LABEL AUGMENTATION FOR NEURAL NETWORKS ROBUSTNESS

Fatemeh Amerehi
University of Limerick, Ireland
fatemeh.amerehi@ul.ie

Patrick Healy
University of Limerick, Ireland
patrick.healy@ul.ie

ABSTRACT

Out-of-distribution generalization can be categorized into two types: common perturbations arising from natural variations in the real world and adversarial perturbations that are intentionally crafted to deceive neural networks. While deep neural networks excel in accuracy under the assumption of identical distributions between training and test data, they often encounter out-of-distribution scenarios resulting in a significant decline in accuracy. Data augmentation methods can effectively enhance robustness against common corruptions, but they typically fall short in improving robustness against adversarial perturbations. In this study, we develop Label Augmentation (LA), which enhances robustness against both common and intentional perturbations and improves uncertainty estimation. Our findings indicate a Clean error rate improvement of up to 23.29% when employing LA in comparisons to the baseline. Additionally, it enhances robustness under common corruptions benchmark by up to 24.23%. When tested against FGSM and PGD attacks, improvements in adversarial robustness are noticeable, with enhancements of up to 53.18% for FGSM and 24.46% for PGD attacks.

1 INTRODUCTION

Real-world objects exhibit a diverse array of intertwined attributes. While certain characteristics, such as the class identity of the object, are permanent, others, like the lighting conditions or pose of the object, are transient (Gabbay & Hoshen, 2019). In fact, how would you interpret the images in Fig. 1? If you were to present them to someone and ask about their interpretation, they would likely identify ***blurry*** birds and cars in ***snow***. Despite the variations in the images, we are capable of effectively distinguishing between the class identity and the transient attributes of an object.

We already assign names and labels for objects around us, but we also have names for concepts like brightness, warmth, noisiness, and many more. The birds or cars themselves are unchanged, but the sharpness and colors are different. Essentially, the name/labels remain invariant to us, but we still recognize that there exist some other elements that differ from each other. The process of training a machine to make similar distinctions among various attributes in observed data is referred to as disentanglement. It aims to find latent representations that adeptly separate the explanatory factors contributing to variations in the input data (Bengio et al., 2013).

Disentangled representations have been shown to improve generalization to unseen scenarios in both generative and discriminative tasks (Gabbay & Hoshen, 2019; Eom & Ham, 2019; Träuble et al., 2021). Deep Neural Networks (DNNs) generalize well under the assumption of Independent and Identically Distributed (IID) data, where both training and test datasets come from the same distribution. Yet, high IID accuracy does not guarantee out-of-distribution (OOD) generalization where train and test distributions mismatch (Liu et al., 2021a).

Figure 1: What do you see when looking at the images?

Common corruptions (Hendrycks & Dietterich, 2019) and adversarial perturbations (Goodfellow et al., 2014) are two examples of OOD scenarios leading to performance deterioration. A widely used approach to mitigate performance drop is to incorporate data augmentation into the training pipeline (Shorten & Khoshgoftaar, 2019). While data augmentation enhances model robustness, current methods tend to improve either common corruption or adversarial perturbation individually, rather than concurrently enhancing both.

Beyond vulnerability to distributional shifts, another common issue is miscalibration—the tendency of models to generate overconfident predictions when the training examples are IID. This overconfidence is further intensified under OOD scenarios (Ovadia et al., 2019). In this study, for enhanced robustness, we present a simple yet effective method using Label Augmentation (LA) for disentangling the class of an object from irrelevant noise. The LA proves effective in enhancing calibration and robustness against both common and intentional perturbations of input data.

2 Related Works

Focusing on vision models, we review relevant literature on augmentation methods for robustness against distributional shifts, including adversarial attacks, and common corruptions, alongside calibration.

Augmentation methods for robustness under distribution shift. Vision models often experience a drop in performance under common or intentional perturbations of images (Hendrycks & Dietterich, 2019; Szegedy et al., 2013). For instance, they show vulnerability to blur and Gaussian noise (Vasiljevic et al., 2016; Dodge & Karam, 2016), as well as factors such as brightness and contrast (Hendrycks & Dietterich, 2019), occlusion (Zhong et al., 2020), and small translations or rescalings of the input data (Azulay & Weiss, 2018). Additionally, when the model encounters adversarial perturbations, its performance tends to suffer even more (Goodfellow et al., 2014; Papernot et al., 2016a; Tramèr et al., 2017b; Athalye et al., 2018).

To mitigate performance degradation caused by common corruptions, a commonly employed strategy is the incorporation of label-preserving image augmentation into the training pipeline (Shorten & Khoshgoftaar, 2019). In the simplest form, data augmentations translate to simple transformations such as horizontal flipping, color shift, and random cropping (Krizhevsky et al., 2012; He et al., 2016). A more complex array of augmentations includes techniques such as random erasing (DeVries & Taylor, 2017; Zhong et al., 2020), neural style transfer (Jackson et al., 2019; Geirhos et al., 2018), image mixing (Zhang et al., 2017; Inoue, 2018; Summers & Dinneen, 2019; Hong et al., 2021; Yao et al., 2022), training with noise (Lopes et al., 2019; Rusak et al., 2020), randomized manipulations of images (Xu et al., 2023), combination and mixing of augmentation chains (Hendrycks et al., 2019; Modas et al., 2022), or search for an optimal augmentation policy (Cubuk et al., 2019).

Defense mechanisms to tackle adversarial examples—a carefully crafted perturbations to mislead a classifier— include defensive distillation (Papernot et al., 2016b), feature squeezing (Xu et al., 2017), adversarial detection (Metzen et al., 2017; Pang et al., 2018; Deng et al., 2021a), gradient regularization (Tramèr et al., 2017a; Wu et al., 2020), and adversarial training (Goodfellow et al., 2014; Madry et al., 2017). Among these, the most effective strategy is adversarial training, which involves augmenting training data with adversarial examples to enhance its robustness against attacks or to reduce its test error on clean inputs (Goodfellow et al., 2014; Kurakin et al., 2016; Moosavi-Dezfooli et al., 2016; Ford et al., 2019; Bai et al., 2021).

Adversarial examples could be augmented in various ways, including incorporating synthetic data (Gowal et al., 2021; Wang et al., 2023b), unlabeled data (Carmon et al., 2019; Deng et al., 2021b), injecting noise to the hidden layers (Qin et al., 2022), adversarial mixture of transformations (Wang et al., 2021), or reconfiguration of the low and high-frequency components of intermediate feature representations (Bu et al., 2023). Other methods introduce weight perturbation to enhance model robustness (Wu et al., 2020), regulating gradient growth to prevent robust overfitting during multi-step adversarial training (Li et al., 2022), or ensemble training to mitigate vulnerabilities across submodels while preserving comparable accuracy on clean data (Cai et al., 2023).

The effectiveness of adversarial training depends on the choice of adversarial examples. For instance, training exclusively with Fast Gradient Sign Method (FGSM) (Goodfellow et al., 2014) enhances robustness against non-iterative attacks but lacks robustness against iterative attacks such as Projected Gradient Descent (PGD) attack (Kurakin et al., 2016; Madry et al., 2017). Whether adversarial training enhances robustness against common corruptions has conflicting views in the literature. While some studies suggest a positive correlation (Ford et al., 2019; Kireev et al., 2022), others argue that adversarial robustness and robustness to common perturbations are independent (Laugros et al., 2019).

Calibration. Despite performing well in generalization and prediction under the IID setting, DNNs often produce overconfident results, which worsen even more in OOD settings (Guo et al., 2017; Ovadia et al., 2019; Gawlikowski

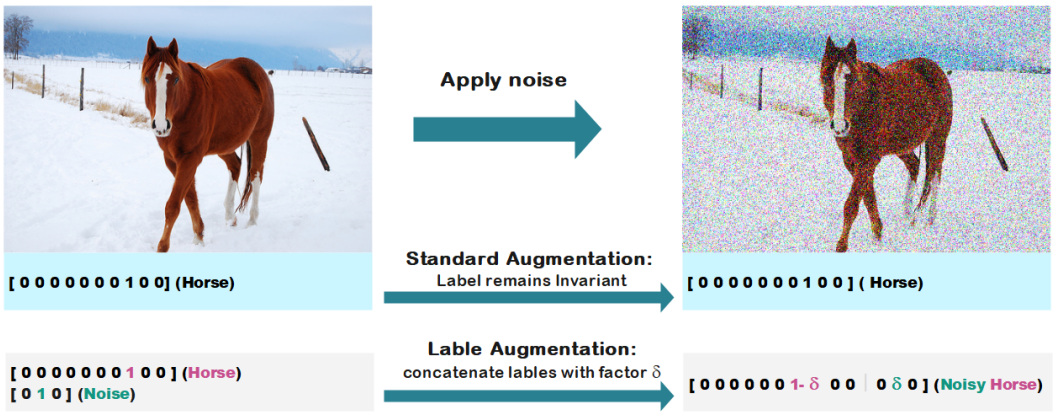

Figure 2: The Cifar10 dataset includes 10 classes representing airplanes, cars, birds, cats, deer, dogs, frogs, horses, ships, and trucks. The one-hot label for horses is [0 0 0 0 0 0 0 1 0 0]. Considering three distinct augmentation operation classes like contrast, noise, and blur; the one-hot label for noise is [0 1 0]. In standard augmentation, labels remain invariant. When applying Label Augmentation with a smoothing factor δ, the resulting label for noisy image of a horse is [0 0 0 0 0 0 0 $1-\delta$ 0 0 0 δ 0]. This maintains invariance with original categories while distinguishing between more abstract concepts, such as noisy and noise-free inputs.

et al., 2021). Well-calibrated uncertainty estimates indicate when the output of models is reliable and when it is questionable. Temperature scaling with a validation set (Guo et al., 2017), or ensembling predictions from independently trained classifiers on the entire dataset with random initialization (Lakshminarayanan et al., 2017), improves calibration. Using soft labels (Szegedy et al., 2016)—a weighted average of one-hot training labels and a uniform distribution over targets—often prevents the network from becoming overly confident in specific labels, thus reducing calibration errors (Müller et al., 2019; Lukasik et al., 2020).

3 Label Augmentation

We now introduce our central idea.

Let $\mathcal{O} = \{o_i\}_{i=1}^{M}$ represent a set of m label-preserving augmentation operations, each of which, when applied to an input data, introduces certain effects to it. Let $\mathcal{Z} = \{z_i\}_{i=1}^{M}$ be the one-hot encoded name for each of the operations. Given a collection of objects $\mathcal{X} = \{x_i\}_{i=1}^{N}$ and a set of labels $\mathcal{Y} = \{y_i\}_{i=1}^{K}$, we humans assign a label y_i to each object x_i based on the attributes we observe in them. Further, let $\mathcal{O}(\mathcal{X})$ represent operations within $\mathcal{O}$ that are applied to objects in $\mathcal{X}$.

If we select an operation $o_j \in \mathcal{O}$ and apply it to $x_i \in \mathcal{X}$ as the certain attributes of the x_i are affected—despite the identity of the class object remains the same for each elements in $\mathcal{O}(\mathcal{X})$—we no longer assign the same label y_i to the transformed objects/images (revisit Fig. 1). Instead, we assign a richer name that incorporates both the class identity and the effect. In essence, we disentangle class identity from transformations/operations.

In a task of K-class classification, the goal is to model the mapping from the input data x_i to its corresponding class label y_i through a DNN $f : \mathcal{X} \to \mathcal{Y}$. Typically, this involves using a softmax output layer and cross-entropy loss to quantify the dissimilarity between y_i, the true class distribution (one-hot encoded), and p_i, the softmax of predictions. The cross-entropy loss, is defined as $\mathcal{L}_{CE}(y_i, p_i) = -\sum_{k=1}^{K} y_{ik} \log p_{ik}$, where y_{ik} represents the k-th element of the true class distribution y_i, and p_{ik} denotes the k-th element of the predicted class distribution p_i.

To extend the model generalization capability to OOD data, existing augmentation methods train with augmented input $\mathcal{O}(\mathcal{X})$ while assigning the same label to transformed and untransformed input data to help the model learn representations that remain invariant to a set of data augmentations. Essentially, existing techniques aim to find a mapping $f : \mathcal{X} \cup \mathcal{O}(\mathcal{X}) \to \mathcal{Y}$. Given the distinction we make in our naming between $\mathcal{X}$ and $\mathcal{O}(\mathcal{X})$, and considering the use of such augmented inputs in training DNNs, would it not be advantageous to explicitly communicate to the model that labels differ in additional factors beyond class identity? To enable this, we employ Label Augmentation (LA).

In LA, the objective is to maintain invariance to the input class category y_i, while simultaneously enabling the distinctions between x_i and its various transformed versions. To achieve this, after any transformation o_j on input data x_i, we simply concatenate the two one-hot labels y_i and z_j to each other with a factor of δ. In other words, whenever we augment the input data, we augment labels as well. Specifically, the labels assigned to $\tilde{x_i} = o_j(x_i)$ are defined as in Eq. 1, which represent a vector of length $K + M$. The label $\tilde{y_i}$ has the value of $1 - \delta$ at position i and the value of δ at position $K + j$. In other words, LA aims to find a more comprehensive mapping $f : \mathcal{X} \cup \mathcal{O}(\mathcal{X}) \rightarrow \text{Concat}_\delta[\mathcal{Y}, \mathcal{Z}]$ that maps augmented collections of input to the augmented labels. This is shown in Fig. 2.

$$\tilde{y_i} = \text{Concat}[(1 - \delta)y_i, \delta z_j] \tag{1}$$

We consider identity transformation with $\delta = 0$ as a specific case that represents untransformed input data. In case of transformation, we select δ to be a small value in order to prevent excessive deviation of the model towards the augmented label. The value of δ is drawn from a uniform distribution: $\delta \sim \text{U}(0.05, 0.1)$.

To ensure the same dimensionality and maintain the class identity y_i for any untransformed input data x_i, we simply expand the one-hot labels y_i from K dimensions to $K+M$. At position i, we assign 1, as before, to represent the class identity. In LA training, the loss is computed as $\mathcal{L}_{LA}(\tilde{y_i}, \tilde{p_i}) = -\sum_{k=1}^{K+M} \tilde{y}_{ik} \log \tilde{p}_{ik}$, where $\tilde{p_i}$ denotes the softmax of predictions for $\tilde{x_i}$.

In the following section, we show that the act of assigning names to operations and augmenting labels leads to better generalization compared to traditional augmentation. Furthermore, as we will demonstrate, this helps achieve better robustness to both common and intentional perturbations, as well as improved calibration.

Figure 3: Examples of augmentation operations applied in Label Augmentation.

4 Experimental Setup

In the following, we elaborate on the dataset, training configuration, the networks employed, and the evaluation metrics for assessing both robustness and calibration. Afterwards, we present the results and analysis.

4.1 Configurations and Metrics

Datasets. We utilize the CIFAR-10 and CIFAR-100 (Krizhevsky et al., 2009) datasets. Both datasets contain 50,000 training images and 10,000 testing images of size $32 \times 32 \times 3$. To assess the robustness of models against common data shift, we evaluate on CIFAR-10-C and CIFAR-100-C benchmark (Hendrycks & Dietterich, 2019). These datasets are created by introducing various distortions to the original CIFAR-10 and CIFAR-100 datasets, and contains a total of 15 corruptions of types such as noise, blur, weather, and digital distortions. Each distortion is incorporated at severity levels $1 \leq s \leq 5$. In the following, we refer to these datasets as CIFAR and CIFAR-C, respectively.

Robustness metrics. In Tables 1, 2, and 3, the Clean Error represents the standard classification error on uncorrupted test data. For a given corruption c, the error rate at corruption severity s is denoted as $\text{E}_{c,s}$. Taking the average error across these severities s; the corruption error CE_c, is computed as $\text{CE}_c = \frac{1}{5}\sum_{s=1}^{5} \text{E}_{c,s}$. Finally, the mean Corruption Error $\text{mCE} = \frac{1}{15}\sum_{c=1}^{15} \text{CE}_c$ is the average of all 15 corruption errors. This gives one value for robustness comparisons against common corruptions (Hendrycks & Dietterich, 2019; Hendrycks et al., 2019).

To measure robustness against adversarial perturbations, we employ FGSM (Goodfellow et al., 2014) and 40-step iterative PGD (Madry et al., 2017) attacks, both with L_∞ constraints using two budgets $\varepsilon = 0.03$ and $\varepsilon = 0.3$. We utilize the implementations provided by the *cleverhans* 4.0 library (Papernot et al., 2018).

Calibration metric. A classifier is considered calibrated when it can consistently predict their accuracy (Guo et al., 2017). For instance, with 100 predictions, each at a confidence level of 0.7, we expect 70 correct classifications. We evaluate the calibration of the network using the Expected Calibration Error (ECE) (Guo et al., 2017) and Root Mean Square (RMS) Calibration Error. Given the finite size of the test sets, ECE and RMS Calibration Error are estimated by grouping all n test examples into M equal size bins, ordered by prediction confidence—the winning softmax score.

Let B_m represent the set of samples whose predictions fall into bin m. The accuracy and confidence of B_m is defined as $\text{acc}(B_m) = \frac{1}{|B_m|}\sum_{i\in B_m} 1(\hat{y}_i = y_i)$ and $\text{conf}(B_m) = \frac{1}{|B_m|}\sum_{i\in B_m} \hat{p}_i$, respectively. Here, $\hat{y}_i$ and y_i represent the predicted and ground-truth labels for input data x_i, and $\hat{p}_i$ is the confidence—winning score—of sample i. The ECE and RMS errors is then defined as in Eq. 2 and Eq. 3, respectively. We use the implementations provided by the *TorchMetric* 1.4.0 library (Lightning, 2024).

$$\text{ECE} = \sum_{m=1}^{M} \frac{|B_m|}{n} \Big|\text{acc}(B_m) - \text{conf}(B_m)\Big| \tag{2}$$

$$\text{RMS} = \sqrt{\sum_{m=1}^{M} \frac{|B_m|}{n} \Big(\text{acc}(B_m) - \text{conf}(B_m)\Big)^2} \tag{3}$$

Training Configuration and hyper-parameter setting. We run all experiments on a GeForce RTX-3080Ti GPU with CUDA Version 12.0 using the PyTorch version 2.0.1. To assess robustness across different architectures, we use a standard LeNet (LeCun et al., 1998), a ResNet-50 (He et al., 2016), a 40-2 Wide ResNet (Zagoruyko & Komodakis, 2016), a 32×4d ResNeXt-50 (Xie et al., 2017), and a Swin Transformer (Liu et al., 2021b). All networks start with a learning rate of 0.1, which decays by a factor of 0.0001 according to a cosine annealing learning rate (Loshchilov & Hutter, 2016). Before any augmentations, we preprocess input images through random horizontal flip and cropping. In all experiments, we train for 25 epochs with default weights and optimize with stochastic gradient descent with a momentum of 0.9. For both training and evaluation, we set the batch size to 1024. Each experiment in Tables 3, 4, and 5 are conducted three times, and the averages along with their corresponding standard deviations are reported.

Baseline comparisons. We begin by comparing LA with traditional augmentations, selecting a set of label-preserving augmentations such as Plasma noise (Nicolaou et al., 2022), Planckian Jitter (Zini et al., 2022), and Gamma adjustment, as illustrated in Fig. 3. First, we train the models with these augmentations using LA, and then repeat the training with normal augmentations, without LA. The results of these experiments are presented in Tables 1 and 2, and we will analyze them in the next section.

To measure the effectiveness of LA, we make a comparison between various augmentation techniques, including Mixup (Zhang et al., 2017), Augmix (Hendrycks et al., 2019), and AutoAugment (Cubuk et al., 2019). Additionally, we include FGSM (Goodfellow et al., 2014) and 10-step iterative PGD (Madry et al., 2017), both with L_∞ constraints and $\varepsilon = 0.3$, for comparisons against adversarial training. Except for LA, in all these experiments, we adjust the last classification layer of networks to output 10 and 100 class categories corresponding to CIFAR-10, and CIFAR-100 datasets, respectively. For LA, depending on the number of operations used for augmentation, we add additional units to accommodate the prediction of augmented classes. After training the models with LA, during the testing phase, we ignore the outputs for augmented labels—prior to the softmax operation—and only consider the class identity labels as the final output of the models. This is because the class categories in CIFAR and CIFAR-C datsets are not linked to augmentation operations like Plasma noise, etc. This could be thought of as asking a person to filter out what they see in a transformed picture and just identify the class without providing extra detail.

Augmentation operations. Corruptions employed in the CIFAR-C benchmark dataset (Hendrycks & Dietterich, 2019) include Gaussian Noise, Shot Noise, Impulse Noise, Defocus Blur, Glass Blur, Motion Blur, Zoom Blur, Snow, Frost, Fog, Brightness, Contrast, Elastic Transform, Pixelate, and JPEG Compression. According to Hendrycks & Dietterich (2019), models evaluated on CIFAR-C should avoid using identical augmentations as those represented in the benchmark.

AutoAugment (Cubuk et al., 2019) searches for various operations for data augmentation, as well as probabilities and magnitudes at which operations are applied. Through this, it identifies the optimal policy for models to achieve the highest validation accuracy on a given target dataset. The operations available for selection during the search, in AutoAugment, includes five geometric transforms (shear x/y, translate x/y, and rotate), two color transforms (color, invert), six intensity transforms (brightness, sharpness, solarize, equalize, autocontrast, contrast, and posterize), as well as cutout (DeVries & Taylor, 2017) and sample pairing (Zhang et al., 2017; Inoue, 2018). Some of these transformations may overlap with those in CIFAR-C.

To avoid this, AugMix (Hendrycks et al., 2019) integrates augmentations from AutoAugment, that do not overlap with the CIFAR-C benchmark (Hendrycks & Dietterich, 2019). Specifically, AugMix employs five geometric transforms (shear x/y, translate x/y, and rotate) and four intensity transforms (solarize, equalize, autocontrast, and posterize). However, ensuring a complete independence between augmentations is challenging. For example, (Rusak et al., 2020) and (Mintun et al., 2021) highlight the similarity between posterize and JPEG compression, as well as shear and translation to blur, respectively. Taking this into account, since we evaluate the robustness on CIFAR-C and also want to compare with AutoAugment and AugMix, we therefore, choose Plasma noise (Nicolaou et al., 2022), Planckian Jitter (Zini et al., 2022), and Gamma adjustment as augmentations to be disjoint from the three. Noteworthy, for the sake of complete comparisons between methods, in our implementation of AutoAugment for CIFAR-C evaluation, unlike AugMix, we do not remove overlapping augmentation like cutout (DeVries & Taylor, 2017), brightness, etc. Moreover, we conduct additional experiments by incorporating Augmix as the augmentation operation in LA. We denote these experiments as AugMix† in Tables 3 and 4. In case of adversarial training, we employ FGSM (Goodfellow et al., 2014) and 10-step iterative PGD (Madry et al., 2017) both with L_∞ constraints and $\varepsilon = 0.3$.

Label Smoothing and Multi-Task Learning. Label smoothing (LS) is an effective technique for regularizing DNNs. It achieves this by generating soft labels through a weighted average between a uniform distribution and the original hard labels. LS is typically used to address overfitting during training, leading to improved classification accuracy (Szegedy et al., 2016; Müller et al., 2019). While LS distributes the probability mass between the correct class label and all other classes, LA allocates the probability mass only between the identity class label and the augmentation labels. We compare the performance of LA against LS, where both techniques employ the same smoothing factor, δ. Furthermore, we compare LA with Multi-Task Learning (MTL), which offers an alternative approach to distinguishing between class identity and an indicator of augmentations. MTL involves optimizing a neural network across multiple related tasks simultaneously, aiming to enhance their generalization capabilities by leveraging shared patterns and representations among tasks (Standley et al., 2020; Zhang & Yang, 2021; Xin et al., 2022). We use a 40-2 Wide ResNet architecture as a shared feature extractor, along with two task-specific output heads: one for predicting class identity and the other for identifying the type of distortions applied to images. We employ δ as the weight for the task of augmentation predictions and $1 - \delta$ for the task of class identity prediction. The results of these experiments are presented in Table 5, for both CIFAR datasets.

4.2 Results

Comparisons of regular data augmentation versus LA. Table 1 compares standard training—where the model is trained on clean data and tested on clean data—with training via LA when using different numbers of operations for augmentations. The clean error improves when employing a single operation, while using two and three operations results in greater error reduction, up to 17.51% when employing Plasma and Gamma ($i.e.$ P.G.†, where notation † signifies label augmentation with factor δ). Similarly, we observe improvement in mCE when employing three operations. Specifically, P.G.† achieves a 22.70% improvement compared to the mCE of standard training. Introducing one additional operation, Jitter, results in a further improvement of 28.56% compared to the standard. Both calibration errors, ECE and RMS, show improvement when utilizing LA. However, values demonstrate no correlation between the number of operations employed and the improvement achieved. Similarly, errors under adversarial attacks FGSM and PGD improve, yet there is no apparent relationship between the number of operations employed and the observed enhancements. More details can be found in Fig. 4 A.

The results from repeating the experiment without using LA are shown in Table 2. In every instance, the Clean and calibration errors deteriorate. The mCE improves in most cases by up to 7.14% when using three operations. The adversarial error shows both improvements and deteriorations in different cases. Overall, there is no clear pattern between the number of operations and the fluctuation of errors.

Based on Tables 1 and 2, as well as Fig. 4 A, it is evident that LA outperforms standard augmentation in minimizing Clean, mCE, calibration, and adversarial errors. As P.G.† demonstrates the most significant enhancement in clean and PGD errors on Wide ResNet-50, while P.G.J.† show better improvements in mCE and calibration, we proceed with these two operations to compare LA with other augmentation methods across different networks.

Comparisons of other augmentations versus LA. Tables 3 and 4 summarize the results obtained by applying Mixup (Zhang et al., 2017), AugMix (Hendrycks et al., 2019), AutoAugment (Cubuk et al., 2019), adversarial training with FGSM (Goodfellow et al., 2014), 10-step iterative PGD (Madry et al., 2017), and LA for various networks, for CIFAR-10 and CIFAR-100 datasets, respectively.

Robustness enhancement for CIFAR-10. From the data presented in Table 3, clearly LA contributes to an improvement in Clean error across various architectures. When considering the average performance across the five networks, LA outperforms all other methods,with P.G.J.† improving the standard training by 23.29%. In terms of enhancing

robustness against common corruptions, there is a consistent improvement across AugMix, AutoAugment, FGSM and LA. However, depending on the employed networks, there is a variation in error rates for Mixup training, either showing a decrease or increase in comparison to the baseline. In all cases, PGD worsens both clean and mCE.

When comparing improvements in calibration errors, similar to the findings of (Wang et al., 2023a), our results show that mixup training tends to increase calibration errors compared to standard training. Similarly, on average, adversarial training negatively affects ECE error. While RMS improves with FGSM training by 4.95%, PGD diminishes it. Other methods all improve uncertainty estimates, among which, AutoAugment outperforms others, with reductions of 44.33% and 37.01% in ECE and RMS, respectively. Among LA training experiments, AugMix$^{+\!\!+}$ surpasses other augmentations with reductions in ECE and RMS by 33.65% and 31.15%, while P.G.J.$^{+\!\!+}$ reduces the calibration errors for both metrics by 2.18% and 7.65%. Under both ε budgets of 0.03 and 0.3, Mixup, FGSM, PGD, and all LA trainings show improvements. The enhancement in robustness from training with FGSM (ε=0.3) against FGSM attacks are 16.15% and 47.22%, compared to the standard adversarial error. However, training with FGSM barely improves error rates on PGD attack with $\varepsilon = 0.3$. In contrast, LA not only generalizes to both attacks but also outperforms adversarial training in both FGSM and PGD when considering P.G.J.$^{+\!\!+}$. More specifically, the robustness to FGSM and PGD improves by 61.24% and 49.54% at budget $\varepsilon = 0.03$ and 53.18% and 24.46% at budget $\varepsilon = 0.3$, respectively. More details on the percentages of improvement compared to standard training can be found in Table 6.

Robustness enhancement for CIFAR-100. Table 4 presents the performance of the aforementioned methods on the CIFAR-100 dataset. Except for Mixup and adversarial training methods, the remaining techniques all improve clean errors, compare to standard training. Specifically, AugMix, AutoAugment, P.G.$^{+\!\!+}$, P.G.J.$^{+\!\!+}$, and AugMix$^{+\!\!+}$ reduce the error by 1.58%, 4.49%, 3.24%, 4.31%, and 3.53%, respectively. In terms of calibration error, on average, almost all methods show improvements compared to standard training, except for PGD. For both low and high attack budgets, all methods improve the baseline except for AutoAugment and AugMix under PGD attacks. With $\varepsilon = 0.03$, the robustness gain from LA training exceeds that from FGSM and PGD training. Specifically, FGSM and PGD reduce FGSM error by 9.87% and 8.46%, respectively, whereas P.G.J.$^{+\!\!+}$ reduces it by 26.94%. Similarly, with $\varepsilon = 0.03$, LA outperforms adversarial training. FGSM and PGD reduce PGD error by 7.94% and 13.66%, respectively, whereas P.G.J.$^{+\!\!+}$ reduces it by 18.41%. However, with a higher budget of $\varepsilon = 0.3$, the gains from FGSM and PGD training for FGSM attacks are 28.29% and 24.40%, while P.G.J.$^{+\!\!+}$ achieves a gain of 16.93%. The robustness gains for PGD are 0.62%, 10.03%, and 10.38% for FGSM, PGD, and P.G.J.$^{+\!\!+}$, respectively. More details can be found in Table 7.

	Operations used							
Error	Std.	P.$^{+\!\!+}$	G.$^{+\!\!+}$	J.$^{+\!\!+}$	P.G.$^{+\!\!+}$	P.J.$^{+\!\!+}$	G.J.$^{+\!\!+}$	P.G.J.$^{+\!\!+}$
Clean	9.54	9.80	9.11	9.05	7.87	8.12	8.37	8.18
mCE	22.69	18.90	19.64	20.59	17.54	18.49	18.98	16.21
ECE	6.33	6.22	5.87	5.84	5.31	6.19	5.76	5.23
RMS	10.52	9.78	9.34	9.05	8.66	9.46	9.53	8.72
FGSM	69.13	49.76	49.65	38.73	39.33	32.37	44.39	34.61
PGD_{40}	94.82	82.77	81.18	83.44	69.71	73.65	77.08	67.89

Table 1: Performance comparisons of baseline to Label Augmentation with different operations on CIFAR-10 using the Wide ResNet-50 architecture. P.$^{+\!\!+}$, G.$^{+\!\!+}$, and J.$^{+\!\!+}$ refer to Plasma noise, Gamma Adjustment, and Planckian Jitter, respectively. The symbol $+\!\!+$ denotes the concatenation of labels with a factor of δ during these operations. Both adversarial training, FGSM and PGD use L_∞ constraints with $\varepsilon = 0.3$.

	Operations used							
Error	Std.	P.	G.	J.	P.G.	P.J.	G.J.	P.G.J.
Clean	9.54	10.36	10.49	10.11	10.75	10.23	9.76	10.17
mCE	22.69	21.66	22.21	22.98	21.25	21.71	23.11	21.07
ECE	6.33	8.12	8.44	8.03	8.22	8.27	8.33	8.61
RMS	10.52	13.10	13.54	12.67	12.55	12.96	12.87	13.74
FGSM	69.13	67.89	71.49	68.88	67.12	69.53	67.99	68.68
PGD_{40}	94.82	93.25	93.73	93.59	93.91	93.88	94.12	93.90

Table 2: The performance comparisons between the baseline and normal augmentation with different operations on CIFAR-10 using the Wide ResNet-50 architecture. P., G., and J. represent Plasma noise, Gamma Adjustment, and Planckian Jitter, respectively. Both adversarial training, FGSM and PGD use L_∞ constraints with $\varepsilon = 0.3$.

			Train								
			Std.	Mixup	AugMix	AutoAug.	FGSM	PGD_{10}	P.G.$^{\#}$	P.G.J.$^{\#}$	AugMix$^{\#}$
Calibration	Clean	LeNet	14.97±0.53	16.81±0.47	14.82±0.46	14.49±0.29	14.83±0.58	26.57±0.95	14.11±0.65	12.89±0.27	17.65±0.75
		ResNet	10.06±0.22	10.43±0.58	9.87±0.11	9.12±0.25	10.59±0.36	21.84±0.73	8.89±0.14	8.46±0.25	9.53±0.25
		ResNeXt	11.37±0.25	11.77±0.73	10.97±0.57	9.46±0.22	11.82±0.21	24.54±0.61	9.18±0.19	8.64±0.31	10.74±0.23
		WResNet	9.57±0.21	10.40±0.23	9.41±0.25	8.34±0.14	10.27±0.64	25.13±1.77	7.96±0.52	8.08±0.19	9.46±0.31
		SwinT	14.24±0.15	12.09±0.18	10.15±0.15	13.43±0.07	10.19±0.26	16.32±2.06	8.31±0.63	8.12±0.05	10.15±0.36
		Mean	12.04±0.27	12.30±0.44	11.04±0.31	10.97±0.19	11.54±0.41	22.88±1.22	9.69±0.43	9.24±0.21	11.51±0.38
	mCE	LeNet	28.24±0.36	28.21±0.81	23.84±0.64	25.15±0.31	28.03±0.44	40.66±0.42	24.05±0.27	23.71±0.09	30.88±0.45
		ResNet	24.71±0.29	22.95±0.57	18.71±0.05	19.47±0.15	22.40±0.71	37.49±0.78	18.56±0.15	18.52±0.22	23.71±0.33
		ResNeXt	24.83±0.34	23.86±0.19	18.75±0.22	19.12±0.18	22.28±0.73	40.50±0.39	18.51±0.11	18.10±0.23	26.45±0.71
		WResNet	22.73±0.27	21.55±0.27	17.01±0.28	17.89±0.27	20.99±0.32	42.58±2.53	16.64±0.39	16.78±0.71	22.75±0.89
		SwinT	23.99±0.14	24.13±0.22	18.61±0.17	23.96±0.10	20.97±0.39	24.27±1.38	17.20±0.38	17.22±0.11	18.61±0.87
		Mean	24.90±0.28	24.14±0.41	19.38±0.27	21.12±0.20	22.93±0.52	37.10±1.10	18.99±0.26	18.87±0.27	24.48±0.65
	ECE	LeNet	6.79±0.23	15.33±1.52	5.15±0.28	2.92±0.06	6.46±0.91	15.29±0.65	6.18±1.02	7.65±0.54	2.18±0.12
		ResNet	6.29±0.05	16.95±2.03	5.38±0.15	3.69±0.19	5.99±1.65	16.32±0.76	5.82±0.38	5.81±0.66	3.04±0.23
		ResNeXt	7.68±0.22	19.06±1.27	6.08±0.41	3.96±0.18	7.72±0.92	19.25±0.27	6.18±2.41	5.83±0.31	5.48±0.11
		WResNet	6.31±0.21	18.45±0.67	5.55±0.34	4.13±0.31	6.45±0.73	25.16±0.98	5.11±0.84	6.19±0.26	6.17±0.39
		SwinT	4.13±0.11	8.73±0.93	3.31±0.12	2.67±0.12	4.91±1.60	3.05±0.25	5.17±1.22	5.04±0.79	3.83±0.75
		Mean	6.24±0.16	15.70±1.28	5.09±0.26	3.47±0.17	6.31±1.16	15.81±0.58	5.69±1.17	5.90±0.51	4.14±0.32
	RMS	LeNet	9.64±0.97	16.00±1.83	7.61±0.58	4.76±0.32	8.92±1.92	18.09±0.78	9.31±0.71	9.49±0.57	3.67±0.19
		ResNet	10.33±0.22	17.49±2.45	8.87±0.25	6.57±0.28	9.56±2.55	19.97±0.35	9.92±0.83	9.18±0.63	5.03±0.28
		ResNeXt	12.06±0.31	19.80±1.39	10.21±0.79	7.04±0.42	11.51±1.03	22.71±0.52	10.02±0.35	9.56±0.77	8.35±0.26
		WResNet	10.37±0.38	19.21±1.05	9.22±0.71	7.71±0.39	10.23±0.78	28.56±0.82	8.56±0.31	8.96±0.43	9.91±0.23
		SwinT	6.48±0.13	9.66±0.89	6.52±0.51	4.71±0.17	6.24±1.33	8.85±0.31	9.01±0.45	7.95±0.55	6.52±0.59
		Mean	9.78±0.40	16.43±1.52	8.49±0.57	6.16±0.32	9.69±1.52	18.84±0.56	9.36±0.53	9.03±0.59	6.70±0.31
Adversarial ($\epsilon = 0.03$)	FGSM	LeNet	52.78±0.66	47.97±1.03	51.27±0.72	52.55±0.36	41.96±0.35	38.79±0.62	28.46±0.53	24.31±0.29	46.34±0.31
		ResNet	44.64±0.35	34.32±0.35	44.93±0.25	48.36±0.51	37.47±1.66	47.83±0.88	19.87±0.62	18.41±0.58	34.56±0.98
		ResNeXt	42.80±0.42	35.49±0.87	42.91±1.44	46.38±0.45	34.04±0.52	47.86±0.92	19.36±0.99	18.00±0.11	33.21±0.83
		WResNet	43.14±0.23	32.03±0.26	43.51±0.38	43.86±0.23	34.62±0.38	49.62±0.49	18.35±0.30	14.44±0.70	29.61±0.86
		SwinT	60.22±0.75	64.77±0.29	61.12±1.56	67.92±0.30	56.16±0.75	20.92±1.27	20.71±0.23	19.24±1.13	27.33±0.90
		Mean	48.72±0.48	42.92±0.56	48.75±0.87	51.81±0.37	40.85±0.73	41.00±0.84	21.35±0.53	18.88±0.56	34.21±0.78
	PGD_{40}	LeNet	75.38±0.46	72.20±1.46	73.34±0.85	73.94±0.36	61.55±0.22	53.27±0.18	41.87±0.73	37.84±0.32	59.10±0.99
		ResNet	78.05±0.16	63.15±0.49	78.59±0.69	81.43±0.72	70.82±0.25	70.11±1.23	41.84±0.54	37.65±0.69	60.51±0.85
		ResNeXt	76.11±0.42	66.02±0.78	75.58±0.77	81.23±0.33	66.11±0.12	68.90±0.96	37.28±0.95	35.31±0.49	53.21±0.73
		WResNet	77.08±0.18	62.63±0.31	77.89±0.51	80.49±0.51	68.04±0.37	71.02±0.97	39.78±0.78	37.81±0.37	48.36±0.82
		SwinT	72.71±0.57	87.26±0.65	67.60±1.09	83.08±1.05	71.44±1.01	21.66±1.07	49.70±0.93	42.81±0.84	67.60±0.88
		Mean	75.87±0.36	70.25±0.74	74.60±0.78	80.03±0.59	67.59±0.39	56.99±0.88	42.09±0.79	38.28±0.54	57.76±0.85
Adversarial ($\epsilon = 0.3$)	FGSM	LeNet	86.36±0.41	78.38±1.07	85.92±1.32	89.54±0.90	52.84±0.72	51.27±1.04	50.11±0.75	47.19±0.77	52.56±0.81
		ResNet	75.96±1.39	62.41±1.96	75.58±0.95	82.52±0.24	49.82±0.95	32.84±0.63	41.21±1.51	36.55±1.24	52.09±0.84
		ResNeXt	75.58±0.59	62.91±1.25	76.46±1.24	81.04±0.82	34.05±0.94	32.68±0.80	38.82±0.63	35.39±0.68	51.29±0.84
		WResNet	69.53±0.88	57.65±1.11	73.30±1.06	78.05±0.45	38.73±0.98	27.56±0.73	38.58±0.85	34.56±0.38	45.55±0.95
		SwinT	91.05±0.83	87.31±1.51	90.91±1.94	92.47±1.26	14.89±2.59	79.70±1.16	37.51±1.09	32.89±1.15	83.56±0.73
		Mean	79.70±0.82	69.73±1.38	80.43±1.30	84.72±0.73	38.07±1.24	44.81±0.87	41.25±0.97	37.32±0.84	57.01±0.83
	PGD_{40}	LeNet	92.34±0.76	91.14±2.94	95.58±0.48	93.21±0.32	92.43±0.68	81.01±1.96	68.03±0.33	65.66±0.83	83.76±1.25
		ResNet	94.68±0.41	82.66±1.68	94.60±0.35	95.39±0.21	94.37±0.31	81.07±1.72	70.50±0.31	66.05±1.27	76.84±1.20
		ResNeXt	94.31±0.44	86.32±1.52	93.94±1.28	95.41±0.12	93.54±0.77	78.74±1.05	65.44±0.82	63.20±0.90	71.84±0.98
		WResNet	94.78±0.53	83.34±0.73	94.79±0.32	95.53±0.59	94.04±0.75	77.35±1.32	69.20±0.35	67.99±1.53	71.17±1.53
		SwinT	92.71±0.82	94.79±1.21	95.06±1.04	94.71±2.35	94.81±1.83	89.03±2.51	94.41±0.56	91.24±1.81	90.91±1.57
		Mean	93.76±0.59	87.65±1.62	94.79±0.69	94.85±0.72	93.84±0.87	81.44±1.71	73.52±0.47	70.83±1.27	78.90±1.31

Table 3: Error rates of various methods across different architectures for CIFAR-10 dataset. LA improves Clean and mCE compared to standard training, and also exhibits superior robustness to adversarial examples and and generalization to attacks, even outperforming adversarial training.

			Train								
			Std.	Mixup	AugMix	AutoAug.	FGSM	PGD_{10}	P.G.$^{\#}$	P.G.J.$^{\#}$	AugMix$^{\#}$
Calibration	Clean	LeNet	39.84±0.31	41.55±0.54	39.41±0.66	38.89±0.34	41.93±0.58	56.99±0.89	38.80±0.68	39.22±0.42	40.07±0.87
		ResNet	31.10±0.25	32.24±0.53	30.87±0.52	29.91±0.32	33.97±0.36	47.47±0.63	30.67±0.61	30.15±0.47	29.72±0.62
		ResNeXt	32.95±0.27	34.89±0.68	32.03±0.53	29.69±0.32	35.66±0.21	51.02±0.76	30.69±0.72	30.01±0.36	30.92±0.73
		WResNet	29.43±0.19	31.38±0.37	28.91±0.72	27.32±0.37	32.26±0.64	42.41±2.01	28.62±0.85	28.59±0.31	27.45±0.58
		SwinT	27.92±0.17	29.05±0.29	27.47±0.32	27.47±0.23	30.93±0.26	34.94±1.99	27.23±0.99	26.32±0.25	27.39±0.70
		Mean	32.25±0.24	33.82±0.48	31.74±0.55	30.66±0.32	34.95±0.41	46.57±1.26	31.20±0.77	30.86±0.36	31.11±0.70
	mCE	LeNet	57.27±0.34	56.45±0.98	51.97±0.76	52.71±0.36	54.18±0.44	66.42±0.55	53.61±0.39	53.41±0.23	54.96±0.46
		ResNet	52.81±0.25	50.43±0.95	44.46±0.19	45.88±0.27	48.50±0.71	58.46±0.58	46.58±0.33	46.32±0.25	47.08±0.66
		ResNeXt	52.62±0.21	51.38±0.44	45.12±0.25	44.98±0.29	49.41±0.73	64.17±0.63	46.66±0.23	45.90±0.22	46.45±0.61
		WResNet	48.33±0.25	48.81±0.57	41.78±0.32	42.78±0.25	46.74±0.32	57.05±1.76	44.48±0.27	44.31±0.39	43.41±0.84
		SwinT	45.73±0.15	45.08±0.32	40.87±0.21	42.45±0.22	43.75±0.39	45.16±1.96	44.73±0.84	41.37±0.21	43.25±0.89
		Mean	51.35±0.24	50.43±0.65	44.84±0.35	45.76±0.28	48.52±0.52	58.25±1.10	47.21±0.41	46.26±0.26	47.03±0.69
	ECE	LeNet	15.79±0.21	10.20±0.98	11.49±0.31	6.37±0.21	15.57±0.91	25.32±1.02	14.08±0.96	13.47±0.54	10.61±0.24
		ResNet	18.40±0.23	9.59±1.71	15.76±0.31	12.32±0.23	15.57±1.65	29.32±1.25	15.51±0.42	14.15±0.73	12.73±0.32
		ResNeXt	20.83±0.17	11.35±0.85	17.52±0.34	12.81±0.27	21.64±0.92	33.70±1.03	15.49±1.56	14.24±0.65	12.87±0.56
		WResNet	18.51±0.33	10.10±0.77	16.54±0.36	13.08±0.25	19.76±0.73	27.52±1.11	16.97±1.01	16.41±0.39	11.88±0.80
		SwinT	14.14±0.35	6.36±1.06	11.39±0.32	7.31±0.19	14.64±1.60	19.37±0.72	14.02±0.88	12.12±0.87	11.09±0.72
		Mean	17.53±0.26	9.52±1.07	14.54±0.33	10.38±0.23	17.44±1.16	27.05±1.03	15.21±0.97	14.08±0.64	11.84±0.53
	RMS	LeNet	18.43±0.23	11.35±1.23	13.59±0.73	28.08±1.81	18.08±1.92	27.59±1.08	16.01±1.23	17.44±0.88	12.81±1.25
		ResNet	22.11±0.27	10.89±1.42	19.21±0.32	15.41±0.22	22.16±2.55	32.28±1.23	14.53±1.08	18.04±0.61	15.66±1.02
		ResNeXt	24.64±0.33	12.55±1.52	21.15±0.89	15.87±0.34	25.39±1.03	36.39±1.12	15.64±1.15	15.53±0.71	15.71±0.85
		WResNet	22.94±0.46	11.21±1.18	20.42±0.77	16.28±0.23	23.91±0.78	30.80±0.95	15.18±0.87	18.15±0.43	15.39±0.83
		SwinT	17.58±0.39	7.56±1.22	14.59±0.25	9.39±0.21	17.86±1.33	22.86±0.73	12.52±0.73	14.49±0.47	15.22±0.74
		Mean	21.14±0.34	10.71±1.31	17.79±0.59	27.41±0.26	21.48±1.52	29.98±1.02	14.78±1.01	16.53±0.62	14.96±0.94
Adversarial ($\epsilon = 0.03$)	FGSM	LeNet	69.41±0.25	67.51±0.96	68.71±0.87	70.62±0.52	63.28±0.35	64.11±2.03	56.95±0.66	57.16±0.85	64.98±1.12
		ResNet	64.16±0.44	58.07±0.94	63.03±0.85	65.54±0.48	57.62±1.66	64.93±1.32	46.78±0.73	46.31±0.39	51.76±1.11
		ResNeXt	62.55±0.47	58.67±0.73	63.41±0.99	63.77±0.53	56.74±0.52	64.52±1.16	46.81±0.52	46.04±0.91	50.17±0.84
		WResNet	61.97±0.42	55.38±0.82	62.16±1.83	63.52±0.58	55.94±0.38	62.29±1.62	48.01±0.65	43.46±0.62	48.27±1.03
		SwinT	75.35±0.32	74.78±0.88	75.23±2.03	78.44±0.89	66.96±0.75	49.39±1.91	47.12±0.49	50.64±0.88	49.19±0.98
		Mean	66.69±0.38	62.88±0.87	66.51±1.31	68.38±0.60	60.11±0.73	61.05±1.61	49.13±0.61	48.72±0.73	52.87±1.02
	PGD_{40}	LeNet	82.87±0.38	82.41±1.26	82.42±1.05	84.42±0.23	77.31±0.22	73.31±0.29	72.09±0.85	73.07±0.30	79.98±1.09
		ResNet	81.78±0.32	78.05±1.18	82.03±0.94	84.45±0.38	75.82±0.25	76.88±0.76	62.99±0.94	62.46±0.68	72.02±0.96
		ResNeXt	80.25±0.43	77.18±1.86	81.34±1.12	83.42±0.77	74.66±0.12	76.23±0.85	64.41±1.15	64.74±0.89	69.10±0.85
		WResNet	82.01±0.35	76.36±1.07	82.19±1.32	83.87±0.31	76.51±0.37	75.70±1.11	63.10±0.79	61.42±0.77	70.81±0.89
		SwinT	84.33±0.65	89.05±0.95	85.34±1.17	88.29±0.96	74.29±1.01	52.94±1.35	63.23±0.25	73.85±0.94	71.06±1.01
		Mean	82.25±0.43	80.61±1.26	82.66±1.12	84.89±0.53	75.72±0.39	71.01±0.87	65.16±0.80	67.11±0.72	72.59±0.96
Adversarial ($\epsilon = 0.3$)	FGSM	LeNet	91.92±0.57	90.73±1.21	89.97±1.52	92.91±0.87	73.54±0.72	69.49±1.25	81.85±1.14	81.85±0.86	87.48±0.87
		ResNet	88.24±0.78	84.96±1.23	85.58±1.23	87.94±0.36	69.55±0.95	63.27±1.06	75.39±1.37	73.61±0.95	76.77±0.91
		ResNeXt	87.21±0.79	84.69±1.38	82.93±1.43	85.71±0.39	69.68±0.94	63.29±1.03	74.69±0.98	70.92±0.73	78.05±1.03
		WResNet	85.46±0.82	83.94±1.63	82.46±1.47	84.64±0.52	64.95±0.98	60.71±1.23	71.62±1.01	67.86±0.79	74.27±0.95
		SwinT	91.38±0.99	89.87±1.65	90.09±1.86	92.53±0.99	40.83±2.59	79.08±1.17	73.91±0.96	74.75±1.06	74.57±0.84
		Mean	88.84±0.79	86.84±1.42	86.21±1.50	88.75±0.63	63.71±1.24	67.17±1.15	75.49±1.09	73.80±0.88	78.23±0.92
	PGD_{40}	LeNet	92.75±0.45	94.39±1.23	92.66±0.89	94.46±0.22	92.94±0.68	83.50±1.46	86.77±0.76	86.33±0.53	92.21±1.15
		ResNet	93.29±0.61	91.69±1.42	92.01±0.88	93.69±0.17	92.76±0.31	81.35±1.95	81.45±0.92	80.79±0.57	89.24±0.55
		ResNeXt	92.63±0.72	92.37±1.03	91.62±0.72	93.61±0.33	92.56±0.77	79.65±1.20	83.98±0.84	83.91±0.44	86.21±0.98
		WResNet	93.11±0.79	90.55±0.97	92.21±0.74	93.71±0.15	92.33±0.75	82.85±1.65	83.02±0.47	82.00±0.68	87.79±1.23
		SwinT	93.92±1.03	95.55±1.76	93.39±1.11	95.63±1.02	92.23±1.83	91.64±1.93	82.15±0.94	92.87±0.85	88.33±1.11
		Mean	93.14±0.72	92.91±1.28	92.38±0.87	94.22±0.38	92.56±0.87	83.80±1.64	83.47±0.79	85.18±0.61	88.76±1.00

Table 4: Error rates of various methods across different architectures for the CIFAR-100 dataset. LA improves Clean, mCE, and adversarial errors compared to standard training. Also, under lower-budget attacks, it consistently outperforms adversarial training.

Comparisons of LS and MTL versus LA. Table 5 presents a summary of the comparison between LS and MTL with LA. For CIFAR-10, the improvement in clean error is 5.22%, 3.66%, and 15.57% for LS, MTL, and LA, respectively. The ECE increases by 32.96% and 14.26% for LS and MTL, respectively, whereas LA enhances both ECE and RMS calibration error by up to 1.90%.

In both low and high ε budget attacks, LA exhibits superior improvement compared to LS and MTL, with enhancements of up to 50.29% and 28.27% for FGSM and PGD with $\varepsilon = 0.3$, respectively. Nevertheless, the effectiveness of LS in improving adversarial error for CIFAR-10 data is noticeable, and have been highlighted in findings of (Shafahi et al., 2019; Pang et al., 2020; Ren et al., 2021), as well.

Similarly, in CIFAR-100, LA outperforms in mCE improvement with 8.32% error reduction. In terms of adversarial errors, it improves the baseline by 29.87% for FGSM and 25.11% for PGD under a low budget $\varepsilon = 0.03$, as well as by 20.59% and 11.93% with a higher budget of $\varepsilon = 0.3$ under FGSM and PGD attacks, respectively.

	Err.	Train: Std.	Train: LS.	Train: MTL.	Train: P.G.J.$^{\#}$
	Clean	9.57±0.21	9.07±1.25	9.22±0.23	8.08±0.19
	mCE	22.73±0.27	21.89±1.79	20.59±0.58	16.78±0.71
	ECE	6.31±0.21	8.39±2.03	7.21±0.81	6.19±0.26
	RMS	10.37±0.38	10.92±1.88	11.17±0.25	8.96±0.43
$\varepsilon = 0.03$	FGSM	43.14±0.23	27.01±1.02	41.31±0.19	14.44±0.70
$\varepsilon = 0.03$	PGD	77.08±0.18	56.6±1.21	66.06±0.62	37.81±0.37
$\varepsilon = 0.3$	FGSM	69.53±0.88	52.36±2.66	71.38±0.55	34.56±0.38
$\varepsilon = 0.3$	PGD	94.78±0.53	82.43±1.54	93.36±0.73	67.99±1.53

CIFAR-10

	Err.	Train: Std.	Train: LS.	Train: MTL.	Train: P.G.J.$^{\#}$
	Clean	29.43±0.19	27.85±1.38	32.23±0.45	28.59±0.31
	mCE	48.33±0.25	47.11±1.91	48.32±0.56	44.31±0.39
	ECE	18.51±0.33	7.89±1.83	22.26±0.92	16.41±0.39
	RMS	22.94±0.46	8.78±1.45	26.79±0.33	18.15±0.43
$\varepsilon = 0.03$	FGSM	61.97±0.42	53.21±0.98	62.01±0.36	43.46±0.62
$\varepsilon = 0.03$	PGD	82.01±0.35	76.22±1.10	77.06±0.74	61.42±0.77
$\varepsilon = 0.3$	FGSM	85.46±0.82	80.44±1.55	84.04±0.92	67.86±0.79
$\varepsilon = 0.3$	PGD	93.11±0.79	91.66±2.03	90.16±0.65	82.00±0.68

CIFAR-100

Table 5: Performance comparisons of Label Smoothing and Multi-task Learning to LA (P.G.J.$^{\#}$) using the Wide ResNet-50 architecture. For both CIFAR-10 and CIFAR-100, LA improves Clean, mCE, Calibration, and adversarial errors compared to standard training and consistently outperforms LS and MTL in improving adversarial robustness.

5 Conclusion

To align our naming convention and label assignment when training the DNNs, we developed Label Augmentation. Essentially, LA assigns one-hot labels to each of the operations used during augmentations. Then, instead of solely augmenting transformed data in the training pipeline, LA involves augmenting labels by concatenating input labels with operation labels, using a factor of δ. This automatically enriches the labels without requiring extra human annotation and has proved to be advantageous in enhancing both robustness against common and adversarial perturbations. In terms of Clean and mCE error, comparative analysis shows LA performing nearly as well as AugMix and AutoAugment. However, in terms of adversarial robustness, LA is significantly better than other augmentation methods and can even outperform adversarial training. LA is flexible and could be employed in other modalities. For instance, future works can utilize LA in audio inputs while training with noisy audio signals. This study, alongside much of the existing research on distributional shift, primarily focuses on evaluating model robustness to 2D image transformations, largely overlooking changes in viewpoint within 3D transformations found in various real-world applications (e.g., autonomous driving). It has been demonstrated that common image classifiers are highly vulnerable to adversarial viewpoints (Dong et al., 2022). Future studies could explore whether employing LA—with rotation as augmentation—can enhance robustness against adversarial viewpoints.

6 Acknowledgement

This work was conducted with the financial support of the Science Foundation Ireland Centre for Research Training in Artificial Intelligence under Grant No. 18/CRT/6223. We would also like to thank the anonymous reviewers for their helpful and informative comments.

REFERENCES

Maksym Andriushchenko, Francesco Croce, Nicolas Flammarion, and Matthias Hein. Square attack: a query-efficient black-box adversarial attack via random search. In *European conference on computer vision*, pp. 484–501. Springer, 2020.

Anish Athalye, Nicholas Carlini, and David Wagner. Obfuscated gradients give a false sense of security: Circumventing defenses to adversarial examples. In *International conference on machine learning*, pp. 274–283. PMLR, 2018.

Aharon Azulay and Yair Weiss. Why do deep convolutional networks generalize so poorly to small image transformations? *arXiv preprint arXiv:1805.12177*, 2018.

Tao Bai, Jinqi Luo, Jun Zhao, Bihan Wen, and Qian Wang. Recent advances in adversarial training for adversarial robustness. *arXiv preprint arXiv:2102.01356*, 2021.

Yoshua Bengio, Aaron Courville, and Pascal Vincent. Representation learning: A review and new perspectives. *IEEE transactions on pattern analysis and machine intelligence*, 35(8):1798–1828, 2013.

Qingwen Bu, Dong Huang, and Heming Cui. Towards building more robust models with frequency bias. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 4402–4411, 2023.

Yi Cai, Xuefei Ning, Huazhong Yang, and Yu Wang. Ensemble-in-one: ensemble learning within random gated networks for enhanced adversarial robustness. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 37, pp. 14738–14747, 2023.

Yair Carmon, Aditi Raghunathan, Ludwig Schmidt, John C Duchi, and Percy S Liang. Unlabeled data improves adversarial robustness. *Advances in neural information processing systems*, 32, 2019.

Francesco Croce and Matthias Hein. Minimally distorted adversarial examples with a fast adaptive boundary attack. In *International Conference on Machine Learning*, pp. 2196–2205. PMLR, 2020a.

Francesco Croce and Matthias Hein. Reliable evaluation of adversarial robustness with an ensemble of diverse parameter-free attacks. In *International conference on machine learning*, pp. 2206–2216. PMLR, 2020b.

Ekin D Cubuk, Barret Zoph, Dandelion Mane, Vijay Vasudevan, and Quoc V Le. Autoaugment: Learning augmentation strategies from data. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 113–123, 2019.

Zhijie Deng, Xiao Yang, Shizhen Xu, Hang Su, and Jun Zhu. Libre: A practical bayesian approach to adversarial detection. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 972–982, 2021a.

Zhun Deng, Linjun Zhang, Amirata Ghorbani, and James Zou. Improving adversarial robustness via unlabeled out-of-domain data. In *International Conference on Artificial Intelligence and Statistics*, pp. 2845–2853. PMLR, 2021b.

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

Samuel Dodge and Lina Karam. Understanding how image quality affects deep neural networks. In *2016 eighth international conference on quality of multimedia experience (QoMEX)*, pp. 1–6. IEEE, 2016.

Yinpeng Dong, Shouwei Ruan, Hang Su, Caixin Kang, Xingxing Wei, and Jun Zhu. Viewfool: Evaluating the robustness of visual recognition to adversarial viewpoints. *Advances in Neural Information Processing Systems*, 35: 36789–36803, 2022.

Chanho Eom and Bumsub Ham. Learning disentangled representation for robust person re-identification. *Advances in neural information processing systems*, 32, 2019.

Nic Ford, Justin Gilmer, Nicolas Carlini, and Dogus Cubuk. Adversarial examples are a natural consequence of test error in noise. *arXiv preprint arXiv:1901.10513*, 2019.

Aviv Gabbay and Yedid Hoshen. Demystifying inter-class disentanglement. *arXiv preprint arXiv:1906.11796*, 2019.

Jakob Gawlikowski, Cedrique Rovile Njieutcheu Tassi, Mohsin Ali, Jongseok Lee, Matthias Humt, Jianxiang Feng, Anna Kruspe, Rudolph Triebel, Peter Jung, Ribana Roscher, et al. A survey of uncertainty in deep neural networks. *arXiv preprint arXiv:2107.03342*, 2021.

Robert Geirhos, Patricia Rubisch, Claudio Michaelis, Matthias Bethge, Felix A Wichmann, and Wieland Brendel. Imagenet-trained cnns are biased towards texture; increasing shape bias improves accuracy and robustness. *arXiv preprint arXiv:1811.12231*, 2018.

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

Sven Gowal, Sylvestre-Alvise Rebuffi, Olivia Wiles, Florian Stimberg, Dan Andrei Calian, and Timothy A Mann. Improving robustness using generated data. *Advances in Neural Information Processing Systems*, 34:4218–4233, 2021.

Chuan Guo, Geoff Pleiss, Yu Sun, and Kilian Q Weinberger. On calibration of modern neural networks. In *International conference on machine learning*, pp. 1321–1330. PMLR, 2017.

HarryKim. Torchattacks a pytorch library for adversarial attacks, 2020. URL https://adversarial-attacks-pytorch.readthedocs.io/en/latest/. Accessed: 2024-05-23.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 770–778, 2016.

Dan Hendrycks and Thomas Dietterich. Benchmarking neural network robustness to common corruptions and perturbations. *arXiv preprint arXiv:1903.12261*, 2019.

Dan Hendrycks, Norman Mu, Ekin D Cubuk, Barret Zoph, Justin Gilmer, and Balaji Lakshminarayanan. Augmix: A simple data processing method to improve robustness and uncertainty. *arXiv preprint arXiv:1912.02781*, 2019.

Minui Hong, Jinwoo Choi, and Gunhee Kim. Stylemix: Separating content and style for enhanced data augmentation. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 14862–14870, 2021.

Hiroshi Inoue. Data augmentation by pairing samples for images classification. *arXiv preprint arXiv:1801.02929*, 2018.

Philip TG Jackson, Amir Atapour Abarghouei, Stephen Bonner, Toby P Breckon, and Boguslaw Obara. Style augmentation: data augmentation via style randomization. In *CVPR workshops*, volume 6, pp. 10–11, 2019.

Klim Kireev, Maksym Andriushchenko, and Nicolas Flammarion. On the effectiveness of adversarial training against common corruptions. In *Uncertainty in Artificial Intelligence*, pp. 1012–1021. PMLR, 2022.

Alex Krizhevsky, Geoffrey Hinton, et al. Learning multiple layers of features from tiny images. 2009.

Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. *Advances in neural information processing systems*, 25, 2012.

Alexey Kurakin, Ian Goodfellow, and Samy Bengio. Adversarial machine learning at scale. *arXiv preprint arXiv:1611.01236*, 2016.

Balaji Lakshminarayanan, Alexander Pritzel, and Charles Blundell. Simple and scalable predictive uncertainty estimation using deep ensembles. *Advances in neural information processing systems*, 30, 2017.

Alfred Laugros, Alice Caplier, and Matthieu Ospici. Are adversarial robustness and common perturbation robustness independant attributes? In *Proceedings of the IEEE/CVF International Conference on Computer Vision Workshops*, pp. 0–0, 2019.

Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

Tao Li, Yingwen Wu, Sizhe Chen, Kun Fang, and Xiaolin Huang. Subspace adversarial training. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 13409–13418, 2022.

PyTorch Lightning. torchmetrics: Metrics for pytorch. https://lightning.ai/docs/torchmetrics/stable/classification/calibration_error.html, 2024. Accessed: April 5, 2024.

Jiashuo Liu, Zheyan Shen, Yue He, Xingxuan Zhang, Renzhe Xu, Han Yu, and Peng Cui. Towards out-of-distribution generalization: A survey. *arXiv preprint arXiv:2108.13624*, 2021a.

Ze Liu, Yutong Lin, Yue Cao, Han Hu, Yixuan Wei, Zheng Zhang, Stephen Lin, and Baining Guo. Swin transformer: Hierarchical vision transformer using shifted windows. In *Proceedings of the IEEE/CVF international conference on computer vision*, pp. 10012–10022, 2021b.

Raphael Gontijo Lopes, Dong Yin, Ben Poole, Justin Gilmer, and Ekin D Cubuk. Improving robustness without sacrificing accuracy with patch gaussian augmentation. *arXiv preprint arXiv:1906.02611*, 2019.

Ilya Loshchilov and Frank Hutter. Sgdr: Stochastic gradient descent with warm restarts. *arXiv preprint arXiv:1608.03983*, 2016.

Michal Lukasik, Srinadh Bhojanapalli, Aditya Menon, and Sanjiv Kumar. Does label smoothing mitigate label noise? In *International Conference on Machine Learning*, pp. 6448–6458. PMLR, 2020.

Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu. Towards deep learning models resistant to adversarial attacks. *arXiv preprint arXiv:1706.06083*, 2017.

Jan Hendrik Metzen, Tim Genewein, Volker Fischer, and Bastian Bischoff. On detecting adversarial perturbations. *arXiv preprint arXiv:1702.04267*, 2017.

Eric Mintun, Alexander Kirillov, and Saining Xie. On interaction between augmentations and corruptions in natural corruption robustness. *Advances in Neural Information Processing Systems*, 34:3571–3583, 2021.

Apostolos Modas, Rahul Rade, Guillermo Ortiz-Jiménez, Seyed-Mohsen Moosavi-Dezfooli, and Pascal Frossard. Prime: A few primitives can boost robustness to common corruptions. In *European Conference on Computer Vision*, pp. 623–640. Springer, 2022.

Seyed-Mohsen Moosavi-Dezfooli, Alhussein Fawzi, and Pascal Frossard. Deepfool: a simple and accurate method to fool deep neural networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 2574–2582, 2016.

Rafael Müller, Simon Kornblith, and Geoffrey E Hinton. When does label smoothing help? *Advances in neural information processing systems*, 32, 2019.

Anguelos Nicolaou, Vincent Christlein, Edgar Riba, Jian Shi, Georg Vogeler, and Mathias Seuret. Tormentor: Deterministic dynamic-path, data augmentations with fractals. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 2707–2711, 2022.

Yaniv Ovadia, Emily Fertig, Jie Ren, Zachary Nado, David Sculley, Sebastian Nowozin, Joshua Dillon, Balaji Lakshminarayanan, and Jasper Snoek. Can you trust your model's uncertainty? evaluating predictive uncertainty under dataset shift. *Advances in neural information processing systems*, 32, 2019.

Tianyu Pang, Chao Du, Yinpeng Dong, and Jun Zhu. Towards robust detection of adversarial examples. *Advances in neural information processing systems*, 31, 2018.

Tianyu Pang, Xiao Yang, Yinpeng Dong, Hang Su, and Jun Zhu. Bag of tricks for adversarial training. *arXiv preprint arXiv:2010.00467*, 2020.

Nicolas Papernot, Patrick McDaniel, and Ian Goodfellow. Transferability in machine learning: from phenomena to black-box attacks using adversarial samples. *arXiv preprint arXiv:1605.07277*, 2016a.

Nicolas Papernot, Patrick McDaniel, Xi Wu, Somesh Jha, and Ananthram Swami. Distillation as a defense to adversarial perturbations against deep neural networks. In *2016 IEEE symposium on security and privacy (SP)*, pp. 582–597. IEEE, 2016b.

Nicolas Papernot, Fartash Faghri, Nicholas Carlini, Ian Goodfellow, Reuben Feinman, Alexey Kurakin, Cihang Xie, Yash Sharma, Tom Brown, Aurko Roy, Alexander Matyasko, Vahid Behzadan, Karen Hambardzumyan, Zhishuai Zhang, Yi-Lin Juang, Zhi Li, Ryan Sheatsley, Abhibhav Garg, Jonathan Uesato, Willi Gierke, Yinpeng Dong, David Berthelot, Paul Hendricks, Jonas Rauber, and Rujun Long. Technical report on the cleverhans v2.1.0 adversarial examples library. *arXiv preprint arXiv:1610.00768*, 2018.

Yao Qin, Chiyuan Zhang, Ting Chen, Balaji Lakshminarayanan, Alex Beutel, and Xuezhi Wang. Understanding and improving robustness of vision transformers through patch-based negative augmentation. *Advances in Neural Information Processing Systems*, 35:16276–16289, 2022.

Qibing Ren, Liangliang Shi, Lanjun Wang, and Junchi Yan. Adversarial robustness via adaptive label smoothing. 2021.

Evgenia Rusak, Lukas Schott, Roland S Zimmermann, Julian Bitterwolf, Oliver Bringmann, Matthias Bethge, and Wieland Brendel. A simple way to make neural networks robust against diverse image corruptions. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part III 16*, pp. 53–69. Springer, 2020.

Ali Shafahi, Amin Ghiasi, Furong Huang, and Tom Goldstein. Label smoothing and logit squeezing: A replacement for adversarial training? *arXiv preprint arXiv:1910.11585*, 2019.

Connor Shorten and Taghi M Khoshgoftaar. A survey on image data augmentation for deep learning. *Journal of big data*, 6(1):1–48, 2019.

Trevor Standley, Amir Zamir, Dawn Chen, Leonidas Guibas, Jitendra Malik, and Silvio Savarese. Which tasks should be learned together in multi-task learning? In *International conference on machine learning*, pp. 9120–9132. PMLR, 2020.

Cecilia Summers and Michael J Dinneen. Improved mixed-example data augmentation. In *2019 IEEE winter conference on applications of computer vision (WACV)*, pp. 1262–1270. IEEE, 2019.

Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian Goodfellow, and Rob Fergus. Intriguing properties of neural networks. *arXiv preprint arXiv:1312.6199*, 2013.

Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jon Shlens, and Zbigniew Wojna. Rethinking the inception architecture for computer vision. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 2818–2826, 2016.

Florian Tramèr, Alexey Kurakin, Nicolas Papernot, Ian Goodfellow, Dan Boneh, and Patrick McDaniel. Ensemble adversarial training: Attacks and defenses. *arXiv preprint arXiv:1705.07204*, 2017a.

Florian Tramèr, Nicolas Papernot, Ian Goodfellow, Dan Boneh, and Patrick McDaniel. The space of transferable adversarial examples. *arXiv preprint arXiv:1704.03453*, 2017b.

Frederik Träuble, Elliot Creager, Niki Kilbertus, Francesco Locatello, Andrea Dittadi, Anirudh Goyal, Bernhard Schölkopf, and Stefan Bauer. On disentangled representations learned from correlated data. In *International Conference on Machine Learning*, pp. 10401–10412. PMLR, 2021.

Igor Vasiljevic, Ayan Chakrabarti, and Gregory Shakhnarovich. Examining the impact of blur on recognition by convolutional networks. *arXiv preprint arXiv:1611.05760*, 2016.

Deng-Bao Wang, Lanqing Li, Peilin Zhao, Pheng-Ann Heng, and Min-Ling Zhang. On the pitfall of mixup for uncertainty calibration. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 7609–7618, 2023a.

Haotao Wang, Chaowei Xiao, Jean Kossaifi, Zhiding Yu, Anima Anandkumar, and Zhangyang Wang. Augmax: Adversarial composition of random augmentations for robust training. *Advances in neural information processing systems*, 34:237–250, 2021.

Zekai Wang, Tianyu Pang, Chao Du, Min Lin, Weiwei Liu, and Shuicheng Yan. Better diffusion models further improve adversarial training. In *International Conference on Machine Learning*, pp. 36246–36263. PMLR, 2023b.

Dongxian Wu, Shu-Tao Xia, and Yisen Wang. Adversarial weight perturbation helps robust generalization. *Advances in neural information processing systems*, 33:2958–2969, 2020.

Saining Xie, Ross Girshick, Piotr Dollár, Zhuowen Tu, and Kaiming He. Aggregated residual transformations for deep neural networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 1492–1500, 2017.

Derrick Xin, Behrooz Ghorbani, Justin Gilmer, Ankush Garg, and Orhan Firat. Do current multi-task optimization methods in deep learning even help? *Advances in neural information processing systems*, 35:13597–13609, 2022.

Mingle Xu, Sook Yoon, Alvaro Fuentes, and Dong Sun Park. A comprehensive survey of image augmentation techniques for deep learning. *Pattern Recognition*, pp. 109347, 2023.

Weilin Xu, David Evans, and Yanjun Qi. Feature squeezing: Detecting adversarial examples in deep neural networks. *arXiv preprint arXiv:1704.01155*, 2017.

Huaxiu Yao, Yu Wang, Sai Li, Linjun Zhang, Weixin Liang, James Zou, and Chelsea Finn. Improving out-of-distribution robustness via selective augmentation. In *International Conference on Machine Learning*, pp. 25407–25437. PMLR, 2022.

Sergey Zagoruyko and Nikos Komodakis. Wide residual networks. *arXiv preprint arXiv:1605.07146*, 2016.

Hongyi Zhang, Moustapha Cisse, Yann N Dauphin, and David Lopez-Paz. mixup: Beyond empirical risk minimization. *arXiv preprint arXiv:1710.09412*, 2017.

Yu Zhang and Qiang Yang. A survey on multi-task learning. *IEEE Transactions on Knowledge and Data Engineering*, 34(12):5586–5609, 2021.

Zhun Zhong, Liang Zheng, Guoliang Kang, Shaozi Li, and Yi Yang. Random erasing data augmentation. In *Proceedings of the AAAI conference on artificial intelligence*, volume 34, pp. 13001–13008, 2020.

Simone Zini, Alex Gomez-Villa, Marco Buzzelli, Bartłomiej Twardowski, Andrew D Bagdanov, and Joost van de Weijer. Planckian jitter: countering the color-crippling effects of color jitter on self-supervised training. *arXiv preprint arXiv:2202.07993*, 2022.

A APPENDIX

The following includes results for evaluating with AutoAttack, employing a wider range of augmentation operations when using LA, and percentages of changes for both CIFAR-10 and CIFAR-100 datasets compared to the baseline.

The following Tables 6 and 7 presents the percentage of improvement of methods compared to the standard training, for CIFAR-10 and CIFAR-100, respectively. Next, table 10 summarizes the impact of incorporating additional operations and intensity levels.

			Train								
			Std.	Mixup	AugMix	AutoAug.	FGSM	PGD_{10}	P.G.$^{\#}$	P.G.J.$^{\#}$	AugMix$^{\#}$
Calibration	Clean	LeNet	0.00	12.29	-1.00	-3.21	-0.94	77.49	-5.74	-13.89	17.90
		ResNet	0.00	3.68	-1.89	-9.34	5.27	117.10	-11.63	-15.90	-5.27
		ResNeXt	0.00	3.52	-3.52	-16.80	3.96	115.83	-19.26	-24.01	-5.54
		WResNet	0.00	8.67	-1.67	-12.85	7.31	162.59	-16.82	-15.57	-0.94
		SwinT	0.00	-15.10	-28.72	-5.69	-28.44	14.61	-41.64	-42.98	-28.72
		Mean	0.00	2.14	-8.29	-8.92	-4.17	90.00	-19.53	-23.29	-4.42
	mCE	LeNet	0.00	-0.11	-15.58	-10.94	-0.74	43.98	-14.84	-16.04	9.35
		ResNet	0.00	-7.12	-24.28	-21.21	-9.35	51.72	-24.89	-25.05	-4.05
		ResNeXt	0.00	-3.91	-24.49	-23.00	-10.27	63.11	-25.45	-27.10	6.52
		WResNet	0.00	-5.19	-25.16	-21.29	-7.66	87.33	-26.79	-26.18	0.09
		SwinT	0.00	0.58	-22.43	-0.13	-12.59	1.17	-28.30	-28.22	-22.43
		Mean	0.00	-3.05	-22.15	-15.19	-7.90	49.00	-23.73	-24.23	-1.69
	ECE	LeNet	0.00	125.77	-24.15	-57.00	-4.86	125.18	-8.98	12.67	-67.89
		ResNet	0.00	169.48	-14.47	-41.34	-4.77	159.46	-7.47	-7.63	-51.67
		ResNeXt	0.00	148.18	-20.83	-48.44	0.52	150.65	-19.53	-24.09	-28.65
		WResNet	0.00	192.39	-12.04	-34.55	2.22	298.73	-19.02	-1.90	-2.22
		SwinT	0.00	111.38	-19.85	-35.35	18.89	-26.15	25.18	22.03	-7.26
		Mean	0.00	151.67	-18.37	-44.33	1.06	153.43	-8.78	-2.18	-33.65
	RMS	LeNet	0.00	65.98	-21.06	-50.62	-7.47	87.66	-3.42	-1.56	-61.93
		ResNet	0.00	69.31	-14.13	-36.40	-7.45	93.32	-3.97	-11.13	-51.31
		ResNeXt	0.00	64.18	-15.34	-41.63	-4.56	88.31	-16.92	-20.73	-30.76
		WResNet	0.00	85.25	-11.09	-25.65	-1.35	175.41	-17.45	-13.60	-4.44
		SwinT	0.00	49.07	0.62	-27.31	-3.70	36.57	39.04	22.69	0.62
		Mean	0.00	68.09	-13.20	-37.01	-4.95	100.86	-4.21	-7.65	-31.51
Adversarial ($\epsilon = 0.03$)	FGSM	LeNet	0.00	-9.11	-2.86	-0.44	-20.50	-26.51	-46.08	-53.94	-12.20
		ResNet	0.00	-23.12	0.65	8.33	-16.06	7.15	-55.49	-58.76	-22.58
		ResNeXt	0.00	-17.08	0.26	8.36	-20.47	11.82	-54.77	-57.94	-22.41
		WResNet	0.00	-25.75	0.86	1.67	-19.75	15.02	-57.46	-66.53	-31.36
		SwinT	0.00	7.56	1.49	12.79	-6.74	-65.26	-65.61	-68.05	-54.62
		Mean	0.00	-11.91	0.07	6.36	-16.15	-15.83	-56.17	-61.24	-29.78
	PGD_{40}	LeNet	0.00	-4.22	-2.71	-1.91	-18.35	-29.33	-44.45	-49.80	-21.60
		ResNet	0.00	-19.09	0.69	4.33	-9.26	-10.17	-46.39	-51.76	-22.47
		ResNeXt	0.00	-13.26	-0.70	6.73	-13.14	-9.47	-51.02	-53.61	-30.09
		WResNet	0.00	-18.75	1.05	4.42	-11.73	-7.86	-48.39	-50.95	-37.26
		SwinT	0.00	20.01	-7.03	14.26	-1.75	-70.21	-31.65	-41.12	-7.03
		Mean	0.00	-7.40	-1.67	5.49	-10.91	-24.88	-44.52	-49.54	-23.87
Adversarial ($\epsilon = 0.3$)	FGSM	LeNet	0.00	-9.24	-0.51	3.68	-38.81	-40.63	-41.98	-45.36	-39.14
		ResNet	0.00	-17.84	-0.50	8.64	-34.41	-56.77	-45.75	-51.88	-31.42
		ResNeXt	0.00	-16.76	1.16	7.22	-54.95	-56.76	-48.64	-53.18	-32.14
		WResNet	0.00	-17.09	5.42	12.25	-44.30	-60.36	-44.51	-50.29	-34.49
		SwinT	0.00	-4.11	-0.15	1.56	-61.68	-12.47	-58.80	-63.88	-8.23
		Mean	0.00	-12.50	0.93	6.31	-47.22	-43.77	-48.25	-53.18	-28.47
	PGD_{40}	LeNet	0.00	-1.30	3.51	0.94	0.10	-12.27	-26.33	-28.89	-9.29
		ResNet	0.00	-12.70	-0.08	0.75	-0.33	-14.37	-25.54	-30.24	-18.84
		ResNeXt	0.00	-8.47	-0.39	1.17	-0.82	-16.51	-30.61	-32.99	-23.83
		WResNet	0.00	-12.07	0.01	0.79	-0.78	-18.39	-26.99	-28.27	-24.91
		SwinT	0.00	2.24	2.53	2.16	2.27	-3.97	1.83	-1.59	-1.94
		Mean	0.00	-6.52	1.10	1.16	0.08	-13.14	-21.59	-24.46	-15.85

Table 6: Percentages of Error rates of various methods across different architectures for CIFAR-10 when compared to Standard training. Negative values indicate an improvement in error rates when employing augmentation techniques.

			Train								
			Std.	Mixup	AugMix	AutoAug.	FGSM	PGD_{10}	P.G.$^{\#}$	P.G.J.$^{\#}$	AugMix$^{\#}$
Calibration	Clean	LeNet	0.00	4.29	-1.08	-2.38	5.25	43.05	-2.61	-1.56	0.58
		ResNet	0.00	3.67	-0.74	-3.83	9.23	52.64	-1.38	-3.05	-4.44
		ResNeXt	0.00	5.89	-2.79	-9.89	8.22	54.84	-6.86	-8.92	-6.16
		WResNet	0.00	6.63	-1.77	-7.17	9.62	44.10	-2.75	-2.85	-6.73
		SwinT	0.00	4.05	-1.61	-1.61	10.78	25.14	-2.47	-5.73	-1.90
		Mean	0.00	4.88	-1.58	-4.94	8.38	44.40	-3.24	-4.31	-3.53
	mCE	LeNet	0.00	-1.43	-9.25	-7.96	-5.40	15.98	-6.39	-6.74	-4.03
		ResNet	0.00	-4.51	-15.81	-13.12	-8.16	10.70	-11.80	-12.29	-10.85
		ResNeXt	0.00	-2.36	-14.25	-14.52	-6.10	21.95	-11.33	-12.77	-11.73
		WResNet	0.00	0.99	-13.55	-11.48	-3.29	18.04	-7.97	-8.32	-10.18
		SwinT	0.00	-1.42	-10.63	-7.17	-4.33	-1.25	-2.19	-9.53	-5.42
		Mean	0.00	-1.80	-12.68	-10.89	-5.52	13.44	-8.06	-9.91	-8.42
	ECE	LeNet	0.00	-35.40	-27.23	-59.66	-1.39	60.35	-10.83	-14.69	-32.81
		ResNet	0.00	-47.88	-14.35	-33.04	-15.38	59.35	-15.71	-23.10	-30.82
		ResNeXt	0.00	-45.51	-15.89	-38.50	3.89	61.79	-25.64	-31.64	-38.21
		WResNet	0.00	-45.43	-10.64	-29.34	6.75	48.68	-8.32	-11.35	-35.82
		SwinT	0.00	-55.02	-19.45	-48.30	3.54	36.99	-0.85	-14.29	-21.57
		Mean	0.00	-45.71	-17.08	-40.81	-0.56	54.25	-13.23	-19.71	-32.50
	RMS	LeNet	0.00	-38.42	-26.26	52.36	-1.90	49.70	-13.13	-5.37	-30.49
		ResNet	0.00	-50.75	-13.12	-30.30	0.23	46.00	-34.28	-18.41	-29.17
		ResNeXt	0.00	-49.07	-14.16	-35.59	3.04	47.69	-36.53	-36.97	-36.24
		WResNet	0.00	-51.13	-10.99	-29.03	4.23	34.26	-33.83	-20.88	-32.91
		SwinT	0.00	-57.00	-17.01	-46.59	1.59	30.03	-28.78	-17.58	-13.42
		Mean	0.00	-49.33	-15.84	-19.56	1.61	41.84	-30.10	-20.86	-29.24
Adversarial ($\epsilon = 0.03$)	FGSM	LeNet	0.00	-2.74	-1.01	1.74	-8.83	-7.64	-17.95	-17.65	-6.38
		ResNet	0.00	-9.49	-1.76	2.15	-10.19	1.20	-27.09	-27.82	-19.33
		ResNeXt	0.00	-6.20	1.37	1.95	-9.29	3.15	-25.16	-26.39	-19.79
		WResNet	0.00	-10.63	0.31	2.50	-9.73	0.52	-22.53	-29.87	-22.11
		SwinT	0.00	-0.76	-0.16	4.10	-11.13	-34.45	-37.47	-32.79	-34.72
		Mean	0.00	-5.71	-0.27	2.53	-9.87	-8.46	-26.32	-26.94	-20.71
	PGD_{40}	LeNet	0.00	-0.56	-0.54	1.87	-6.71	-11.54	-13.01	-11.83	-3.49
		ResNet	0.00	-4.56	0.31	3.26	-7.29	-5.99	-22.98	-23.62	-11.93
		ResNeXt	0.00	-3.83	1.36	3.95	-6.97	-5.01	-19.74	-19.33	-13.89
		WResNet	0.00	-6.89	0.22	2.27	-6.71	-7.69	-23.06	-25.11	-13.66
		SwinT	0.00	5.60	1.20	4.70	-11.91	-37.22	-25.02	-12.43	-15.74
		Mean	0.00	-1.99	0.51	3.21	-7.94	-13.66	-20.77	-18.41	-11.74
Adversarial ($\epsilon = 0.3$)	FGSM	LeNet	0.00	-1.29	-2.12	1.08	-20.00	-24.40	-10.96	-10.96	-4.83
		ResNet	0.00	-3.72	-3.01	-0.34	-21.18	-28.30	-14.56	-16.58	-13.00
		ResNeXt	0.00	-2.89	-4.91	-1.72	-20.10	-27.43	-14.36	-18.68	-10.50
		WResNet	0.00	-1.78	-3.51	-0.96	-24.00	-28.96	-16.19	-20.59	-13.09
		SwinT	0.00	-1.65	-1.41	1.26	-55.32	-13.46	-19.12	-18.20	-18.40
		Mean	0.00	-2.26	-2.97	-0.11	-28.29	-24.40	-15.03	7.00	-11.95
	PGD_{40}	LeNet	0.00	1.77	-0.10	1.84	0.20	-9.97	-6.45	-6.92	-0.58
		ResNet	0.00	-1.72	-1.37	0.43	-0.57	-12.80	-12.69	-13.40	-4.34
		ResNeXt	0.00	-0.28	-1.09	1.06	-0.08	-14.01	-9.34	-9.41	-6.93
		WResNet	0.00	-2.75	-0.97	0.64	-0.84	-11.02	-10.84	-4.77	-5.71
		SwinT	0.00	1.74	-0.56	1.82	-1.80	-2.43	-12.53	-1.12	-5.95
		Mean	0.00	-0.25	-0.82	1.16	-0.62	-10.03	-10.38	-7.11	-4.71

Table 7: Percentages of Error rates of various methods across different architectures for CIFAR-100 when compared to Standard training. Negative values indicate an improvement in error rates when employing augmentation techniques.

The impact of incorporating additional operations and intensity levels.

Table 10 presents the results obtained by employing additional augmentation operations and intensity levels while utilizing LA.

	Err.	Train Std.	Train LA.
	Clean	9.57±0.21	6.79±1.72
	mCE	22.73±0.27	13.72±1.07
	ECE	6.31±0.21	5.23±0.66
	RMS	10.37±0.38	9.26±0.40
$\varepsilon = 0.03$	FGSM	43.14±0.23	14.15±4.24
$\varepsilon = 0.03$	PGD	77.08±0.18	35.04±0.44
$\varepsilon = 0.3$	FGSM	69.53±0.88	25.89±5.69
$\varepsilon = 0.3$	PGD	94.78±0.53	50±3.20
	AA	86.05±-	72.07±-

Table 8: **CIFAR-10**

	Err.	Train Std.	Train LA.
	Clean	29.43±0.19	27.74±1.26
	mCE	48.33±0.25	40.52±1.67
	ECE	18.51±0.33	18.13±0.77
	RMS	22.94±0.46	22.37±0.59
$\varepsilon = 0.03$	FGSM	61.97±0.42	39.65±3.25
$\varepsilon = 0.03$	PGD	82.01±0.35	56.82±1.04
$\varepsilon = 0.3$	FGSM	85.46±0.82	56.67±2.88
$\varepsilon = 0.3$	PGD	93.11±0.79	71.36±2.75
	AA	93.97±-	82.22±-

Table 9: **CIFAR-100**

Table 10: The error rates of Wide ResNet-50 architectures when employing LA with seven augmentation types: Plasma, Gamma, PlanckianJitter, ColorJiggle, Equalize, Posterize, and Rain across three severity levels. The parameter δ corresponds to the intensity of the added noise, setting $\delta = 0.8, 0.6, 0.2$, for high, moderate, and slight noise. Similar trends in error rate improvements are observed with additional augmentation types. While improving the baseline error, in some cases, it exhibit a lower degree of improvement compared to employing fewer numbers of operations.

The performance under AutoAttack.

AutoAttack (Croce & Hein, 2020b) ensembles two parameter-free versions of the PGD attack, along with the Fast Adaptive Boundary attack (Croce & Hein, 2020a) and the Square Attack (Andriushchenko et al., 2020), to create a diverse testing framework. We evaluate using AutoAttack with L_∞ constraints at $\varepsilon = 0.03$. We use the implementations provided by the *TorchAttack* 3.5.1 library (HarryKim, 2020).

Tables 11, 12, and 13 present a summary of the performance of various training methods under AutoAttack across both the CIFAR-10 and CIFAR-100 datasets. While the robustness of the methods falls below their performance under FGSM or PGD attacks, LA consistently outperforms all other methods. Specifically, in comparison to standard training error, Mixup and PGD worsen it by 1.02% and 6.29%, respectively, while Augmix, AutoAugment, FGSM, PGD, P.G.$^+$, P.G.J.$^+$, and AugMix$^+$ improve it by 2.02%, 1.66%, 4.44%, 7.61%, 11.76%, and 13.76%, respectively, for CIFAR-10. For CIFAR-100, PGD shows a slight increase of 0.19%, whereas all other methods improve it by 0.73%, 1.63%, 1.78%, 1.49%, 6.68%, 5.64%, and 11.93% for Mixup, Augmix, AutoAugment, FGSM, P.G.$^+$, P.G.J.$^+$, and AugMix$^+$, respectively. In LS and MTL, the improvement under AA are 2.12% and 4.37% for CIFAR-10 and 1.11% and 1.49% for CIFAR-100, respectively.

	Train								
	Std.	Mixup	AugMix	AutoAug.	FGSM	PGD_{10}	P.G.$^+$	P.G.J.$^+$	AugMix$^+$
CIFAR-10	86.05	86.93	84.31	84.62	82.23	91.46	79.50	75.93	74.21
CIFAR-100	93.97	93.28	92.44	92.30	92.57	94.15	87.69	88.67	82.76

Table 11: The error rates of different methods using Wide ResNet-50 architectures for the CIFAR-10 dataset under AutoAttack with L_∞ constraints at $\varepsilon = 0.03$.

	Train			
Err.	Std.	LS.	MTL.	P.G.J.$^+$
AA	86.05	84.23	82.29	75.93

Table 12: **CIFAR-10** The error rates of MTL, LS, and LA methods using Wide ResNet-50 architectures for the CIFAR-10 dataset under AutoAttack with L_∞ constraints at $\varepsilon = 0.03$.

	Train			
Err.	Std.	LS.	MTL.	P.G.J.$^+$
AA	93.97	92.93	92.57	88.67

Table 13: **CIFAR-100** The error rates of MTL, LS, and LA methods using Wide ResNet-50 architectures for the CIFAR-100 dataset under AutoAttack with L_∞ constraints at $\varepsilon = 0.03$.

Percentages of error rate variations compared to the standard training.

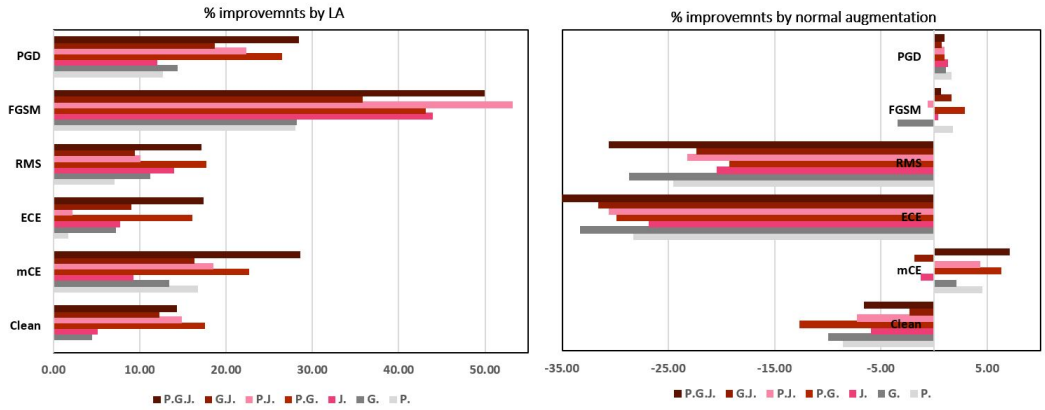

Figure 4: Percentages of error rate variations compared to the standard training on Wide ResNet-50: left side employing LA, right side with normal augmentations. While normal augmentation can enhance mCE to a considerable degree, it comes at the expense of Clean and calibration errors. On the other hand, regardless of the type and the number of operations used in augmenting with LA, we can see improvements in Clean, mCE, calibration, and adversarial errors. However, using two or three types of operations proves even more effective.