (ii) Label-mixing transformations including mixup (Zhang et al., 2017; Inoue, 2018), which produces new data by randomly mixing the features of two data points (e.g. a cat and a dog) – the labels are also mixed; (iii) Compositions of (label-invariant) transformations such as random cropping and rotating followed by horizontal flipping.

To gain insight into the effects of linear transformations, we consider a conceptually simple over-parametrized model proposed in Bartlett et al. (2019); Xie et al. (2020); Hastie et al. (2019) that captures the need to add more data as in image settings. Suppose we are given  $n$  training data points  $x_1, \dots, x_n \in \mathbb{R}^p$  as  $X \in \mathbb{R}^{n \times p}$  with labels  $Y \in \mathbb{R}^n$ . In this setting, the labels obey the linear model under ground truth parameters  $\beta \in \mathbb{R}^p$ , i.e.  $Y = X\beta + \varepsilon$ , where  $\varepsilon \in \mathbb{R}^n$  denotes i.i.d. random noise with mean zero and variance  $\sigma^2$ .

Importantly, we assume that  $p > n$ , hence the span of the training data points does not include the entire space

---

<sup>\*</sup>Equal contribution <sup>1</sup>Department of Computer Science, Stanford University <sup>2</sup>Department of Statistics, The Wharton School, University of Pennsylvania. Correspondence to: all authors <{senwu, hongyang, gvaliant, chrismre}@cs.stanford.edu>.

Transformations	Example	Improvement
Label-invariant	Rotate ( $F$ )	$\frac{(\beta^\top P_X^\perp Fx)^2}{n}$
Label-mixing	Mixup	$\frac{\ X\beta\ ^2}{n^2}$
Composition	Rotate and flip ( $F_1 F_2$ )	$\frac{(\beta^\top P_X^\perp F_1 F_2 x)^2}{n}$

Table 1. Illustrative examples of our theoretical results. For label-invariant transformations and their compositions, they can reduce the estimation error at a rate proportional to the added information. Label-mixing reduces estimation error through a regularization effect.

of  $\mathbb{R}^p$ . We consider the ridge estimator  $\hat{\beta}$  with a fixed  $\ell_2$  regularization parameter and measure the estimation error of the ridge estimator by its distance to  $\beta$ .

Our first insight is that *label-invariant transformations* can *add new information* to the training data. We use our theoretical setup described above to present a precise statement. For a data point  $(x, y)$ , a label-invariant transformation  $F \in \mathbb{R}^{p \times p}$  in the regression setting produces a new data point  $Fx$  with label  $y$ . We show that by adding  $P_X^\perp Fx$  which is outside the span of the training data, we reduce the estimation error of the ridge estimator at a rate proportional to  $(\beta^\top P_X^\perp Fx)^2/n$  (see Theorem 3.1 for the result). Here  $P_X^\perp$  denotes the projection to the orthogonal space of  $X^\top X$ . In Section 4, we validate the insight for classification settings on MNIST by showing that a label-invariant transformation can indeed add new information by reducing the bias (intrinsic error score) of the model.

Our second insight is that *label-mixing transformations* can *provide a regularization effect*. In the theoretical setup, given two data points  $(x_1, y_1)$  and  $(x_2, y_2)$ , the mixup transformation with parameter  $\alpha$  sampled from the Beta distribution produces  $\alpha x_1 + (1 - \alpha)x_2$  with label  $\alpha y_1 + (1 - \alpha)y_2$  (Zhang et al., 2017). Interestingly, mixup does not *add new information* since the mixed data lies in the span of the training data. However, we show that mixup plays a regularization effect through shrinking the training data relative to the  $\ell_2$  regularization. The final result is that adding the mixup sample reduces estimation error by  $\Theta(\|X\beta\|^2/n^2)$  (see Theorem 3.3 for the result). In Section 4, we validate the insight on MNIST by showing that mixing same-class digits can reduce the variance (instability) of the model.

Finally, for *compositions of label-invariant transformations*, we can show their effect for adding new information as a corollary of the base case (see Corollary 3.4 for details and we provide the validation on MNIST in Section 4). We provide an illustration of the results in Table 1.

**Algorithmic results.** Building on our theory, we propose an uncertainty-based random sampling scheme which, among the transformed data points, picks those with the highest

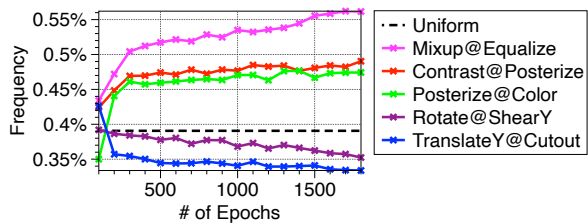


Figure 1. Our method learns and reduces the frequencies of the better performing transformations during training. Base model: PyramidNet+ShakeDrop (Han et al., 2017; Yamada et al., 2018). Dataset: CIFAR-10. See Section 5.3 for the details.

loss, i.e. “providing the most information”. We find that there is a large variance among the performance of different transformations and their compositions. Unlike RandAugment (Cubuk et al., 2019), which averages the effect of all transformations, the idea behind our sampling scheme is to better select the transformations with strong performance.

We show that our proposed scheme applies to a wide range of datasets and models. First, our sampling scheme achieves higher accuracy by finding more useful transformations compared to RandAugment on three different CNN architectures. For example, our method outperforms RandAugment by 0.59% on CIFAR-10 and 1.24% on CIFAR-100 using Wide-ResNet-28-10, and 1.54% on ImageNet using ResNet-50. Figure 1 shows the *key insight* behind our proposed method. We compare the frequencies of transformations (cf. Section 5 for their descriptions) sampled by our method. As the training procedure progresses, our method gradually learns transformations providing new information (e.g. Rotate followed by ShearY) and *reduces* their frequencies. On the other hand, the frequencies of transformations such as Mixup followed by Equalize *increase* because they produce samples with large errors that the model cannot learn.

Second, we achieve similar test accuracy on CIFAR-10 and CIFAR-100 compared to the state-of-the-art Adversarial AutoAugment (Zhang et al., 2020). By contrast, our scheme is conceptually simpler and computationally more efficient; Since our scheme does not require training an additional adversarial network, the training cost reduces by at least 5x. By further enlarging number of augmented samples by 4 times, we achieve test accuracy 85.02% on CIFAR-100, which is higher than Adversarial AutoAugment by 0.49%. Finally, as an extension, we apply our scheme on a sentiment analysis task and observe improved performance compared to the previous work of Xie et al. (2019).

**Notations.** We use the big-O notation  $f(n) \leq O(g(n))$  to indicate that  $f(n) \leq C \cdot g(n)$  for a fixed constant  $C$  and large enough  $n$ . We use  $f(n) \lesssim g(n)$  to denote that  $f(n) \leq O(g(n))$ . For a matrix  $X \in \mathbb{R}^{d_1 \times d_2}$ , let  $X^\dagger$  denote the Moore-Penrose pseudoinverse of  $X$ .

## 2. Preliminaries

Recall that  $X = [x_1^\top, \dots, x_n^\top] \in \mathbb{R}^{n \times p}$  denotes the training data, where  $x_i \in \mathbb{R}^p$  for  $1 \leq i \leq n$ . Let  $\text{Id}_p \in \mathbb{R}^{p \times p}$  denote the identity matrix. Let  $P_X$  denote the projection matrix onto the row space of  $X$ . Let  $P_X^\perp = \text{Id}_p - P_X$  denote the projection operator which is orthogonal to  $P_X$ . The ridge estimator with parameter  $\lambda < 1$  is given by

$$\hat{\beta}(X, Y) = (X^\top X + n\lambda \text{Id})^{-1} X^\top Y,$$

which arises from solving the mean squared loss

$$\frac{1}{2n} \min_{w \in \mathbb{R}^p} \|Xw - Y\|_F^2 + \frac{\lambda}{2} \|w\|^2.$$

We use  $\hat{\beta}^F$  to denote the ridge estimator when we augment  $(X, Y)$  using a transformation function  $F$ . The estimation error of  $\hat{\beta}$  is given by  $e(\hat{\beta}) := \mathbb{E}_\varepsilon [\|\hat{\beta} - \beta\|^2]$ . Next we define the label-invariance property for regression settings.

**Definition 2.1** (Label-invariance). *For a matrix  $F \in \mathbb{R}^{p \times p}$ , we say that  $F$  is label-invariant over  $\mathcal{X} \subseteq \mathbb{R}^p$  for  $\beta \in \mathbb{R}^p$  if*

$$x^\top \beta = (Fx)^\top \beta, \text{ for any } x \in \mathcal{X}.$$

As an example, consider a 3-D setting where  $\mathcal{X} = \{(a, b, 0) : \forall a, b \in \mathbb{R}\}$ . Let  $\beta = (1, -1/2, 1/2)^\top$  and

$$F = \begin{pmatrix} 1 & 0 & 0 \\ 0 & \cos \frac{\pi}{2} & \sin \frac{\pi}{2} \\ 0 & -\sin \frac{\pi}{2} & \cos \frac{\pi}{2} \end{pmatrix}.$$

Then  $(\text{Id} - F^\top)\beta = (0, 0, 1)$  is orthogonal to  $\mathcal{X}$ . Hence  $F$  is a label-preserving rotation with degree  $\frac{\pi}{2}$  over  $\mathcal{X}$  for  $\beta$ .

In addition to rotations and mixup which we have described, linear transformations are capable of modeling many image transformations. We list several examples below.

*Horizontal flip.* The horizontal flip of a vector along its center can be written as an orthonormal transformation where

$$F = \begin{pmatrix} 0 & \dots & 0 & 1 \\ 0 & \dots & 1 & 0 \\ 1 & 0 & \dots & 0 \end{pmatrix}.$$

*Additive composition.* For two transformations  $F_1$  and  $F_2$ , their additive composition gives  $x^{\text{aug}} = F_1 x + F_2 x$ . For example, *changing the color of an image and adding Gaussian noise* is an additive composition with  $F_1 x$  being a color transformation and  $F_2 x$  being a Gaussian perturbation.

*Multiplicative composition.* In this case,  $x^{\text{aug}} = F_1 F_2 x$ . For example, a *rotation followed by a cutout* of an image is a multiplicative composition with  $F_2$  being a rotation matrix and  $F_1$  being a matrix which zeros out certain regions of  $x$ .

## 3. Analyzing the Effects of Transformation Functions in an Over-parametrized Model

How should we think about the effects of applying a transformation? Suppose we have an estimator  $\hat{\beta}$  for a linear model  $\beta \in \mathbb{R}^p$ . The bias-variance decomposition of  $\hat{\beta}$  is

$$e(\hat{\beta}) = \underbrace{\left\| \mathbb{E}_\varepsilon [\hat{\beta}] - \beta \right\|^2}_{\text{bias}} + \underbrace{\left\| \hat{\beta} - \mathbb{E}_\varepsilon [\hat{\beta}] \right\|^2}_{\text{variance}} \quad (1)$$

In the context of data augmentation, we show the following two effects from applying a transformation.

*Adding new information.* The bias part measures the error of  $\hat{\beta}$  after taking the expectation of  $\varepsilon$  in  $\hat{\beta}$ . Intuitively, the bias part measures the intrinsic error of the model after taking into account the randomness which is present in  $\hat{\beta}$ . A transformation may improve the bias part if  $x^{\text{aug}}$  is outside  $P_X$ . We formalize the intuition in Section 3.1.

*Regularization.* Without adding new information, a transformation may still reduce  $e(\hat{\beta})$  by playing a regularization effect. For example with mixup,  $x^{\text{aug}}$  is in  $P_X$ . Hence adding  $x^{\text{aug}}$  does not add new information to the training data. However, the mixup sample reweights the training data and the  $\ell_2$  regularization term in the ridge estimator. We quantify the effect in Section 3.2.

### 3.1. Label-Invariant Transformations

We quantify the effect of label-invariant transformations which add new information to the training data.

**Example.** Given a training data point  $(x, y)$ , let  $(x^{\text{aug}}, y^{\text{aug}})$  denote the augmented data where  $y^{\text{aug}} = y$ . Intuitively, based on our training data  $(X, Y)$ , we can infer the label of any data point within  $P_X$  (e.g. when  $\sigma$  is sufficiently small). If  $x^{\text{aug}}$  satisfies that  $P_X x^{\text{aug}} = 0$ , then add  $(x^{\text{aug}}, y^{\text{aug}})$  does not provide new information. On the other hand if  $P_X^\perp x^{\text{aug}} \neq 0$ , then adding  $x^{\text{aug}}$  expands the subspace over which we can obtain accurate estimation. Moreover, the added direction corresponds to  $P_X^\perp x^{\text{aug}}$ . Meanwhile, since  $y^{\text{aug}} = y$ , it contains a noise part which is correlated with  $\varepsilon$ . Hence the variance of  $\hat{\beta}$  may increase.

To derive the result, we require a technical twist: instead of adding the transformed data  $Fx$  directly, we are going to add its projection onto  $P_X^\perp$ . This is without loss of generality since we can easily infer the label of  $P_X^\perp Fx$ . Let  $\hat{\beta}^F$  denote the ridge estimator after adding the augmented data to the  $(X, Y)$ . Our result is stated as follows.

**Theorem 3.1.** *Suppose we are given a set of  $n$  covariates  $X \in \mathbb{R}^{n \times p}$  with labels  $Y = X\beta + \varepsilon$ , where  $\beta \in \mathbb{R}^p$  and  $\varepsilon \in \mathbb{R}^n$  has mean 0 and variance  $\sigma^2$ . Let  $F \in \mathbb{R}^{p \times p}$  be a label-invariant transformation over  $X$  for  $\beta$ . Let  $(x, y)$  be a data point from  $(X, Y)$ .*

**Algorithm 1** Uncertainty-based sampling of transformations

---

1: **Input.** a batch of  $B$  data point  $(x_1, y_1), (x_2, y_2), \dots, (x_B, y_B)$   
**Require.**  $K$  transformations  $F_1, F_2, \dots, F_K$ , the current model  $\hat{\beta}$  and the associated loss  $l_{\hat{\beta}}$ . Default transformations  $G_1, \dots, G_H$ .  
**Param.**  $L$ : number of composition steps;  $C$ : number of augmented data per input data;  $S$ : number of selected data points used for training.  
**Return:** a list of  $B \cdot S$  transformed data  $T$ .

2: **for**  $i = 1, \dots, B$  **do**  
 3:   **for**  $k = 1, \dots, C$  **do**  
 4:     Let  $F_{j_1}, \dots, F_{j_L}$  be  $L$  transformations sampled uniformly at random without replacement from  $F_1, \dots, F_K$ .  
 5:     Compute  $x_k^{\text{aug}}$  and  $y_k^{\text{aug}}$  by applying  $F_{j_1}, \dots, F_{j_L}$  and the default ones  $G_1, \dots, G_H$  sequentially on  $(x_i, y_i)$ .  
 6:     Infer the loss of  $(x_k^{\text{aug}}, y_k^{\text{aug}})$  as  $l^k = l_{\hat{\beta}}(x_k^{\text{aug}}, y_k^{\text{aug}})$ .  
 7:   **end for**  
 8:   Choose the  $S$  data points from  $\{x_k^{\text{aug}}, y_k^{\text{aug}}\}_{k=1}^C$  that have the highest losses  $l^k$  and add them to  $T$ .  
 9: **end for**

---

Let  $z = P_X^\perp Fx$  and  $y^{\text{aug}} = y - \text{Diag}[(X^\top)^\dagger Fx] Y$ . Suppose that we augment  $(X, Y)$  with  $(z, y^{\text{aug}})$ . Then, we have that

$$e(\hat{\beta}) - e(\hat{\beta}^F) \geq \frac{(2\lambda(n+1) - (1-2\lambda)\|z\|^2)}{\lambda^2(n+1 + \|z\|^2)} \cdot \langle z, \beta \rangle^2 - \frac{(2 + 2 \frac{\|P_X Fx\|^2}{\mu_{\min}(X)^2})}{\lambda^2(n+1)^2} \cdot \sigma^2 \|z\|^2. \quad (2)$$

Moreover, when  $\|z\|^2 = o(n)$  and  $\frac{\langle z, \beta \rangle^2}{\|z\|^2} \geq \log n(1 + \frac{\|P_X Fx\|^2}{\mu_{\min}(X)^2}) \frac{\sigma^2}{\lambda n}$  (including  $\sigma = 0$ ), we have

$$0 \leq e(\hat{\beta}) - e(\hat{\beta}^F) - (2 + o(1)) \frac{\langle z, \beta \rangle^2}{\lambda n} \leq \frac{\text{poly}(\gamma/\lambda)}{n^2}. \quad (3)$$

In the above,  $\mu_{\min}(X)$  denotes the smallest singular value of  $X$ . For a vector  $v$ ,  $\text{Diag}[v] \in \mathbb{R}^{d \times d}$  denotes a diagonal matrix with the  $i$ -th diagonal entry being the  $i$ -th entry of  $v$ .  $\text{poly}(\gamma/\lambda)$  denotes a polynomial of  $\gamma/\lambda$ .

Theorem 3.1 shows that the reduction of estimation error scales with  $\langle z, \beta \rangle^2$ , the correlation between the new signal and the true model. Intuitively, as long as  $n$  is large enough, then equation (3) will hold. The proof is by carefully comparing the bias and variance of  $\hat{\beta}$  and  $\hat{\beta}^{\text{aug}}$  after adding the augmented data point. On the other hand, we remark that adding  $Fx$  directly into  $X$  does not always reduce  $e(\hat{\beta})$ , even when  $\langle z, \beta \rangle^2 = \Theta(1)$  (cf. (Xie et al., 2020)).

Another remark is that for augmenting a sequence of data points, one can repeated apply Theorem 3.1 to get the result. We leave the details to Appendix A.1.

**Connection to augmentation sampling schemes.** We derive a corollary for the idea of random sampling used in RandAugment. Let  $\{F_i\}_{i=1}^K$  be a set of  $K$  label-invariant transformations. We consider the effect of randomly sampling a transformation from  $\{F_i\}_{i=1}^K$ .

**Corollary 3.2.** *In the setting of Theorem 3.1, let  $\{F_i\}_{i=1}^K$  be  $K$  label-invariant transformations. For a data point*

$(x, y)$  from  $(X, Y)$  and  $i = 1, \dots, K$ , let  $z_i = P_X^\perp F_i x$  and  $y_i^{\text{aug}} = y - \text{Diag}[(X^\top)^\dagger F_i x] Y$ .

Suppose that  $(z, y^{\text{aug}})$  is chosen uniformly at random from  $\{z_i, y_i^{\text{aug}}\}_{i=1}^K$ . Then we have that

$$\begin{aligned} & \mathbb{E}_{z, y^{\text{aug}}} \left[ e(\hat{\beta}) - e(\hat{\beta}^{\text{unif}\{F_i\}_{i=1}^K}) \right] \\ &= \frac{2 + o(1)}{K} \left( \sum_{i=1}^K \frac{\langle z_i, \beta \rangle^2}{\lambda n} \right) + \frac{\text{poly}(\gamma/\lambda)}{n^2}. \end{aligned}$$

The proof follows directly from Theorem 3.1. Corollary 3.2 implies that the effect of random sampling is simply an average over all the transformations. However, if there is a large variance among the effects of the  $K$  transformations, random sampling could be sub-optimal.

**Our proposed scheme: uncertainty-based sampling.**

Our idea is to use the sampled transformations more efficiently via an *uncertainty-based* sampling scheme. For each data point, we randomly sample  $C$  (compositions of) transformations. We pick the ones with the highest losses after applying the transformation. This is consistent with the intuition of Theorem 3.1. The larger  $\langle z, \beta \rangle^2$ , the higher the loss of  $(x^{\text{aug}}, y^{\text{aug}})$  would be under  $\hat{\beta}$ .

Algorithm 1 describes our procedure in detail. In Line 4-6, we compute the losses of  $C$  augmented data points. In Line 8, we select the  $S$  data points with the highest losses for training. For each batch, the algorithm returns  $S\hat{B}$  augmented samples.

**3.2. Label-Mixing Transformations: Mixup**

We show that mixup plays a regularization effect through reweighting the training data and the  $\ell_2$  regularization term.

Specifically, we analyze the following procedure. Let  $\alpha \in [0, 1]$  be sampled from a Beta distribution with fixed parameters (Zhang et al., 2017). Let  $(x_i, y_i)$  and  $(x_j, y_j)$  be two data points selected uniformly at random from the train-

ing data. We add the mixup sample  $(x^{\text{aug}}, y^{\text{aug}})$  into  $(X, Y)$ , with  $x^{\text{aug}} = \alpha x_i + (1 - \alpha)x_j$  and  $y^{\text{aug}} = \alpha y_i + (1 - \alpha)y_j$ .

We illustrate that adding the mixup sample is akin to shrinking the training data relative to the regularization term. Assuming that  $\sum_{i=1}^n x_i = 0$ , we have

$$\begin{aligned} \mathbb{E}_{x^{\text{aug}}} [x^{\text{aug}} x^{\text{aug}\top}] &= \frac{(1 - 2\alpha)^2}{n} \cdot X^\top X \\ \Rightarrow \mathbb{E}_{x^{\text{aug}}} [X^\top X + x^{\text{aug}} x^{\text{aug}\top} + (n + 1)\lambda \text{Id}] \\ &= \left(1 + \frac{(1 - 2\alpha)^2}{n}\right) X^\top X + (n + 1)\lambda \text{Id}, \end{aligned}$$

Hence the mixup sample shrinks the  $X^\top X$  term relative to the  $\ell_2$  regularization term in  $\hat{\beta}^{\text{aug}}$ ! Below we describe our result formally.

**Theorem 3.3.** *Let  $\gamma > 1$  be a fixed constant which does not grow with  $n$ . Let  $\{x_k\}_{k=1}^n$  be  $n$  training samples which satisfy that  $\sum_{k=1}^n x_k = 0$  and  $\|x_k\| \leq \gamma$ , for all  $1 \leq k \leq n$ . Assume that  $n$  is large enough (e.g.  $n \gtrsim \frac{(\gamma + \lambda)^3}{\lambda^4} \max(\|\beta\|^2 \gamma^4, \sigma^2 \gamma^2)$ ). In expectation over the randomness of  $\alpha, x_i, x_j$ , we have that*

$$\mathbb{E}_{\alpha, x_i, x_j} [e(\hat{\beta}) - e(\hat{\beta}^{\text{mixup}})] \gtrsim \frac{\lambda^2 \|X\beta\|^2}{\gamma^3 n^2}.$$

We remark that the assumption that  $\sum_{i=1}^n x_i = 0$  is indeed satisfied in image classification settings. This is because a normalization step, which normalizes the mean of every RGB channel to be zero, is applied on all the images. The intuition behind the proof of Theorem 3.3 is that the mixup sample shrinks the training data, which reduces the bias of the estimator. The proof can be found in Appendix A.2.

### 3.3. Compositions of Label-Invariant Transformations

Our theory can also be applied to quantify the amount of new information added by compositions of transformations. We first describe an example to show that taking compositions expands the search space of transformation functions.

**Example.** Consider two transformations  $F_1$  and  $F_2$ , e.g. a rotation and a horizontal flip. Suppose we are interested in finding a transformed sample  $(x^{\text{aug}}, y^{\text{aug}})$  such that adding the sample reduces the estimation error of  $\hat{\beta}^{\text{aug}}$  the most. With additive compositions, the search space for  $x^{\text{aug}}$  becomes

$$\{Fx : x \in \mathcal{X}, F \in \{F_1, F_2, F_1 + F_2\}\},$$

which is strictly a superset compared to using  $F_1$  and  $F_2$ .

Based on Theorem 3.1, we can derive a simple corollary which quantifies the incremental benefit of additively composing a new transformation.

**Corollary 3.4.** *In the setting of Theorem 3.1, let  $F_1, F_2$  be two label-invariant transformations. For a data point  $(x, y)$  from  $(X, Y)$  and  $i \in \{1, 2\}$ , let  $z_i = P_X^\dagger F_i x$  and  $y_i^{\text{aug}} = y - \text{Diag}[(X^\top)^\dagger F_i x] Y$ . The benefit of composing  $F_2$  with  $F_1$  is given as follows*

$$\begin{aligned} e(\hat{\beta}^{F_1}) - e(\hat{\beta}^{F_1 + F_2}) &= (2 + o(1)) \frac{\langle z_1, \beta \rangle^2 - \langle z_1 + z_2, \beta \rangle^2}{\lambda n} \\ &\quad + \frac{\text{poly}(\gamma/\lambda)}{n^2}. \end{aligned}$$

Corollary 3.4 implies that the effect of composing  $F_2$  with  $F_1$  may either be better or worse than applying  $F_1$ . This is consistent with our experimental observation (described in Section 4). We defer the proof of Corollary 3.4 to Appendix A.3. We remark that the example and the corollary also apply to multiplicative compositions.

## 4. Measuring the Effects of Transformation Functions

We validate the theoretical insights from Section 3 on MNIST (LeCun et al., 1998). To extend our results from the regression setting to the classification setting, we propose two metrics that correspond to the bias and the variance of a linear model. Our idea is to decompose the average prediction accuracy (over all test samples) into two parts similar to equation (1), including an *intrinsic error score* which is deterministic and an *instability score* which varies because of randomness.

We show three claims: i) Label-invariant transformations such as rotations can add new information by reducing the intrinsic error score. ii) As we increase the fraction of same-class mixup digits, the instability score decreases. iii) Composing multiple transformations can either increase the accuracy (and intrinsic error score) or decrease it. Further, we show how to select a core set of transformations.

**Metrics.** We train  $k$  independent samples of multi-layer perceptron with hidden layer dimension 100. Let  $\hat{\beta}_i$  denote the predictor of the  $i$ -th sample, for  $1 \leq i \leq k$ . For each data point  $x$ , let  $M(x)$  denote the majority label in  $\{\hat{\beta}_i\}_{i=1}^k$  (we break ties randomly). Clearly, the majority label is the best estimator one could get, given the  $k$  independent predictors, without extra information. Then, we define the *intrinsic error score* as

$$\frac{1}{n} \cdot \left( \sum_{(x,y) \in (X,Y)} \mathbb{1}\{M(x) \neq y\} \right). \quad (4)$$

We define the *instability score* as

$$\frac{1}{n} \cdot \left( \sum_{(x,y) \in (X,Y)} \left( \frac{1}{k} \cdot \sum_{i=1}^k \mathbb{1}\{\hat{\beta}_i(x) \neq M(x)\} \right) \right). \quad (5)$$

	Avg. Acc.	Error Score	Instab. Score
Baseline	98.08%	1.52%	0.95%
Cutout	98.31%	1.43%	0.86%
RandCrop	98.61%	<b>1.01%</b>	0.88%
Rotation	<b>98.65%</b>	1.08%	<b>0.77%</b>

Table 2. Measuring the intrinsic error and instability scores of individual transformations on MNIST. The three transformations all reduce the intrinsic error score compared to the baseline.

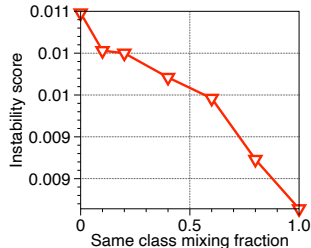


Figure 2. The instability score decreases as we increase the fraction of same-class mixup digits on MNIST.

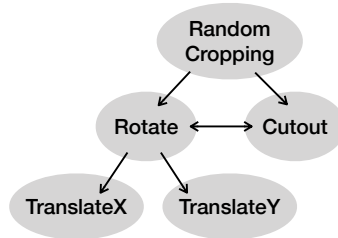


Figure 3. Visualizing which transformations are beneficial for each other. On MNIST, the translations do not provide additional benefit beyond the rest three transformations.

Compared to equation (1), the first score corresponds to the bias of  $\hat{\beta}$ , which can change if we add new information to the training data. The second score corresponds to the variance of  $\hat{\beta}$ , which measures the stability of the predicted outcomes in the classification setting. For the experiments we sample 9 random seeds, i.e. set  $k = 9$ .

**Label-invariant transformations.** Recall that Section 3.1 shows that label-invariant transformations can reduce the bias of a linear model by adding new information to the training data. Correspondingly, our hypothesis is that the intrinsic error score should decrease after augmenting the training dataset with suitable transformations.

Table 2 shows the result for applying rotation, random cropping and the cutout (i.e. cutting out a piece of the image) respectively. The baseline corresponds to using no transformations. We observe that the three transformations all reduce the model error score and the instability score, which confirms our hypothesis. There are also two second-order effects: i) A higher instability score hurts the average prediction accuracy of random cropping compared to rotation. ii) A higher intrinsic error score hurts the average prediction accuracy of cutout compared to random cropping.

To further validate that rotated data “adds new information”, we show that the intrinsic error score decreases by correcting data points that are “mostly incorrectly” predicted by the baseline model. Among the data points that are predicted correctly after applying the rotation but are wrong on the baseline model, the average accuracy of these data points on the baseline model is 25%. In other words, for 75% of the time, the baseline model makes the wrong prediction for these data. We leave the details to Appendix B.

**Label-mixing transformations.** Section 3.2 shows that in our regression setting, adding a mixup sample has a regularization effect. What does it imply for classification problems? We note that our regression setting comprises a single parameter class  $\beta$ . For MNIST, there are 10 different classes of digits. Therefore, a plausible hypothesis

following our result is that mixing same-class digits has a regularization effect.

We conduct the following experiment to verify the hypothesis. We vary the fraction of mixup data from mixing same-class vs. different-class digits. We expect that as we increase the same class mixing fraction, the instability score decreases. The results of figure 2 confirm our hypothesis. We also observe that mixing different class images behaves differently compared to mixing same class images. The details are left to Appendix D.

**Compositions of transformations.** Recall that Section 3.3 shows the effect of composing two transformations may be positive or negative. We describe examples to confirm the claim. By composing rotation with random cropping, we observe an accuracy of 98.72% with intrinsic error score 0.86%, which are both lower than rotation and random cropping individually. A negative example is that composing translating the X and Y coordinate (acc. 98.0%) is worse than translating X (acc. 98.38%) or Y (acc. 98.19%) individually. Same for the other two scores.

*Identifying core transformations.* As an in-depth study, we select five transformation including rotation, random cropping, cutout, translation of the x-axis and the y-axis. These transformations can all change the geometric position of the digits in the image. Our goal is to identify a subset of core transformations from the five transformations. For example, since the rotation transformation implicitly changes the x and the y axis of the image, a natural question is whether composing translations with rotation helps or not.

To visualize the results, we construct a directed graph with the five transformations. We compose two transformations to measure their model error scores. An edge from transformation A to transformation B means that applying B after A reduces the model error score by over 10%. Figure 3 shows the results. We observe that rotation, random cropping and cutout can all help some other transformations, whereas the translations do not provide an additional benefit for the rest.

## 5. Experiments

We test our uncertainty-based transformation sampling scheme on both image and text classification tasks. First, our sampling scheme achieves more accurate results by finding more useful transformations compared to RandAugment. Second, we achieve comparable test accuracy to the SoTA Adversarial AutoAugment on CIFAR-10, CIFAR-100, and ImageNet with less training cost because our method is conceptually simpler. Finally, we evaluate our scheme in text augmentations to help train a sentiment analysis model.

### 5.1. Experimental Setup

**Datasets and models.** We consider the following datasets and models in our experiments.

*CIFAR-10 and CIFAR-100:* The two datasets are colored images with 10 and 100 classes, respectively. We evaluate our proposed method for classifying images using the following models : Wide-ResNet-28-10 (Zagoruyko & Komodakis, 2016), Shake-Shake (26 2x96d) (Gastaldi, 2017), and PyramidNet+ShakeDrop (Han et al., 2017; Yamada et al., 2018).

*Street view house numbers (SVHN):* This dataset contains color house-number images with 73,257 core images for training and 26,032 digits for testing. We use Wide-ResNet-28-10 model for classifying these images.

*ImageNet Large-Scale Visual Recognition Challenge (ImageNet):* This dataset includes images of 1000 classes, and has a training set with roughly 1.3M images, and a validation set with 50,000 images. We select ResNet-50 (He et al., 2016) to evaluate our method.

**Comparison methods.** For image classification tasks, we compare Algorithm 1 with AutoAugment (AA) (Cubuk et al., 2018), Fast AutoAugment (Fast AA) (Lim et al., 2019), Population Based Augmentation (PBA) (Ho et al., 2019), RandAugment (RA) (Cubuk et al., 2019), and Adversarial AutoAugment (Adv. AA) (Zhang et al., 2020). We also include the baseline model with the following default setting. For CIFAR-10 and CIFAR-100, we flip each image horizontally with probability 0.5 and then randomly crop a  $32 \times 32$  sub-image from the padded image. For SVHN, we apply the cutout to every image. For ImageNet, we randomly resize and crop a  $224 \times 224$  sub-image from the original image and then flip the image horizontally with probability 0.5.

**Training procedures.** Recall that Algorithm 1 contains three parameters, how many composition steps ( $L$ ) we take, how many augmented data points ( $C$ ) we generate and how many ( $S$ ) we select for training. We set  $L = 2$ ,  $C = 4$  and  $S = 1$  for our experiments on CIFAR datasets and SVHN. We set  $L = 2$ ,  $C = 8$  and  $S = 4$  for our experiments on ImageNet. We consider  $K = 16$  transforma-

tions in Algorithm 1, including AutoContrast, Brightness, Color, Contrast, Cutout, Equalize, Invert, Mixup, Posterize, Rotate, Sharpness, ShearX, ShearY, Solarize, TranslateX, TranslateY. See e.g. (Shorten & Khoshgoftaar, 2019) for descriptions of these transformations.

As is common in previous work, we also include a parameter to set the probability of applying a transformation in Line 4 of Algorithm 1. We set this parameter and the magnitude of each transformation randomly in a suitable range. We apply the augmentations over the entire training dataset. We report the results averaged over four random seeds.

### 5.2. Experimental Results

We apply Algorithm 1 on three image classification tasks (CIFAR10, CIFAR100, SVHN, and ImageNet) over several models. Table 3 summarizes the result. We highlight the comparisons to RandAugment and Adversarial AutoAugment since they dominate the other benchmark methods.

#### Improving classification accuracy over RandAugment.

For Wide-ResNet-28-10, we find that our method outperforms RandAugment by 0.59% on CIFAR-10 and 1.24% on CIFAR-100. If we do not use Mixup, our method still outperforms RandAugment by 0.45% on CIFAR-10. For Shake-Shake and PyramidNet+ShakeDrop, our method improves the accuracy of RandAugment by 0.27% and 0.16% on CIFAR-10, respectively. For ResNet-50 on ImageNet dataset, our method achieves top-1 accuracy 79.14% which outperforms RandAugment by 1.54%.

#### Improving training efficiency over Adversarial AutoAugment.

Our method achieves comparable accuracy to the current state-of-the-art on CIFAR-10, CIFAR-100, and ImageNet. Algorithm 1 uses additional inference cost to find the uncertain samples. And we estimate that the additional cost equals half of the training cost. However, the inference cost is 5x cheaper compared to training the adversarial network of Adversarial AutoAugment, which requires generating 8 times more samples for training.

#### Further improvement by increasing the number of augmented samples.

Recall that Algorithm 1 contains a parameter  $S$  which controls how many new labeled data we generate per training data. The results in Table 3 use  $C = 4$  and  $S = 1$ , but we can further boost the prediction accuracy by increasing  $C$  and  $S$ . In Table 4, we find that by setting  $C = 8$  and  $S = 4$ , our method improves the accuracy of Adversarial AutoAugment on CIFAR-100 by 0.49% on Wide-ResNet-28-10.

### 5.3. Ablation Studies

**Histogram of selected transformations.** We examine the transformations selected by Algorithm 1 to better understand its difference compared to RandAugment. For this

## On the Generalization Effects of Linear Transformations in Data Augmentation

Dataset	Model	Baseline	AA	Fast AA	PBA	RA	Adv. AA	Ours
CIFAR-10	Wide-ResNet-28-10	96.13	97.32	97.30	97.42	97.30	<b>98.10</b>	97.89%(±0.03%)
	Shake-Shake (26 2x96d)	97.14	98.01	98.00	97.97	98.00	98.15	<b>98.27%</b> (±0.05%)
	PyramidNet+ShakeDrop	97.33	98.52	98.30	98.54	98.50	98.64	<b>98.66%</b> (±0.02%)
CIFAR-100	Wide-ResNet-28-10	81.20	82.91	82.70	83.27	83.30	84.51	<b>84.54%</b> (±0.09%)
SVHN	Wide-ResNet-28-10	96.9	98.1	-	-	<b>98.3</b>	-	<b>98.3%</b> (±0.03%)
ImageNet	ResNet-50	76.31	77.63	77.60	-	77.60	<b>79.40</b>	79.14%

Table 3. Test accuracy (%) on CIFAR-10, CIFAR-100, and SVHN. We compare our method with default data augmentation (Baseline), AutoAugment (AA), Fast AutoAugment (Fast AA), Population Based Augmentation (PBA), RandAugment (RA), and Adversarial AutoAugment (Adv. AA). Our results are averaged over four random seeds except ImageNet experiments.

Dataset	Adv. AA	Ours ( $S = 4$ )
CIFAR-10	98.10(±0.15%)	<b>98.16%</b> (±0.05%)
CIFAR-100	84.51(±0.18%)	<b>85.02%</b> (±0.18%)

Figure 4. Increasing the number of augmented data points per training sample can further improve accuracy.

purpose, we measure the frequency of transformations selected by Algorithm 1 every 100 epoch. If a transformation generates useful samples, then the model should learn from these samples. And the loss of these transformed data will decrease as a result. On the other hand, if a transformation generates bad samples that are difficult to learn, then the loss of these bad samples will remain large.

Figure 1 shows the sampling frequency of five compositions. We test the five compositions on a vanilla Wide-ResNet model. For transformations whose frequencies are decreasing, we get: Rotate and ShearY, 93.41%; TranslateY and Cutout, 93.88%. For transformations whose frequencies are increasing, we get: Posterize and Color, 89.12% (the results for other two are similar and we omit the details). Hence the results confirm that Algorithm 1 learns and reduces the frequencies of the better performing transformations.

### 5.4. Extension to Text Augmentations

While we have focused on image augmentations throughout the paper, we can also our ideas to text augmentations. We extend Algorithm 1 to a sentiment analysis task as follows. We choose BERT<sub>LARGE</sub> as the baseline model, which is a 24 layer transformer network from (Devlin et al., 2018). We apply our method to three augmentations: back-translation (Yu et al., 2018), switchout (Wang et al., 2018), and word replace (Xie et al., 2019).

*Dataset.* We use the Internet movie database (IMDb) with 50,000 movie reviews. The goal is to predict whether the sentiment of the review is positive or negative.

*Comparison methods.* We compare with pre-BERT SoTA, BERT<sub>LARGE</sub>, and unsupervised data augmentation

	RA	Adv. AA	Ours ( $S = 1$ )
Training ( $\times$ )	1.0	8.0	$\sim 1.5$

Figure 5. Comparing the training cost between our method, RA and Adv. AA on CIFAR-10 relative to RA. The training cost of Adv. AA is cited from the authors (Zhang et al., 2020).

(UDA) (Xie et al., 2019). UDA uses BERT<sub>LARGE</sub> initialization and training on 20 supervised examples and DBPeDia (Lehmann et al., 2015) as an unsupervised source.

*Results.* We find that our method achieves test accuracy 95.96%, which outperforms all the other methods by at least 0.28%. For reference, the result of using Pre-BERT SoTA is 95.68%. The result of using BERT<sub>LARGE</sub> is 95.22%. The result of using UDA is 95.22%.

## 6. Related Work

**Image augmentations.** We describe a brief summary and refer interested readers to the excellent survey by (Shorten & Khoshgoftaar, 2019) for complete references.

Data augmentation has become a standard practice in computer vision such as image classification tasks. First, individual transformations such as horizontal flip and mixup have shown improvement over strong vanilla models. Beyond individual transformations, one approach to search for compositions of transformations is to train generative adversarial networks to generate new images as a form of data augmentation (Sixt et al., 2018; Laine & Aila, 2016; Odena, 2016; Gao et al., 2018). Another approach is to use reinforcement learning based search methods (Hu et al., 2019). The work of Cubuk et al. (2018) searches for the augmentation schemes on a small surrogate dataset. While this idea reduces the search cost, it was shown that using a small surrogate dataset results in sub-optimal augmentation policies (Cubuk et al., 2019).

The work of Kuchnik & Smith (2018) is closely related to ours since they also experiment with the idea of uncertainty-



based sampling. Their goal is different from our work in that they use this idea to find a representative sub-sample of the training dataset that can still preserve the performance of applying augmentation policies. Recently, the idea of mixing data has been applied to semi-supervised learning by mixing feature representations as opposed to the input data (Berthelot et al., 2019).

**Theoretical studies.** Dao et al. (2019) propose a kernel theory to show that label-invariant augmentations are equivalent to transforming the kernel matrix in a way that incorporates the prior of the transformation. Chen et al. (2019) use group theory to show that incorporating the label-invariant property into an empirical risk minimization framework reduces variance.

Our theoretical setup is related to Xie et al. (2020); Raghunathan et al. (2020), with several major differences. First, in our setting, we assume that the label of an augmented data is generated from the training data, which is deterministic. In their setting, the label of an augmented data includes new information because the random noise is freshly drawn. Second, we consider the ridge estimator as opposed to the minimum norm estimator, since the ridge estimator includes an  $\ell_2$  regularization which is commonly used in practice. Finally, it would be interesting to extend our theoretical setup beyond linear settings (e.g. (Li et al., 2018; Zhang et al., 2019; Montanari et al., 2019)).

## 7. Conclusions and Future Work

In this work, we studied the theory of data augmentation in a simplified over-parametrized linear setting that captures the need to add more labeled data as in image settings, where there are more parameters than the number of data points. Despite the simplicity of the setting, we have shown three novel insights into three categories of transformations. We verified our theoretical insights on MNIST. And we proposed an uncertainty-based sampling scheme which outperforms random sampling. We hope that our work can spur more interest in developing a better understanding of data augmentation methods. Below, we outline several questions that our theory cannot yet explain.

First, one interesting future direction is to further uncover the mysterious role of mixup (Zhang et al.17). Our work has taken the first step by showing the connection between mixup and regularization. Meanwhile, Table 7 (in Appendix) shows that mixup can reduce the model error score (bias) on CIFAR-10. Our theory does not explain this phenomenon because our setup implicit assumes a single class (the linear model  $\beta$ ) for all data. We believe that extending our work to a more sophisticated setting, e.g. mixed linear regression model, is an interesting direction to explain the working of mixup augmentation over multiple classes.

Second, it would be interesting to consider the theoretical benefit of our proposed uncertainty-based sampling algorithm compared to random sampling, which can help tighten the connection between our theory and the proposed algorithm. We would like to remark that addressing this question likely requires extending the models and tools that we have developed in this work. Specifically, one challenge is how to come up with a data model that will satisfy the label-invariance property for a large family of linear transformations. We leave these questions for future work.

**Acknowledgements.** We gratefully acknowledge the support of DARPA under Nos. FA86501827865 (SDH) and FA86501827882 (ASED); NIH under No. U54EB020405 (Mobilize), NSF under Nos. CCF1763315 (Beyond Sparsity), CCF1563078 (Volume to Velocity), and 1937301 (RTML); ONR under No. N000141712266 (Unifying Weak Supervision); the Moore Foundation, NXP, Xilinx, LETI-CEA, Intel, IBM, Microsoft, NEC, Toshiba, TSMC, ARM, Hitachi, BASF, Accenture, Ericsson, Qualcomm, Analog Devices, the Okawa Foundation, American Family Insurance, Google Cloud, Swiss Re, the HAI-AWS Cloud Credits for Research program, and members of the Stanford DAWN project: Teradata, Facebook, Google, Ant Financial, NEC, VMWare, and Infosys. The U.S. Government is authorized to reproduce and distribute reprints for Governmental purposes notwithstanding any copyright notation thereon. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views, policies, or endorsements, either expressed or implied, of DARPA, NIH, ONR, or the U.S. Government.

Gregory Valiant’s contributions were supported by NSF awards 1804222, 1813049 and 1704417, DOE award DE-SC0019205 and an ONR Young Investigator Award. Our experiments are partly run on Stanford University’s SOAL cluster hosted in the Department of Management Science and Engineering (cf. <https://sharad.com/soal-cluster/>).

## References

- Bartlett, P. L., Long, P. M., Lugosi, G., and Tsigler, A. Benign overfitting in linear regression. *arXiv preprint arXiv:1906.11300*, 2019.
- Berthelot, D., Carlini, N., Goodfellow, I., Papernot, N., Oliver, A., and Raffel, C. A. Mixmatch: A holistic approach to semi-supervised learning. In *Advances in Neural Information Processing Systems*, pp. 5050–5060, 2019.
- Chen, S., Dobriban, E., and Lee, J. H. Invariance reduces variance: Understanding data augmentation in deep learning and beyond. *arXiv preprint arXiv:1907.10905*, 2019.

- Cubuk, E. D., Zoph, B., Mane, D., Vasudevan, V., and Le, Q. V. Autoaugment: Learning augmentation policies from data. *arXiv preprint arXiv:1805.09501*, 2018.
- Cubuk, E. D., Zoph, B., Shlens, J., and Le, Q. V. Randaugment: Practical data augmentation with no separate search. *arXiv preprint arXiv:1909.13719*, 2019.
- Dao, T., Gu, A., Ratner, A. J., Smith, V., De Sa, C., and Ré, C. A kernel theory of modern data augmentation. *Proceedings of machine learning research*, 97:1528, 2019.
- Devlin, J., Chang, M.-W., Lee, K., and Toutanova, K. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018.
- Friedman, J., Hastie, T., and Tibshirani, R. *The elements of statistical learning*, volume 1. Springer series in statistics New York, 2001.
- Gao, H., Shou, Z., Zareian, A., Zhang, H., and Chang, S.-F. Low-shot learning via covariance-preserving adversarial augmentation networks. In *Advances in Neural Information Processing Systems*, pp. 975–985, 2018.
- Gastaldi, X. Shake-shake regularization. *arXiv preprint arXiv:1705.07485*, 2017.
- Han, D., Kim, J., and Kim, J. Deep pyramidal residual networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 5927–5935, 2017.
- Hastie, T., Tibshirani, R., and Friedman, J. *The elements of statistical learning: data mining, inference, and prediction*. Springer Science & Business Media, 2009.
- Hastie, T., Montanari, A., Rosset, S., and Tibshirani, R. J. Surprises in high-dimensional ridgeless least squares interpolation. *arXiv preprint arXiv:1903.08560*, 2019.
- Hataya, R., Zdenek, J., Yoshizoe, K., and Nakayama, H. Faster autoaugment: Learning augmentation strategies using backpropagation. *arXiv preprint arXiv:1911.06987*, 2019.
- He, K., Zhang, X., Ren, S., and Sun, J. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 770–778, 2016.
- Ho, D., Liang, E., Stoica, I., Abbeel, P., and Chen, X. Population based augmentation: Efficient learning of augmentation policy schedules. *arXiv preprint arXiv:1905.05393*, 2019.
- Hu, Z., Tan, B., Salakhutdinov, R. R., Mitchell, T. M., and Xing, E. P. Learning data manipulation for augmentation and weighting. In *Advances in Neural Information Processing Systems*, pp. 15738–15749, 2019.
- Inoue, H. Data augmentation by pairing samples for images classification. *arXiv preprint arXiv:1801.02929*, 2018.
- Krizhevsky, A., Hinton, G., et al. Learning multiple layers of features from tiny images. 2009.
- Kuchnik, M. and Smith, V. Efficient augmentation via data subsampling. *arXiv preprint arXiv:1810.05222*, 2018.
- Laine, S. and Aila, T. Temporal ensembling for semi-supervised learning. *arXiv preprint arXiv:1610.02242*, 2016.
- LeCun, Y., Bottou, L., Bengio, Y., and Haffner, P. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.
- Lehmann, J., Isele, R., Jakob, M., Jentzsch, A., Kontokostas, D., Mendes, P. N., Hellmann, S., Morsey, M., Van Kleef, P., Auer, S., et al. Dbpedia—a large-scale, multilingual knowledge base extracted from wikipedia. *Semantic Web*, 6(2):167–195, 2015.
- Li, Y., Ma, T., and Zhang, H. Algorithmic regularization in over-parameterized matrix sensing and neural networks with quadratic activations. In *Conference On Learning Theory*, pp. 2–47, 2018.
- Lim, S., Kim, I., Kim, T., Kim, C., and Kim, S. Fast autoaugment. In *Advances in Neural Information Processing Systems*, pp. 6665–6675, 2019.
- Maas, A. L., Daly, R. E., Pham, P. T., Huang, D., Ng, A. Y., and Potts, C. Learning word vectors for sentiment analysis. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, 2011.
- Montanari, A., Ruan, F., Sohn, Y., and Yan, J. The generalization error of max-margin linear classifiers: High-dimensional asymptotics in the overparametrized regime. *arXiv preprint arXiv:1911.01544*, 2019.
- Odena, A. Semi-supervised learning with generative adversarial networks. *arXiv preprint arXiv:1606.01583*, 2016.
- Raghunathan, A., Xie, S. M., Yang, F., Duchi, J., and Liang, P. Understanding and mitigating the tradeoff between robustness and accuracy. *arXiv preprint arXiv:2002.10716*, 2020.
- Rajput, S., Feng, Z., Charles, Z., Loh, P.-L., and Papailiopoulos, D. Does data augmentation lead to positive margin? *arXiv preprint arXiv:1905.03177*, 2019.

- Ratner, A. J., Ehrenberg, H., Hussain, Z., Dunnmon, J., and Ré, C. Learning to compose domain-specific transformations for data augmentation. In *Advances in neural information processing systems*, pp. 3236–3246, 2017.
- Riedel, K. S. A sherman–morrison–woodbury identity for rank augmenting matrices with application to centering. *SIAM Journal on Matrix Analysis and Applications*, 13(2):659–662, 1992.
- Shorten, C. and Khoshgoftaar, T. M. A survey on image data augmentation for deep learning. *Journal of Big Data*, 6(1):60, 2019.
- Sixt, L., Wild, B., and Landgraf, T. Rendergan: Generating realistic labeled data. *Frontiers in Robotics and AI*, 5:66, 2018.
- Wang, X., Pham, H., Dai, Z., and Neubig, G. Switchout: an efficient data augmentation algorithm for neural machine translation. *arXiv preprint arXiv:1808.07512*, 2018.
- Xie, Q., Dai, Z., Hovy, E., Luong, M.-T., and Le, Q. V. Unsupervised data augmentation for consistency training. *arXiv preprint arXiv:1904.12848*, 2019.
- Xie, S. M., Raghunathan, A., Yang, F., Duchi, J. C., and Liang, P. When covariate-shifted data augmentation increases test error and how to fix it, 2020.
- Yamada, Y., Iwamura, M., Akiba, T., and Kise, K. Shake-drop regularization for deep residual learning. *arXiv preprint arXiv:1802.02375*, 2018.
- Yu, A. W., Dohan, D., Luong, M.-T., Zhao, R., Chen, K., Norouzi, M., and Le, Q. V. Qanet: Combining local convolution with global self-attention for reading comprehension. *arXiv preprint arXiv:1804.09541*, 2018.
- Zagoruyko, S. and Komodakis, N. Wide residual networks. *arXiv preprint arXiv:1605.07146*, 2016.
- Zhang, H., Cisse, M., Dauphin, Y. N., and Lopez-Paz, D. mixup: Beyond empirical risk minimization. *arXiv preprint arXiv:1710.09412*, 2017.
- Zhang, H., Sharan, V., Charikar, M., and Liang, Y. Recovery guarantees for quadratic tensors with limited observations. In *International Conference on Artificial Intelligence and Statistics (AISTATS)*, 2019.
- Zhang, X., Wang, Q., Zhang, J., and Zhong, Z. Adversarial autoaugment. In *International Conference on Learning Representations*, 2020.

**Organization of the Appendix.** In Appendix A, we fill in the proofs for our main results in Section 3. We also discuss about extensions of our results to augmenting multiple data points and interesting future directions. In Appendix B, we complement Section 4 with a full list of evaluations on label-invariant transformations. In Appendix C, we describe the implementation and the experimental procedures in more detail. In Appendix D, we provide further experimental results on why mixup helps improve performance in image classification tasks.

## A. Supplementary Materials for Section 3

**Notations.** For a matrix  $X \in \mathbb{R}^{d_1 \times d_2}$ , we use  $\mu_{\max}(X)$  to denote its largest singular value. Let  $X_{i,j}$  denote its  $(i, j)$ -th entry, for  $1 \leq i \leq d_1$  and  $1 \leq j \leq d_2$ . For a diagonal matrix  $D \in \mathbb{R}^{d \times d}$  where  $D_{i,i} > 0$  for all  $1 \leq i \leq d$ , we use  $\frac{1}{D} \in \mathbb{R}^{d \times d}$  to denote its inverse. We use big-O notation  $g(n) = O(f(n))$  to denote a function  $g(n) \leq C \cdot f(n)$  for a fixed constant  $C$  as  $n$  grows large. The notation  $o(1)$  denotes a function which converges to zero as  $n$  goes to infinity. Let  $a \lesssim b$  denote that  $a \leq C \cdot b$  for a fixed constant  $C$ .

**The Sherman-Morrison formula.** Suppose  $X$  is an invertible square matrix and  $u, v \in \mathbb{R}^d$  are column vectors. Then  $X + uv^\top$  is invertible if and only if  $1 + u^\top X v$  is not zero. In this case,

$$(X + uv^\top)^{-1} = X^{-1} - \frac{A^{-1}uv^\top A^{-1}}{1 + u^\top A^{-1}v}. \quad (6)$$

### A.1. Proof of Theorem 3.1

We describe the following lemma, which tracks the change of bias and variance for augmenting  $(X, Y)$  with  $(z, y)$ .

**Lemma A.1.** *Let  $X^\top X = UDU^\top$  denote the singular vector decomposition of  $X^\top X$ . Denote by  $\varepsilon^{\text{aug}} = y^{\text{aug}} - z^\top \beta$ . In the setting of Theorem 3.1, we have that*

$$\text{bias}(\hat{\beta}) - \text{bias}(\hat{\beta}_{\text{aug}}) \geq \frac{(2\lambda(n+1) - (1-2\lambda)\|z\|^2)\langle z, \beta \rangle^2}{\lambda^2(n+1 + \|z\|^2)^2} \quad (7)$$

$$\begin{aligned} \text{var}(\hat{\beta}) - \text{var}(\hat{\beta}_{\text{aug}}) &\geq 2 \mathbb{E}_\varepsilon \left[ \langle z \varepsilon^{\text{aug}}, (X^{\text{aug}\top} X^{\text{aug}} + (n+1)\lambda \text{Id})^{-1} X^\top \varepsilon \rangle \right] \\ &\quad - \mathbb{E}_\varepsilon \left[ \varepsilon^{\text{aug}2} \right] \left\| (X^{\text{aug}\top} X^{\text{aug}} + (n+1)\lambda \text{Id})^{-1} z \right\|^2 \end{aligned} \quad (8)$$

*Proof.* The reduction of bias is given by

$$\begin{aligned} \text{bias}(\hat{\beta}) - \text{bias}(\hat{\beta}_{\text{aug}}) &= \left\| (X^\top X + n\lambda \text{Id})^{-1} X^\top X \beta - \beta \right\|^2 - \left\| (X^{\text{aug}\top} X^{\text{aug}} + (n+1)\lambda \text{Id})^{-1} X^{\text{aug}\top} X^{\text{aug}} \beta - \beta \right\|^2 \\ &= \lambda^2 \left( \left\| \left( \frac{X^\top X}{n} + \lambda \text{Id} \right)^{-1} \beta \right\|^2 - \left\| \left( \frac{X^{\text{aug}\top} X^{\text{aug}}}{n+1} + \lambda \text{Id} \right)^{-1} \beta \right\|^2 \right). \end{aligned} \quad (9)$$

Denote by  $K_{n+1} = \left( \frac{X^\top X}{n+1} + \lambda \text{Id} \right)^{-1}$  and  $K_n = \left( \frac{X^\top X}{n} + \lambda \text{Id} \right)^{-1}$ . We observe that

$$\begin{aligned} \left( \frac{X^{\text{aug}\top} X^{\text{aug}}}{n+1} + \lambda \text{Id} \right)^{-1} &= \left( \frac{X^\top X}{n+1} + \lambda \text{Id} + \frac{zz^\top}{n+1} \right)^{-1} \\ &= K_{n+1} - \frac{\frac{1}{n+1} \cdot K_{n+1} z z^\top K_{n+1}}{1 + \frac{1}{n+1} z^\top K_{n+1} z} \end{aligned} \quad (10)$$

$$= K_{n+1} - \frac{\frac{1}{n+1} \frac{1}{\lambda^2} z z^\top}{1 + \frac{\|z\|^2}{\lambda^2(n+1)}} \quad (\text{because } P_X z = 0)$$

$$= K_n - U \Delta U^\top - \frac{\frac{1}{(n+1)\lambda^2} z z^\top}{1 + \frac{\|z\|^2}{n+1}}, \quad (11)$$

where we use  $\Delta \in \mathbb{R}^{n \times n}$  to denote a diagonal matrix with the  $i$ -th diagonal entry being  $\frac{1}{n(n+1)} \frac{D_{i,i}}{(D_{i,i}/n+\lambda)(D_{i,i}/(n+1)+\lambda)}$ , i.e.  $\Delta = (\frac{1}{\frac{D}{n}+\lambda} - \frac{1}{\frac{D}{n+1}+\lambda})$ . From this we infer that  $\text{bias}(\hat{\beta}) - \text{bias}(\hat{\beta}^{\text{aug}})$  is equal to

$$\begin{aligned} & \lambda^2 \left( 2\langle K_n \beta, U \Delta U^\top + \frac{1}{1 + \frac{\|z\|^2}{n+1}} \frac{(n+1)\lambda^2 z z^\top}{n+1} \beta \rangle - \left\| \left( U \Delta U^\top + \frac{1}{1 + \frac{\|z\|^2}{n+1}} \frac{(n+1)\lambda^2 z z^\top}{n+1} \right) \beta \right\|^2 \right) \\ &= \frac{2\langle z, \beta \rangle^2}{\lambda(n+1 + \|z\|^2)} + \lambda^2 \left( 2\langle U \frac{1}{D+\lambda} U^\top \beta, U \Delta U^\top \beta \rangle - \|U \Delta U^\top \beta\|^2 \right) - \frac{\|z\|^2 \langle z, \beta \rangle^2}{\lambda^2(n+1 + \|z\|^2)^2}, \end{aligned}$$

where we again use the fact that  $U^\top z = 0$ . We note that for large enough  $n$ ,  $\frac{1}{D+\lambda} \Delta$  strictly dominates  $\Delta^2$ . Hence we conclude that

$$\text{bias}(\hat{\beta}) - \text{bias}(\hat{\beta}^{\text{aug}}) \geq \frac{(2\lambda(n+1) - (1-2\lambda)\|z\|^2)\langle z, \beta \rangle^2}{\lambda^2(n+1 + \|z\|^2)^2}, \quad (12)$$

which proves equation (7). For the change of variance, We have that

$$\begin{aligned} \text{var}(\hat{\beta}) - \text{var}(\hat{\beta}^{\text{aug}}) &= \mathbb{E}_\varepsilon \left[ \left\| (X^\top X + n\lambda \text{Id})^{-1} X^\top \varepsilon \right\|^2 \right] - \mathbb{E}_\varepsilon \left[ \left\| (X^{\text{aug}\top} X^{\text{aug}} + (n+1)\lambda \text{Id})^{-1} X^{\text{aug}\top} \varepsilon^{\text{aug}} \right\|^2 \right] \\ &= \sigma^2 \cdot \text{Tr} \left[ \left( (X^\top X + n\lambda \text{Id})^{-2} - (X^{\text{aug}\top} X^{\text{aug}} + (n+1)\lambda \text{Id})^{-2} \right) X^\top X \right] \\ &\quad + 2 \mathbb{E}_\varepsilon \left[ \langle z \varepsilon^{\text{aug}}, (X^{\text{aug}\top} X^{\text{aug}} + (n+1)\lambda \text{Id})^{-1} X^\top \varepsilon \rangle \right] - \mathbb{E}_\varepsilon \left[ \varepsilon^{\text{aug}2} \right] \left\| (X^{\text{aug}\top} X^{\text{aug}} + (n+1)\lambda \text{Id})^{-1} z \right\|^2 \\ &\geq 2 \mathbb{E}_\varepsilon \left[ \langle z \varepsilon^{\text{aug}}, (X^{\text{aug}\top} X^{\text{aug}} + (n+1)\lambda \text{Id})^{-1} X^\top \varepsilon \rangle \right] - \mathbb{E}_\varepsilon \left[ \varepsilon^{\text{aug}2} \right] \left\| (X^{\text{aug}\top} X^{\text{aug}} + (n+1)\lambda \text{Id})^{-1} z \right\|^2. \end{aligned} \quad (13)$$

Hence equation (8) is proved by noting that equation (13) is positive. This is because  $(X^\top X + n\lambda \text{Id})^{-2} - (X^{\text{aug}\top} X^{\text{aug}} + (n+1)\lambda \text{Id})^{-2}$  is PSD. And the trace of the product of two PSD matrices is always positive.  $\square$

Based on Lemma A.1, we can prove Theorem 3.1.

*Proof of Theorem 3.1.* We note that

$$e(\hat{\beta}) - e(\hat{\beta}^{\text{aug}}) = \text{bias}(\hat{\beta}) - \text{bias}(\hat{\beta}^{\text{aug}}) + \text{var}(\hat{\beta}) - \text{var}(\hat{\beta}^{\text{aug}}).$$

The change of bias has been given by equation (7). For equation (8), we note that  $z$  is orthogonal to  $(X^{\text{aug}\top} X^{\text{aug}} + \lambda \text{Id})^{-1} X^\top$ . Hence the first term is zero. The second terms simplifies to

$$-\frac{\|z\|^2}{\lambda^2(n+1)^2} \mathbb{E}_\varepsilon \left[ \varepsilon^{\text{aug}2} \right]. \quad (14)$$

For  $\varepsilon^{\text{aug}}$ , denote by  $\varepsilon_y = y - x^\top \beta$ , we note that  $\varepsilon^{\text{aug}}$  is equal to  $\varepsilon_y - \text{Diag}[(X^\top)^\dagger F x] \varepsilon$ , Hence

$$\begin{aligned} \mathbb{E}_\varepsilon \left[ \varepsilon^{\text{aug}2} \right] &\leq 2 \mathbb{E}_\varepsilon \left[ \varepsilon_y^2 \right] + 2 \mathbb{E}_\varepsilon \left[ \varepsilon^\top \text{Diag}[(X^\top)^\dagger (x - z)]^2 \varepsilon \right] \\ &= 2\sigma^2 + 2\sigma^2 \text{Diag}[(X^\top)^\dagger F x]^2 \\ &\leq 2\sigma^2 \left( 1 + \frac{\|P_X F x\|^2}{\mu_{\min}(X)^2} \right). \end{aligned}$$

Combined with equation (14) and equation (7), we have proved equation (2).

For equation (3), by plugging in the assumption that  $\|z\|^2 = o(n)$  and  $\frac{\langle z, \beta \rangle^2}{\|z\|^2} \geq (\log n) \cdot (1 + \frac{\|P_X F x\|^2}{\mu_{\min}(X)^2}) \sigma^2 / (\lambda n)$  into equation (2), we have that

$$e(\hat{\beta}) - e(\hat{\beta}^{\text{aug}}) \geq \frac{2\lambda(n+1) - (1-2\lambda)\|z\|^2 - 10 \log n}{\lambda^2(n+1 + \|z\|^2)} \langle z, \beta \rangle^2 \geq 2(1 + o(1)) \frac{\langle z, \beta \rangle^2}{\lambda n}.$$

For the other side of equation (3), from the calculation of equation (12), we have that

$$\text{bias}(\hat{\beta}) - \text{bias}(\hat{\beta}^{\text{aug}}) \leq \frac{(2\lambda(n+1) - (1-2\lambda)\|z\|^2)\langle z, \beta \rangle^2}{\lambda^2(n+1 + \|z\|^2)} + O\left(\frac{\text{poly}(\gamma, 1/\lambda)}{n^2}\right)$$

For the change of variance, from the calculation of equation (13), we have

$$\begin{aligned} \text{var}(\hat{\beta}) - \text{var}(\hat{\beta}^{\text{aug}}) &\leq \sigma^2 \cdot \text{Tr} \left[ \left( (X^\top X + n\lambda \text{Id})^{-2} - (X^{\text{aug}\top} X^{\text{aug}} + (n+1)\lambda \text{Id})^{-2} \right) X^\top X \right] \\ &\leq \sigma^2 \cdot \text{Tr} \left[ \left( (X^\top X + n\lambda \text{Id})^{-2} - (X^\top X + (n+1)\lambda \text{Id})^{-2} \right) X^\top X \right] + O\left(\frac{\sigma^2 \cdot \text{poly}(\gamma, 1/\lambda)}{n^2}\right) \\ &\leq \frac{\sigma^2 \cdot \text{poly}(\gamma, 1/\lambda)}{n^2} \end{aligned}$$

Therefore, we have shown that

$$e(\hat{\beta}) - e(\hat{\beta}^{\text{aug}}) \leq 2(1 + o(1)) \frac{\langle z, \beta \rangle^2}{\lambda n} + O\left(\frac{\text{poly}(\gamma, 1/\lambda)}{n^2}\right).$$

Hence the proof of the theorem is complete.  $\square$

**Extension to Adding Multiple Data Points.** We can also extend Theorem 3.1 to the case of adding multiple transformed data points. The idea is to repeatedly apply Theorem 3.1 and equation (3). We describe the result as a corollary as follows.

**Corollary A.2.** *In the setting of Theorem 3.1. Let  $(x_1^{\text{aug}}, y_1^{\text{aug}}), (x_2^{\text{aug}}, y_2^{\text{aug}}), \dots, (x_t^{\text{aug}}, y_t^{\text{aug}})$  denote a sequence of data points obtained via label-invariant transformations from  $(X, Y)$ . Denote by  $(X_0, Y_0) = (X, Y)$ . For  $i = 1, \dots, t$ , let  $z_i = P_{X_{i-1}}^\perp x_i^{\text{aug}}$  and  $y_i^{\text{aug}'} = y_i^{\text{aug}} - \text{Diag}[(X_i^\top)^\dagger x_i^{\text{aug}}] Y_i$ . Suppose we augment  $(X_{i-1}, Y_{i-1})$  with  $(z_i, y_i^{\text{aug}'})$ . Let  $(X_i, Y_i) \in (\mathbb{R}^{(n+i) \times p}, \mathbb{R}^{n+i})$  denote the augmented dataset.*

*Suppose that  $\lambda < 1/4$  is a small fixed constant. Let  $t = O(n)$  and  $\|z_i\|^2 = o(n)$ , for  $i = 1, \dots, t$ . Under the assumption that  $\frac{\langle z_i, \beta \rangle^2}{\|z_i\|^2} \gtrsim (1 + \frac{\|P_{X_i} x_i^{\text{aug}}\|^2}{\mu_{\min}(X_i)}) \sigma^2 / (\lambda(n+i))$ , we have*

$$e(\hat{\beta}(X, Y)) - e(\hat{\beta}(X_t, Y_t)) \gtrsim \sum_{i=1}^t \frac{\langle z_i, \beta \rangle^2}{\lambda n}.$$

The proof of the above result follows by repeatedly applying Theorem 3.1. The details are omitted.

## A.2. Proof of Theorem 3.3

Similar to the proof of Theorem 3.1, we also track the change of the bias and the variance of  $\hat{\beta}$ . The difference is that for the mixup transformation,  $x^{\text{aug}}$  lies in the row span of  $X$ . The following Lemma tracks the change of bias and variance in this setting.

**Lemma A.3.** *Let  $\varepsilon^{\text{mixup}} = \alpha \varepsilon_i + (1 - \alpha) \varepsilon_j$ . In the setting of Theorem 3.3, we have that*

$$\text{bias}(\hat{\beta}) - \text{bias}(\hat{\beta}^{\text{mixup}}) \geq \frac{2\lambda^2}{n+1} \cdot \beta^\top U \left( \frac{D}{n} + \lambda \right)^{-1} \left( \frac{D}{n+1} + \lambda \right)^{-1} U^\top x^{\text{aug}} x^{\text{aug}\top} U \left( \frac{D}{n+1} + \lambda \right)^{-1} U^\top \beta - \frac{3\|\beta\|^2 \gamma^4}{\lambda^2(n+1)^2} \quad (15)$$

$$\text{var}(\hat{\beta}) - \text{var}(\hat{\beta}^{\text{mixup}}) \geq -3\sigma^2 \cdot \text{Tr} \left[ (X^\top X + n\lambda \text{Id})^{-1} x^{\text{aug}} x^{\text{aug}\top} (X^\top X + n\lambda \text{Id})^{-1} \right] \quad (16)$$

*Proof of Lemma A.3.* Denote by  $K_n = (X^\top X + \lambda \text{Id})^{-1}$ ,  $K_{n+1} = (X^\top X + \lambda \cdot \text{Id})^{-1}$ ,  $C = \frac{1}{n+1} x^{\text{aug}\top} K_{n+1} x^{\text{aug}}$  and  $Z = \frac{K_{n+1} x^{\text{aug}} x^{\text{aug}\top} K_{n+1}}{1+C}$ . We note that  $K_n = U \left( \frac{D}{n} + \lambda \right)^{-1} U^\top + \frac{1}{\lambda} P_X^\perp$  and  $K_{n+1} = U \left( \frac{D}{n+1} + \lambda \right)^{-1} U^\top + \frac{1}{\lambda} P_X^\perp$ . Recall that  $\Delta \in \mathbb{R}^{n \times n}$  is a diagonal matrix with the  $i$ -th entry being  $\frac{D_{i,i}}{(D_{i,i}/n + \lambda)(D_{i,i}/(n+1) + \lambda)}$ .

From equation (9) and (10), we have that the change of bias is equal to

$$\begin{aligned} \text{bias}(\hat{\beta}) - \text{bias}(\hat{\beta}^{\text{mixup}}) &= 2\lambda^2 \langle (K_n \beta, \left( \frac{U \Delta U^\top}{n(n+1)} + \frac{Z}{n+1} \right) \beta) \rangle - \lambda^2 \left\| \left( \frac{U \Delta U^\top}{n(n+1)} + \frac{Z}{n+1} \right) \beta \right\|^2 \\ &= \lambda^2 \beta^\top U \left( 2 \left( \frac{D}{n} + \lambda \right)^{-1} \frac{\Delta}{n(n+1)} - \frac{\Delta^2}{n^2(n+1)^2} - 2 \frac{\frac{1}{\frac{D}{n+1} + \lambda} U^\top x^{\text{aug}} x^{\text{aug}^\top} U \frac{1}{\frac{D}{n+1} + \lambda}}{n(n+1)^2(1+C)} \right) U^\top \beta \end{aligned} \quad (17)$$

$$+ \frac{2\lambda^2}{n+1} \langle K_n \beta, Z \beta \rangle - \frac{\lambda^2}{(n+1)^2} \|Z \beta\|^2 \quad (18)$$

For equation (17), we have used the fact that  $P_X^\perp z = 0$ . We observe that for large enough  $n$ , equation (17) is always positive, since the negative terms are both on the order of  $O(\frac{1}{n^3})$ . Hence we focus on equation (18). First, we observe that

$$\begin{aligned} \langle K_n \beta, Z \beta \rangle &= \beta^\top K_n K_{n+1} x^{\text{aug}} x^{\text{aug}^\top} K_{n+1} \beta - \frac{C}{C+1} \langle K_n \beta, K_{n+1} x^{\text{aug}} x^{\text{aug}^\top} K_{n+1} \beta \rangle \\ &= \beta^\top U \left( \frac{D}{n} + \lambda \right)^{-1} \left( \frac{D}{n+1} + \lambda \right)^{-1} U^\top x^{\text{aug}} x^{\text{aug}^\top} U \left( \frac{D}{n+1} + \lambda \right)^{-1} U^\top \beta - \frac{C}{C+1} \langle K_n \beta, K_{n+1} x^{\text{aug}} x^{\text{aug}^\top} K_{n+1} \beta \rangle. \end{aligned}$$

We observe that

$$\begin{aligned} \frac{C}{C+1} \left| \langle K_n \beta, K_{n+1} x^{\text{aug}} x^{\text{aug}^\top} K_{n+1} \beta \rangle \right| &\leq \frac{1}{n+1} \|\beta\|^2 \|x^{\text{aug}}\|^4 \frac{1}{\lambda^4} \leq \frac{1}{n+1} \frac{\|\beta\|^2 \gamma^4}{\lambda^4} \\ \|Z \beta\| &\leq \frac{1}{\lambda^2} \cdot \|\beta\| \cdot \|x^{\text{aug}}\|^2 \leq \frac{\|\beta\| \gamma^2}{\lambda^2}. \end{aligned}$$

Hence (15) is proved.

Next, denote by  $K_{\text{aug}} = (X^{\text{aug}^\top} X^{\text{aug}} + \lambda \text{Id})^{-1}$ . We have that the change of variance is given as

$$\begin{aligned} \text{var}(\hat{\beta}) - \text{var}(\hat{\beta}^{\text{mixup}}) &= \sigma^2 \cdot \text{Tr} \left[ \frac{K_n}{n} X^\top X \frac{K_n}{n} \right] - \mathbb{E} \left[ \text{Tr} \left[ K_{\text{aug}} X^{\text{aug}^\top} \varepsilon^{\text{aug}} \varepsilon^{\text{aug}^\top} X^{\text{aug}} K_{\text{aug}} \right] \right] \\ &\geq \sigma^2 \cdot \text{Tr} \left[ \frac{K_n}{n} X^\top X \frac{K_n}{n} \right] - \mathbb{E} \left[ \text{Tr} \left[ K X^{\text{aug}^\top} \varepsilon^{\text{aug}} \varepsilon^{\text{aug}^\top} X^{\text{aug}} K \right] \right], \end{aligned}$$

because  $K^2 - K_{\text{aug}}^2$  is PSD and the trace of the product of two PSD matrices is always positive. Next, recall that  $\varepsilon^{\text{aug}} = \alpha \varepsilon_i + (1 - \alpha) \varepsilon_j$ , we have

$$\begin{aligned} \mathbb{E} \left[ X^{\text{aug}^\top} \varepsilon^{\text{aug}} \varepsilon^{\text{aug}^\top} X^{\text{aug}} \right] &= \sigma^2 \cdot \left( (\alpha^2 + (1 - \alpha)^2) \cdot z z^\top + 2\alpha x^{\text{aug}} x_i^\top + 2(1 - \alpha) z x_j^\top + X^\top X \right) \\ &= \sigma^2 \cdot \left( (\alpha^2 + (1 - \alpha)^2 + 2) \cdot x^{\text{aug}} x^{\text{aug}^\top} + X^\top X \right). \end{aligned} \quad (19)$$

Hence equation (16) is proved.  $\square$

Based on Lemma A.3, we can finish the proof of Theorem 3.3.

*Proof of Theorem 3.3.* We recall that  $z = \alpha x_i + (1 - \alpha) x_j$ , where  $x_i, x_j$  are sampled uniformly at random from  $X$ . Therefore,

$$\begin{aligned} \mathbb{E}_{x_i, x_j} \left[ x^{\text{aug}} x^{\text{aug}^\top} \right] &= \frac{\alpha^2 + (1 - \alpha)^2}{n} X^\top X + \frac{2\alpha(1 - \alpha)}{n} \sum_{1 \leq i \leq n} x_i \left( \sum_{j \neq i} x_j^\top \right) \\ &= \frac{\alpha^2 + (1 - \alpha)^2}{n} X^\top X + \frac{2\alpha(1 - \alpha)}{n} \left( \left( \sum_{i=1}^n x_i \right) \left( \sum_{i=1}^n x_i^\top \right) - X^\top X \right) \\ &= \frac{(1 - 2\alpha)^2}{n} X^\top X, \end{aligned}$$

where in the last line we used the assumption that  $\sum_{i=1}^n x_i = 0$ . For  $\alpha$  sampled from a Beta distribution with parameters  $a, b$ , we have  $\mathbb{E}_\alpha [(1 - 2\alpha)^2] = 1 - \frac{4ab(a+b)}{(a+b)^2(a+b+1)} = \frac{(a-b)^2(a+b)}{(a+b)^2(a+b+1)} + \frac{1}{a+b+1}$  which is a constant when  $a, b$  are constant values. Denote this value by  $c$ . By plugging in the above equation back to equation (15), we have that in expectation over the randomness of  $\alpha, x_i, x_j$ , the change of bias is at least

$$\begin{aligned} \text{bias}(\hat{\beta}) - \text{bias}(\hat{\beta}^{\text{mixup}}) &\geq \frac{2\lambda^2 c}{n(n+1)} \beta^\top U \frac{D}{\left(\frac{D}{n+1} + \lambda\right)^2 \left(\frac{D}{n} + \lambda\right)} U^\top \beta - \frac{3\|\beta\|^2 \gamma^4}{\lambda^2(n+1)^2} \\ &\geq \frac{2\lambda^2 c}{(\gamma + \lambda)^3(n+1)} \frac{\|X\beta\|^2}{n} - \frac{3\|\beta\|^2 \gamma^4}{\lambda^2(n+1)^2} \end{aligned}$$

Similarly, using equation (16), we obtain that in expectation over the randomness of  $z$ , the change of variance is at least

$$\begin{aligned} \text{var}(\hat{\beta}) - \text{var}(\hat{\beta}^{\text{mixup}}) &\geq -\frac{3\sigma^2 c}{n} \cdot \text{Tr} \left[ (X^\top X + n\lambda \text{Id})^{-1} X^\top X (X^\top X + n\lambda \text{Id})^{-1} \right] \\ &= -\frac{3\sigma^2 c}{n} \cdot \text{Tr} \left[ \frac{D}{(D + n\lambda)^2} \right] \\ &= -\frac{3\sigma^2 c}{\lambda^2 n^3} \|X\|^2 \\ &\geq -\frac{3\sigma^2 c \gamma^2}{\lambda^2 n^2} \end{aligned}$$

In summary, when  $n$  is large enough, e.g. for

$$n \geq \frac{6(\gamma + \lambda)^3}{\lambda^4} \max\left(\frac{\|\beta\|^2 \gamma^4}{c}, \sigma^2 \gamma^2\right),$$

we conclude that

$$e(\hat{\beta}) - e(\hat{\beta}^{\text{mixup}}) \geq \frac{\lambda^2 c \|X\beta\|^2}{(\gamma + \lambda)^3 n^2} \gtrsim \frac{\lambda^2 \|X\beta\|^2}{\gamma^3 n^2}.$$

□

**Discussions.** Our result on the mixup augmentations opens up several immediate open questions. First, our setting assumes that there is a single linear model  $\beta$  for all data. An interesting direction is to extend our results to a setting with multiple linear models. This extension can better capture the multi-class prediction problems that arise in image classification tasks.

Second, our result also deals with adding a simple mixup sample. The limitation is that adding the mixup sample violates the assumption that  $\sum_{i=1}^n x_i = 0$  we make for the proof to go through. However, one would expect that the assumption is still true in expectation after adding the mixup sample. Formally establishing the result is an interesting question.

### A.3. Proof of Corollary 3.4

*Proof of Corollary 3.4.* The proof is by applying equation (3) over  $z_1 + z_2$  and  $z_1$  twice. The result follows by adding up results from the two. The details are omitted. □

### A.4. The Minimum Norm Estimator

In addition to the ridge estimator, the minimum norm estimator has also been studied in the over-parametrized linear regression setting (Bartlett et al., 2019; Xie et al., 2020). The minimum norm estimator is given by solving the following.

$$\begin{aligned} \min_{w \in \mathbb{R}^p} \|w\|^2 \\ \text{s.t. } Xw = Y. \end{aligned}$$

It is not hard to show that the solution to the above is given as  $\hat{\beta} = (X^\top X)^\dagger X^\top Y$ . We have focused on the ridge estimator since it is typically the case in practice that an  $l_2$ -regularization is added to the objective. In this part, we describe the implications of our results for the minimum norm estimator.



For this case, we observe that

$$\mathbb{E}_\varepsilon \left[ \hat{\beta} \right] - \beta = (X^\top X)^\dagger X^\top X \beta - X \beta = -P_X^\perp \beta,$$

where  $P_X^\perp$  is the projection orthogonal to  $X^\top X$ . Hence we can simplify the bias term to be  $\beta^\top P_X^\perp \beta$ . The variance term is  $\text{Tr}((X^\top X)^\dagger)$ .

**Dealing with psuedoinverse.** We need the following tools for rank-one updates on matrix psuedoinverse. Suppose  $X \in \mathbb{R}^{d \times d}$  is not invertible and  $u \in \mathbb{R}^d$  are column vectors. Let  $v = P_X u$  denote the projection of  $u$  onto the subspace of  $X$ . Let  $s = P_X^\perp u$  denote the projection of  $u$  orthogonal to  $X$ . Suppose that  $1 + u^\top X^\dagger u \neq 0$  and  $s, v$  are nonzero. In this case,

$$\begin{aligned} (X + uu^\top)^\dagger &= X^\dagger - X^\dagger v s^\top - s^\top v^\top X^\dagger + (1 + u^\top X^\dagger u) s^\dagger s^\top \\ &= X^\dagger - X^\dagger u s^\top - s^\dagger u^\top X^\dagger + (1 + u^\top X^\dagger u) s^\dagger s^\top. \end{aligned} \quad (20)$$

If  $s$  is zero, in other words  $u$  is in the row span of  $X$ , then

$$(X + uu^\top)^\dagger = X^\dagger - \frac{X^\dagger u u^\top X^\dagger}{1 + u^\top X^\dagger u}.$$

The proof of these identities can be found in (Riedel, 1992).

**Label-invariant transformations.** When  $\sigma = 0$ , i.e. there is no label in the labels, we can get a result qualitatively similar to Theorem 3.1 for the minimum norm estimator. This is easy to see because the bias term is simply  $\beta^\top P_X^\perp \beta$ . When we add  $z$  which is orthogonal to the span of  $X$ , then we get that

$$P_{X^{\text{aug}}}^\perp = P_X^\perp - \frac{z z^\top}{\|z\|^2}.$$

Hence the bias of  $\hat{\beta}$  reduces by  $\frac{\langle z, \beta \rangle^2}{\|z\|^2}$ .

When  $\sigma \neq 0$ , the variance of  $\hat{\beta}$  can increase at a rate which is proportional to  $1/\|z\|^2$ . This arises when we apply (20) on  $(X^\top X + z z^\top)^\dagger$ . In Theorem 3.1, we can circumvent the issue because the ridge estimator ensures the minimum singular value for  $P_X^\perp$  is at least  $\lambda$ .

**The mixup transformations.** For the minimum norm estimator, we show that adding a mixup sample always increases the estimation error for  $\beta$ . This is described as follows.

**Proposition A.4.** *Let  $X_{\text{aug}}$  and  $Y_{\text{aug}}$  denote the augmented data after adding  $(x^{\text{aug}}, y^{\text{aug}})$  to  $(X, Y)$ . Then for the minimum norm estimator, we have that: a) the bias is unchanged; b) the variance increases, for any constant  $\alpha \in (0, 1)$  which does not depend on  $n$ .*

*Proof.* For Claim a), we note that since  $P_X^\perp x_{\text{aug}} = 0$ , we have  $P_{X_{\text{aug}}}^\perp = P_X^\perp$ . Hence, the bias term stays unchanged after adding  $(x_{\text{aug}}, y_{\text{aug}})$ .

For Claim b), we note that the variance of  $\hat{\beta}$  for the test error is given by

$$\text{Tr} \left[ (X_{\text{aug}}^\top X_{\text{aug}})^\dagger (X_{\text{aug}}^\top X_{\text{aug}})^\dagger X_{\text{aug}}^\top \varepsilon^{\text{aug}} \varepsilon^{\text{aug}^\top} X_{\text{aug}} \right]$$

Taking the expectation over  $\varepsilon$ , we have (c.f. equation (19))

$$\mathbb{E}_\varepsilon \left[ X_{\text{aug}}^{\text{aug}^\top} \varepsilon^{\text{aug}} \varepsilon^{\text{aug}^\top} X_{\text{aug}} \right] = \sigma^2 \cdot \left( (\alpha^2 + (1 - \alpha)^2 + 2) x_{\text{aug}}^{\text{aug}} x_{\text{aug}}^{\text{aug}^\top} + X^\top X \right).$$

Meanwhile, for  $(X_{\text{aug}}^\top X_{\text{aug}})^\dagger$ , we use the Sherman-Morrison formula to decompose the change of the rank-1 update  $x_{\text{aug}} x_{\text{aug}}^\top$ . Since  $P_X^\perp x_{\text{aug}} = 0$  and  $x_{\text{aug}}^\top K x_{\text{aug}} \lesssim \|x_{\text{aug}}\|^2/n < 1$ . Denote by  $K = (X^\top X)^\dagger$ , we have

$$(X_{\text{aug}}^\top X_{\text{aug}})^\dagger = K - \frac{K x_{\text{aug}} x_{\text{aug}}^\top K}{1 + x_{\text{aug}}^\top K x_{\text{aug}}}$$

To sum up, we have that

$$\begin{aligned} & \mathbb{E} \left[ \text{Tr} \left[ (X_{\text{aug}}^\top X_{\text{aug}})^\dagger (X_{\text{aug}}^\top X_{\text{aug}})^\dagger X_{\text{aug}}^\top \varepsilon^{\text{aug}} \varepsilon^{\text{aug}\top} X_{\text{aug}} \right] \right] \\ &= \sigma^2 \text{Tr} [K] + (\alpha^2 + (1 - \alpha)^2) \text{Tr} [K K x_{\text{aug}} x_{\text{aug}}^\top] \cdot (1 + O(\frac{1}{n})) \end{aligned}$$

Hence the variance of the minimum norm estimator increases after adding the mixup sample. □

## B. Supplementary Materials for Section 4

We present a full list of results using our proposed metrics to study label-invariant transformations on MNIST. Then, we provide an in-depth study of the data points corrected by applying the rotation transformation. Finally, we show that we can get a good approximation of our proposed metrics by using 3 random seeds. The training details are deferred to Appendix C.

**Further results on label-invariant transformations.** For completeness, we also present a full list of results for label-invariant transformations on MNIST in Table 4. The descriptions of these transformations can be found in the survey of (Shorten & Khoshgoftaar, 2019).

	Avg. Acc.	Error score	Instability
Baseline	98.08% ( $\pm 0.08\%$ )	1.52%	0.95%
Invert	97.29% ( $\pm 0.18\%$ )	1.85%	1.82%
VerticalFlip	97.30% ( $\pm 0.09\%$ )	1.84%	1.82%
HorizontalFlip	97.37% ( $\pm 0.09\%$ )	1.83%	1.72%
Blur	97.90% ( $\pm 0.12\%$ )	1.62%	1.24%
Solarize	97.96% ( $\pm 0.07\%$ )	1.51%	1.27%
Brightness	98.00% ( $\pm 0.10\%$ )	1.69%	1.05%
Posterize	98.04% ( $\pm 0.08\%$ )	1.59%	1.12%
Contrast	98.08% ( $\pm 0.10\%$ )	1.61%	0.93%
Color	98.10% ( $\pm 0.06\%$ )	1.59%	0.88%
Equalize	98.12% ( $\pm 0.09\%$ )	1.53%	0.93%
AutoContrast	98.17% ( $\pm 0.07\%$ )	1.50%	0.87%
Sharpness	98.17% ( $\pm 0.08\%$ )	1.45%	0.92%
Smooth	98.18% ( $\pm 0.02\%$ )	1.47%	0.99%
TranslateY	98.19% ( $\pm 0.11\%$ )	1.27%	1.13%
Cutout	98.31% ( $\pm 0.09\%$ )	1.43%	0.86%
TranslateX	98.37% ( $\pm 0.07\%$ )	1.21%	1.00%
ShearY	98.54% ( $\pm 0.07\%$ )	1.17%	0.85%
ShearX	98.59% ( $\pm 0.09\%$ )	<b>1.00%</b>	0.84%
RandomCropping	98.61% ( $\pm 0.05\%$ )	1.01%	0.88%
Rotation	<b>98.65%</b> ( $\pm 0.06\%$ )	1.08%	<b>0.77%</b>

Table 4. Measuring the intrinsic error and instability scores of label-invariant transformations on MNIST. The results are obtained using 9 random seeds.

Next we describe the details regarding the experiment on rotations. Recall that our goal is to show that adding rotated data points provides new information. We show the result by looking at the data points that are corrected after adding the rotated data points. Our hypothesis is that the data points that are corrected should be wrong “with high confidence” in the baseline model, as opposed to being wrong “with marginal confidence”.

Here is our result. Among the 10000 test data points, we observe that there are 57 data points which the baseline model predicts wrong but adding rotated data corrected the prediction. And there are 18 data points which the baseline model predicts correctly but adding rotated data. Among the 57 data points, the average accuracy of the baseline model over the 9 random seeds is 19.50%. Among the 18 data points, the average accuracy of the augmented model is 29.63%. The result confirms that adding rotated data corrects data points whose accuracies are low (at 19.50%).

*Efficient estimation.* We further observe that on MNIST, three random seeds are enough to provide an accurate estimation of

## On the Generalization Effects of Linear Transformations in Data Augmentation

	Baseline	HorizontalFlip	Contrast	Cutout	TranslateY	ShearX	RandomCropping	Rotation
9 seeds	1.75%	1.83%	1.61%	1.43%	1.27%	1.00%	1.01%	1.08%
3 seeds	1.72%	2.13%	1.78%	1.54%	1.49%	1.10%	1.11%	1.16%

Table 5. The result of estimating the intrinsic error score using 3 seeds is within 16% of the estimation result using 9 seeds.

the intrinsic error score. Furthermore, we can tell which augmentations are effective versus those which are not effective from the three random seeds.

### C. Supplementary Materials for Section 5

We describe details on our experiments to complement Section 5. In Appendix C.1, we review the datasets used in our experiments. In Appendix C.2, we describe the models we use on each dataset. In Appendix C.3, we summarize the training procedures for all experiments.

#### C.1. Datasets

We describe the image and text datasets used in the experiments in more detail.

**Image datasets.** For image classification tasks, the goal is to determine the object class shown in the image. We select four popular image classification benchmarks in our experiments.

**MNIST.** MNIST (LeCun et al., 1998) is a handwritten digit dataset with 60,000 training images and 10,000 testing images. Each image has size  $28 \times 28$  and belongs to 1 of 10 digit classes.

**CIFAR-10.** CIFAR-10 (Krizhevsky et al., 2009) has 60,000  $32 \times 32$  color images in 10 classes, with 6,000 images per class where the training and test sets have 50,000 and 10,000 images, respectively.

**CIFAR-100.** CIFAR-100 is similar to CIFAR-10 while CIFAR-100 has 100 more fine-grained classes containing 600 images each.

**Street view house numbers (SVHN).** This dataset contains color house-number images with 73,257 core images for training and 26,032 digits for testing which is similar in flavor to MNIST.

**ImageNet Large-Scale Visual Recognition Challenge (ImageNet).** This dataset is a subset of original ImageNet dataset which has over 15M labeled high-resolution images belonging to roughly 22,000 categories. In this dataset, it includes images of 1000 classes, and has roughly 1.3M training images and 50,000 validation images.

**Text dataset.** For text classification task, the goal is to understand the sentiment opinions expressed in the text based on the context provided. We choose *Internet movie database (IMDb)*.

**Internet movie database (IMDb).** IMDb (Maas et al., 2011) consists of a collection of 50,000 reviews, with no more than 30 reviews per movie. The dataset contains 25,000 labeled reviews for training and 25,000 for test.

#### C.2. Models

We describe the models we use in the our experiments.

**Image datasets.** For the experiments on image classification tasks, we consider five different models including multi-layer perceptron (MLP), Wide-ResNet, Shake-Shake (26 2x96d), PyramidNet+ShakeDrop, and ResNet.

For the MLP model, we reshape the image and feed it into a two layer perceptron, followed by a classification layer.

For the Wide-ResNet model, we use the standard Wide-ResNet model proposed by (Zagoruyko & Komodakis, 2016). We select two Wide-ResNet models in our experiment: Wide-ResNet-28-2 with depth 28 and width 2 as well as Wide-ResNet-28-10 with depth 28 and width 10.

For Shake-Shake, we use the model proposed by (Gastaldi, 2017) with depth 26, 2 residual branches. The first residual block has a width of 96, namely Shake-Shake (26 2x96d).

On the Generalization Effects of Linear Transformations in Data Augmentation

Dataset	Model	Batch Size	LR	WD	LR Schedule	Epoch
MNIST	MLP	500	0.1	0.0001	cosine	100
CIFAR-10	Wide-ResNet-28-2	128	0.1	0.0005	multi-step	200
CIFAR-10	Wide-ResNet-28-10	128	0.1	0.0005	cosine	200
CIFAR-10	Shake-Shake (26 2x96d)	128	0.01	0.001	cosine	1800
CIFAR-10	PyramidNet+ShakeDrop	64	0.05	0.00005	cosine	1800
CIFAR-100	Wide-ResNet-28-10	128	0.1	0.0005	cosine	200
SVHN	Wide-ResNet-28-10	128	0.005	0.005	cosine	200
ImageNet	ResNet-50	160	0.1	0.0001	multi-step	270

Table 6. Model hyperparameters on MNIST and CIFAR-10/CIFAR-100. LR means learning rate, WD means weight decay and LR Schedule means learning rate scheduling. The parameters we choose are based on the parameters used for the model previously.

For PyramidNet, we choose the model proposed by (Han et al., 2017) with ShakeDrop (Yamada et al., 2018). We set the depth as 272 and alpha as 200 in our experiments. We refer to the model as PyramidNet+ShakeDrop.

For ResNet, we select the model proposed by (He et al., 2016) with 50 layers. We refer to the model as ResNet-50.

For CIFAR-10 and CIFAR-100, after applying the transformations selected by Algorithm 1, we apply randomly cropping, horizontal flipping, cutout and mixup.

For ImageNet, we apply random resized cropping, horizontal flipping, color jitter and mixup after performing Algorithm 1.

Similarly, for SVHN we apply randomly cropping and cutout after using Algorithm 1.

**Text dataset.** For the experiments on text classification tasks, we use a state-of-the-art language model call BERT (Devlin et al., 2018). We choose the BERT<sub>LARGE</sub> uncased model, which contains 24 layers of transformer.

### C.3. Training procedures.

We describe the training procedures for our experiments.

**Implementation details.** During the training process, for each image in a mini-batch during the optimization, we apply Algorithm 1 to find the most uncertain data points and use them to training the model. We only use the augmented data points to train the model and do not use the original data points. The specific transformations we use and the default transformations are already mentioned in Section 5.1.

**Image classification tasks.** We summarize the model hyperparameters for all the models in Table 6.

**Text classification tasks.** For text classification experiments, we follow the BERT<sub>LARGE</sub> fine-tune procedure proposed by (Devlin et al., 2018). We set the dropout rate of 0.1 and apply grid search to tune the learning rate from  $\{1e^{-5}, 2e^{-5}, 3e^{-5}\}$ , batch size of 32 and 64, and the number of epochs from 2 to 10.

## D. Additional Experiments on Mixup

We present additional experiments to provide micro insights on the effects of applying the mixup augmentation. First, we consider the performance of mixing same class images versus mixing different class images. We show that on CIFAR-10, mixup can reduce the intrinsic error score but mixing same class images does not reduce the intrinsic error score. Second, we consider the effect of mixup on the size of the margin of the classification model. We show that mixup corrects data points which have large margins from the baseline model.

**Training procedures.** We apply the MLP model on MNIST and the Wide-ResNet-28-10 model on CIFAR-10. For the training images, there is a default transformation of horizontal flipping with probability 0.5 and random cropping. Then we apply the mixup transformation.

**Comparing mixup with  $\ell_2$ -regularization.** We tune the optimal  $\ell_2$  regularization parameter on the baseline model. Then, we run mixup with the optimal  $\ell_2$  parameter. We also consider what happens if we only mix data points with the same class labels. Since mixup randomly mixes two images, the mixing ratio for same class images is 10%. Table 7 shows the results.

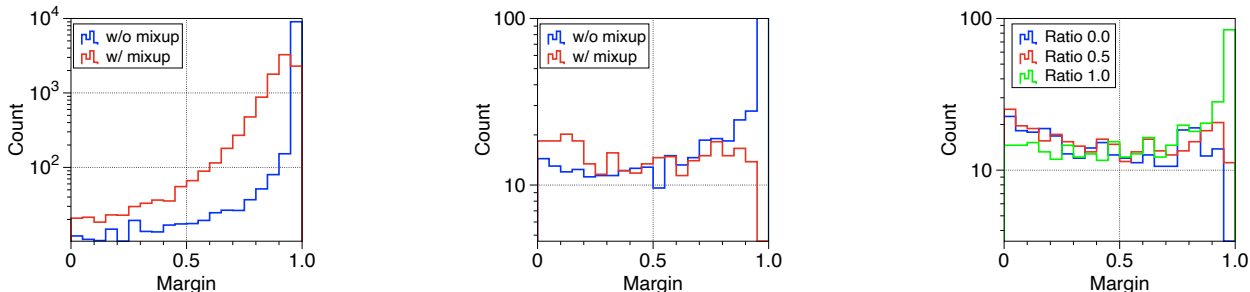
We make several observations. (i) Adding  $\ell_2$  regularization reduces the instability score. This is expected since a higher value of  $\lambda$  reduces the variance part in equation (1). (ii) On MNIST, mixing same class images improves stability compared to mixup. This is consistent with our results in Section 4. Furthermore, optimally tuning the regularization parameter outperforms mixup. (iii) on CIFAR-10, mixing same class images as well as mixup reduces the instability score by more than 10%. Furthermore, we observe that mixing different class images also reduces the intrinsic error score.

Dataset	MNIST			CIFAR-10		
	Avg. Acc.	Error score	Instability	Avg. Acc.	Error score	Instability
Baseline	98.07%±0.08%	1.52%	0.95%	87.33%±0.31%	9.61%	7.97%
Tuning $\ell_2$ reg.	<b>98.31%</b> ±0.05%	<b>1.50%</b>	0.67%	91.23%±0.15%	6.55%	5.50%
Mixup-same-class w/ $\ell_2$	98.15%±0.06%	1.78%	<b>0.61%</b>	91.59%±0.21%	6.55%	<b>4.96%</b>
Mixup w/ $\ell_2$ reg.	98.21%±0.05%	<b>1.58%</b>	0.70%	<b>92.44%</b> ±0.32%	<b>5.71%</b>	<b>4.75%</b>

Table 7. Measuring the intrinsic error and instability scores of mixing same class images versus mixup. The optimal  $\ell_2$  regularization parameter is  $10^{-3}$  for MNIST and  $5 \times 10^{-4}$  for CIFAR-10.

**Effects over margin sizes on the cross-entropy loss.** We further explore why mixup can improve the average prediction accuracy on CIFAR-10. Intuitively, when mixing two images with different losses, the resulting image will have a loss which is lower than the image with the higher loss. For the classification setting, instead of measuring the losses, we look at the margin, which is the difference between the largest predicted probability and the second largest predicted probability over the 10 classes. It is well-known that a larger margin implies better robustness and generalization properties (Friedman et al., 2001; Hastie et al., 2009).

By comparing the margin distribution between the baseline model versus the one after applying the mixup augmentations, we provide several interesting observations which shed more light on why mixup works. Figure 6 shows the results. First, we observe that among the images for which the baseline model predicts incorrectly, mixup is able to correct those images whose margin is very large! Second, we observe that among the images for which the baseline model predicts correctly, applying mixup increased the size of the margin for the images whose margin is small on the baseline model.



(a) Correctly predicted images on the baseline model.

(b) Incorrectly predicted images on the baseline model.

(c) Corrected images after applying mixup with a certain mixing ratio.

Figure 6. On CIFAR-10, the images corrected by applying the mixup augmentation have large margins on the baseline model. Here the margin is defined as the largest probability minus the second largest probability predicted by the cross entropy loss. By varying the mixing ratio, we observe that mixing same class images only does not correct the the data points which are predicted incorrectly with large margins on the baseline model.

We further observe that the above effect arises from mixing different class images. We compare the effects of three settings, including i) Mix images from different classes only (Ratio 0.0). ii) For half of the time, mix images from different classes; Otherwise, mix images from the same classes (Ratio 0.5). iii) Mix images from the same classes only (Ratio 1.0). In Figure 6c, we observe the effect of correcting large margin data points does not occur when we only mix same class images.