Decision Boundary-aware Data Augmentation for Adversarial Training

Chen Chen, Jingfeng Zhang, *Member, IEEE*, Xilie Xu, Lingjuan Lyu*, *Member, IEEE*, Chaochao Chen, Tianlei Hu, and Gang Chen*, *Member, IEEE*

Abstract—*Adversarial training* (AT) is a typical method to learn adversarially robust deep neural networks via training on the *adversarial variants* generated by their natural examples. However, as training progresses, the training data becomes less *attackable*, which may undermine the enhancement of model robustness. A straightforward remedy is to incorporate more training data, but it may incur an unaffordable cost. To mitigate this issue, in this paper, we propose a deCisiOn bounDary-aware data Augmentation framework (CODA): in each epoch, the CODA directly employs the meta information of the previous epoch to guide the augmentation process and generate more data that are close to the decision boundary, i.e., attackable data. Compared with the *vanilla mixup*, our proposed CODA can provide a higher ratio of attackable data, which is beneficial to enhance model robustness; it meanwhile mitigates the model's linear behavior between classes, where the linear behavior is favorable to the *standard training for generalization* but not to the *adversarial training for robustness*. As a result, our proposed CODA encourages the model to predict invariantly in the cluster of each class. Experiments demonstrate that our proposed CODA can indeed enhance adversarial robustness across various adversarial training methods and multiple datasets.

Index Terms—adversarial robustness, data augmentation

1 Introduction

Deep neural networks (DNNs) trained with a standard learning procedure are vulnerable to *adversarial examples* [1], [2], [3], [4]. *Adversarial training* (AT) is one of the most effective strategies for enhancing the model robustness against the adversarial examples [5]. To obtain *adversarial robustness*, AT methods alternatively generate adversarial data and optimize model parameters on the generated adversarial data [5], [6], [7], [8], [9], [10], [11], [12], [13], [14], [15], [16], [17], [18], [19], [20].

Recent studies on AT suggested the unequal treatment of data, as data usually do not contribute equally during adversarial training [7], [8], [21], [22]. In particular, [22] divided the training data into two categories—*attackable data* and *guarded data*, in which attackable/guarded data are close to/far away from the (class) *decision boundary* that can/cannot be attacked. To enhance adversarial robustness, attackable data are particularly useful in learning the decision boundary [8], [22].

However, as AT progresses, the ratio of attackable data decreases significantly, which may jeopardize the enhancement of adversarial robustness. We train a typical AT method, i.e., standard adversarial training (SAT)[1] [5] and report the ratio of attackable data in Fig. 1(a) (blue solid line). After 30-th epoch (with a reduced learning rate), the ratio of attackable data drops rapidly. This is because the model fine-tunes the decision boundary, and thus more and more training data become guarded. Meanwhile, as shown in Fig. 1(b), after 30-th epoch, the robust accuracy of SAT (blue line) ceases to rise and begins to drop. This strong correlation between this ratio and the robustness urges us to introduce more attackable data for AT.

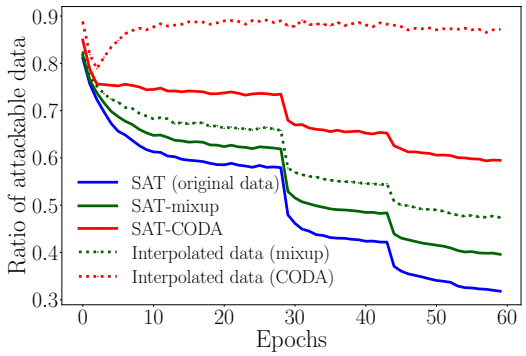
A straightforward remedy for the shortage of attackable data is to incorporate more training data [25]. [25] and [26] showed that to learn a robust model, AT generally requires substantially more training data than *standard training* (ST). Nevertheless, gathering additional data especially with high-quality labels is often expensive; therefore, [27], [28], and [29] leveraged a massive amount of unlabeled data for training adversarially robust models. However, in many privacy-sensitive applications such as biomedical [30] or financial [31] domains, collecting massive unlabeled data is very costly and sometimes not possible at all [32].

To mitigate this shortage, this paper proposes a novel data augmentation framework— *deCisiOn bounDary-aware data Augmentation*(CODA)—for AT. Different from the previous work that requires collecting massively additional data, our proposed CODA directly employs the meta information of the previous epoch to guide the augmentation process and generates more data that are close to the decision boundary, i.e., attackable data. Then, the augmented data can be used for the model training of current epoch (details can be referred to Section 3).

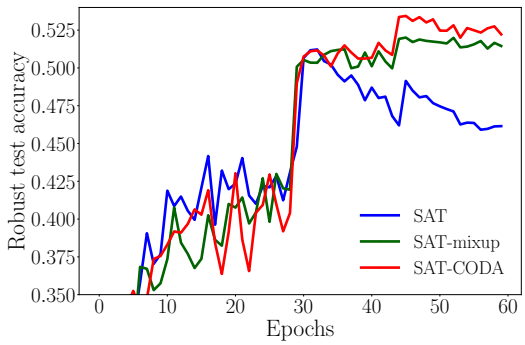
We remark that our CODA is sophisticatedly designed

- *C. Chen, C. Chen, and T. Hu are with the College of Computer Science and Technology, Zhejiang University, Hangzhou 310027, China. E-mail:{cc33, zjuccc, htl}@zju.edu.cn.*
- *J. Zhang is with the RIKEN Center for Advanced Intelligence Project (AIP), Tokyo 103-0027, Japan. E-mail: jingfeng.zhang@riken.jp.*
- *X. Xu is with the School of Computing, National University of Singapore, Singapore 117417. E-mail: e0792471@u.nus.edu.*
- *L. Lyu is with Sony AI, Tokyo, 108-0075 Japan. E-mail: lingjuanlvsmile@gmail.com.*
- *G. Chen is with the State Key Laboratory of CAD&CG, College of Computer Science and Technology, Zhejiang University, Hanghou 310027, China. E-mail: cg@zju.edu.cn.*
- *Chen Chen and Jingfeng Zhang contributed equally to this work.*
- *Lingjuan Lyu and Gang Chen are the corresponding authors.*

1. The detail of SAT is shown in Section 2.2.



(a) Ratio of attackable data of different methods.



(b) Robust test accuracy of different methods.

Fig. 1. The effectiveness of involving more attackable data. We train Standard adversarial training (SAT) [5], SAT-mixup (SAT combined with vanilla mixup), and SAT-CODA (SAT combined with CODA) with ResNet-18 [23] on the CIFAR-10 dataset [24]. The detailed settings are in Section 4.5.1. (a): The attackable data ratio of SAT (blue solid line) decreases rapidly as training progresses. Mixup (green dotted line) can increase the ratio of attackable data but still cannot keep a high ratio of attackable data after Epoch 30. CODA (red dotted line) can always generate a high ratio of attackable data (nearly 90%). (b): The robust test accuracy is evaluated under PGD-20 attack. All three methods have similar robust test accuracy before Epoch 30, due to that all three methods have a high ratio of attackable data. After Epoch 30 (with a reduced learning rate), the robust test accuracy of SAT drops significantly, which is strongly related to the low ratio of attackable data. SAT-CODA achieves the highest robust test accuracy.

for AT, with the objective to mitigate a *model's linear behavior between classes*. The model's linear behavior is first introduced by the vanilla mixup [33]; it encourages the model predictions to transit linearly from class to class, in order to provide a smooth change of prediction confidence [33] (upper two panels of Fig. 3). However, this linear behavior is favorable to ST for the generalization (and weak robustness against FGSM attack [34]) but not to AT for the strong robustness against the adaptive attacks. AT encourages the model to be locally constant within the input's neighborhood [3], [35], thus encouraging invariant predictions within the cluster of each class. To this end, our CODA mitigates the defect of vanilla mixup (i.e., introducing the linear behavior) and encourages sharp transitions between classes (lower two panels of Fig. 3); it meanwhile can generate more attackable data, thus further enhancing model robustness.

We summarize our contributions as follows.

- To the best of our knowledge, we are the first to show that the vanilla mixup [33] introduces the model's linear behavior between classes, which may adversely affect adversarial robustness (Section 2.5).
- We design a novel deCisiOn bounDary-aware data Augmentation framework (CODA) for adversarial training (AT) (Section 3). To enhance adversarial robustness, CODA provides more data close to the decision boundary, i.e., attackable data. Moreover, CODA mitigates the undesirable linear behaviors in mixup-based methods, which can further improve robust accuracy.
- Our CODA is a compatible approach, which can be combined with any existing adversarial training methods, e.g., Standard adversarial training (SAT) [5], TRADES [10], GAIRAT [22], and FastAT [17].
- Experiments on three real-world datasets and adversarial training methods corroborate the efficacy of our CODA in enhancing adversarial robustness (Section 4). For example, by combining CODA with GAIRAT (GAIRAT-CODA), we can improve the robust accuracy of the original GAIRAT by 8.28% under PGD-20 attack on CIFAR-10 dataset.

2 Notation and Related Work

Section 2.1 introduces the notation used in this paper. Section 2.2 reviews a typical AT method, i.e., standard adversarial training (SAT) by [5]. Section 2.3 reviews several image transformation-based methods that can enhance AT. Section 2.4 reviews several recent AT methods that claimed the unequal treatment of data. Section 2.5 reviews the mixup [33] and its relation with robustness. We also highlight the differences between our CODA and existing mixup-based adversarial training techniques [36], [37], [38].

2.1 Notation

Let (x, d_∞) be the input feature space $\mathcal{X}$ with the infinity distance metric $d_\infty(x, x') = \|x - x'\|_\infty$ and $\mathcal{B}_\epsilon(x) = \{x' \in \mathcal{X} | d_\infty(x, x') < \epsilon\}$ be the closed ball of radius $\epsilon > 0$ centered at x in $\mathcal{X}$. Let $S = \{(x_i, y_i)\}_{i=1}^n$ be the original dataset, where $x_i \in \mathcal{X}$ is the input, $y_i \in \{0,1\}^C$ is the one-hot label of x_i, and C is the number of classes. Let $f_\theta(\cdot) \in [0,1]^C$ be a score function with parameters θ, K be the maximum number of steps in PGD, and α be the step size of projected gradient descent (PGD). Let $f_\theta^l(\cdot) \in [0,1]$ be the l-th element of the model's prediction $f_\theta(\cdot)$, and $y^l \in \{0,1\}$ be the l-th element of the one-hot label y. We use a tilde over a natural data x to denote its adversarial variant $\tilde{x}$. We use a line over a pair of input and label (e.g., $(\overline{x},\overline{y})$) to denote an interpolated data where $\overline{x}$ is the input and $\overline{y} \in [0,1]^C$ is the soft label. Let λ_{mi} be the interpolation weight for mixup (or SAT-mixup), and λ be the interpolation weight for CODA (or CODA-based methods).

2.2 Standard adversarial training

The objective function of standard adversarial training (SAT) [5] can be expressed as:

$$\min_{f_\theta \in \mathcal{F}} \frac{1}{n} \sum_{i=1}^{n} \ell(f_\theta(\tilde{x}_i), y_i), \tag{1}$$

where

$$\tilde{x}_i = \arg\max_{\tilde{x}\in\mathcal{B}_\epsilon(x_i)} \ell(f_\theta(\tilde{x}), y_i). \tag{2}$$

$\tilde{x}_i$ is the most adversarial data within the ϵ-ball centered at x. $f_\theta(\cdot) : \mathcal{X} \to [0,1]^C$ is a score function with parameters θ. $\ell(\cdot,\cdot)$ is the loss function (e.g., the cross-entropy loss). Adversarial training can be divided into two steps: (1) the first step maximizes the loss to find adversarial data; (2) the second step minimizes the loss on the adversarial data w.r.t. the parameters θ.

SAT uses projected gradient descent (PGD) to approximate Eq. (2) and generate adversarial data. Given a starting point $x^{(0)} \in \mathcal{X}$, step size $\alpha > 0$, radius $\epsilon > 0$, and maximum number of steps K, PGD searches adversarial data as follows:

$$x^{(k+1)} = \Pi_{\mathcal{B}_\epsilon(x^{(0)})}\left(x^{(k)} + \alpha\,\mathrm{sign}(\nabla_{x^{(k)}}\ell(f_\theta(x^{(k)}), y))\right), \tag{3}$$

where $k = 0, \ldots, K-1$ is the step number, $x^{(0)}$ refers to natural data or natural data perturbed by a small Gaussian or uniform random noise, y is the corresponding label for the natural data, $x^{(k)}$ is the adversarial data at step k, and $\Pi_{\mathcal{B}_\epsilon(x^{(0)})}(\cdot)$ projects the adversarial data onto the ϵ-ball centered at x^0. Eq. (3) increases the loss by gradient ascent and then projects the updated adversarial data back onto the ϵ-ball centered at x^0.

After K steps of PGD, SAT utilizes the generated adversarial data x^K to update the model parameters θ. For evaluation, we use *robust accuracy*, which is the fraction of predictions that the model correctly makes on the adversarial data generated by the adversarial attacks such as the PGD attack (in Eq. (3)).

2.3 Image transformation in AT

There are several studies [38], [39], [40], [41], [42], [43], [44] that aims to handle AT by image transformation. [39] proposed to use dimensionality reduction techniques such as principal component analysis (PCA) to defend against adversarial attacks. However, such techniques are not applicable in modern CNN models. [40] and [41] enhanced robustness by compressing the input images. Nevertheless, compressing methods reduce the resolution of input images, which may affect the natural accuracy (accuracy on natural data). [42] and [43] introduced noise to defend against adversarial attacks. [44] random resized the input images and added random padding to the input images. [38] utilized mixup [33] to improve robustness. However, these methods [38], [42], [43], [44] dealt with adversarial attacks in inference time, while our work concentrates on improving robustness in the training phase.

2.4 Unequal contribution of data in AT

Recent studies [8], [21], [22], [45] showed that data do not contribute equally during adversarial training. Max-margin adversarial training (MMA) [21] demonstrated that adversarial data close to the model's decision boundary are more important to the robust model, while adversarial data far from the decision boundary are less important. Misclassification aware adversarial training (MART) [8] differentiated the misclassified and correctly classified data during AT. Customized adversarial training (CAT) [45] adaptively customized the perturbation bound and the corresponding label for each training data according to its distance to the decision boundary. Geometry-aware instance-reweighted adversarial training (GAIRAT) [22] treated data unequally and paid more attention to data close to the decision boundary. Specifically, GAIRAT splited all the data into two categories—*attackable data* and *guarded data*. Data close to the *decision boundary* are called attackable data, while guarded data are far from the decision boundary. Formally, a sample (x, y) is said to be attackable, if

$$\arg\max_c f_\theta^c(\tilde{x}) \neq \arg\max_c y^c, \tag{4}$$

where $\tilde{x}$ is the adversarial variant of natural data x, $\arg\max_c f_\theta^c(\tilde{x})$ is the predicted class of $\tilde{x}$, and $\arg\max_c y^c$ is the ground truth class of x. On the other hand, a sample (x, y) is said to be guarded, if

$$\arg\max_c f_\theta^c(\tilde{x}) = \arg\max_c y^c. \tag{5}$$

The adversarial variants of attackable data can be easily misclassified by model $f_\theta(\cdot)$, while the adversarial variants of the guarded data are more difficult to be misclassified. Note that instead of the distance of the data to the decision boundary, GAIRAT classified the data by whether it can be misclassified or not, which is more efficient than computing the distance. GAIRAT claimed that the attackable data can help improve adversarial training, while the guarded data are less important.

Motivated by GAIRAT, in this paper, we propose a deCisiOn bounDary-aware data Augmentation framework called CODA, which utilizes the meta information of the previous epoch to guide the augmentation process and generate more data that are close to the decision boundary. In this way, CODA can obtain a high ratio of attackable data to enhance AT.

2.5 Mixup for robustness

In standard training (ST), mixup has been widely used to improve the generalization of models [33], [34], [46], [47], [48], [49]. Mixup augments the original dataset with interpolated data as follows.

$$\begin{aligned}\overline{x} &= \lambda_{mi} x_i + (1-\lambda_{mi}) x_j,\\ \overline{y} &= \lambda_{mi} y_i + (1-\lambda_{mi}) y_j,\end{aligned} \tag{6}$$

where $(\overline{x}, \overline{y})$ is the interpolated example, (x_i, y_i) and (x_j, y_j) are two randomly selected examples from the original dataset S, $\lambda_{mi} \sim Beta(\alpha_{mi}, \alpha_{mi})$ is the interpolation weight for mixup, $Beta(\alpha_{mi}, \alpha_{mi})$ is the Beta distribution with parameter $\alpha_{mi} \in (0, \infty)$. By training with these interpolated data, Mixup encourages the model to behave linearly in-between training data. Such linear behavior is able to reduce the amount of undesirable oscillations in standard training (ST).

Recent studies [33], [34] also discussed the adversarial robustness of mixup. [34] showed that mixup (in ST) can improve the robustness against fast gradient sign attack (FGSM) [3]. Nevertheless, mixup (in ST) fails to defend against stronger adaptive attacks [33] (e.g., PGD [5] and CW attacks [50]).

To defend against adaptive attacks, several mixup-based adversarial training techniques have been proposed [36], [37]. Interpolated adversarial training (IAT) [36] trained on the interpolations of adversarial data along with the interpolations of natural data. Mixup with targeted labeling adversarial training (M-TLAT) [37] combined vanilla mixup with targeted labeling to enhance AT. In this work, we combine vanilla mixup with standard adversarial training (SAT) [5] and propose SAT-mixup. The detailed procedure of SAT-mixup is shown in Appendix A. First, we use mixup to generate interpolated data. In this way, we can obtain a new dataset based on the interpolated data and the natural data. Then, we compute the adversarial variants based on the new dataset. Last, we train the model on these adversarial variants. Compared to IAT [36], SAT-mixup does not utilize the natural data for training and uses only the adversarial (variants of natural) data for training.

However, these mixup-based methods may hurt the adversarial robustness, which is discussed in the next section.

3 DECISION BOUNDARY-AWARE DATA AUGMENTATION FRAMEWORK (CODA)

In this section, we present our motivation, define attackable data with soft labels, and introduce our proposed deCisiOn bounDary-aware data Augmentation framework (CODA).

3.1 Motivation of CODA

3.1.1 Abundant attackable data can further enhance adversarial robustness

As discussed in Section 2.4, data may contribute unequally during AT, and more data that are close to decision boundary (attackable data) can better benefit the robustness enhancement. However, as shown in Fig. 1, as the training progresses, the ratio of attackable data drops rapidly. This is because the model fine-tunes the decision boundary in each epoch. By optimizing the parameters, the model increases the distances between the decision boundary and the training data (i.e., margin) [43], [51], [52]. As training progresses, training data are farther to the decision boundary. Thus, more and more training data become guarded, and attackable ratio decreases. We illustrate the change of decision boundary with a binary classification toy example in Fig. 2. At the beginning of the training, the decision boundary cannot well-separate the data. After training for several epochs, the decision boundary can better fit the data, and thus the number of attackable data (red and dark blue-colored) reduces rapidly. Besides the toy example, we further conduct an auxiliary experiment to show the change of decision boundary and the number of attackable data on CIFAR-10 dataset [24]. The settings and results are shown in Appendix B. As shown in the appendix, most of the data are attackable at the beginning of training and only few data are attackable after training for several epochs. This observation motivates us to involve more attackable data to enhance adversarial robustness during training.

3.1.2 Adversarial robustness requires the local invariance

A straightforward way to involve more attackable data is utilizing vanilla mixup [33] in AT. However, directly

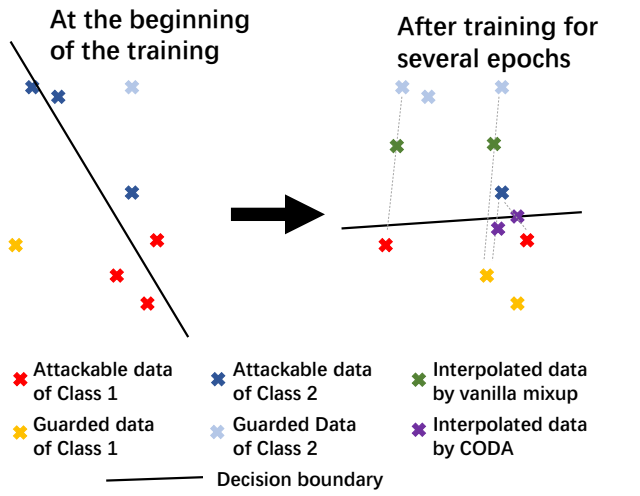


Fig. 2. The illustration of the decision boundary and interpolated data in a binary classification toy example. At the beginning of the training, the decision boundary cannot well separate the data, and thus most of the data are attackable. After training for several epochs, the decision boundary can well separate the data, but the number of attackable data reduces rapidly. Interpolated data by vanilla mixup (green-colored) are likely to be far away from the decision boundary while interpolated data by CODA (purple-colored) are likely to be close to the decision boundary.

employing vanilla mixup in AT will lead to linear behavior, which originates from the randomly sampled λ_{mi}. [3], [35], and [22] demonstrated that in AT, the model predictions are supposed to be locally invariant to the neighborhood of the inputs. [21] proposed to maximize the distances from the inputs to the model's decision boundary, which encouraged the invariant predictions in the inputs' neighborhood. The above studies all validate that a well-trained adversarial model should be locally invariant. This implies that the model's linear behavior is never the desideratum of AT.

We further illustrate the prediction confidence of SAT combined with vanilla mixup (SAT-mixup) and SAT combined with CODA (SAT-CODA) in Fig. 3. For ease of analysis, we first randomly select 2,000 samples from 2 random classes (1,000 samples for each class) from CIFAR-10 dataset [24]. Among them, 1,600 samples (800 for each class) are randomly chosen for training, and the rest 400 samples are used for testing. Second, we embed all the samples into a 2-dimensional (2D) space with PCA [53]. Third, we train on these 2D samples with a small neural network with four fully connected layers (size 10–10–5–2) by using SAT-mixup and SAT-CODA, respectively. We visualize the results in Fig. 3. The green shading represents $f^1(x)$ (the prediction confidence for "airplane"), and the blue shading represents $f^2(x)$ (the prediction confidence for "truck"). The red shading is the margin for the two classes.

The left two panels show the training data while the test data are demonstrated in the right two panels. In the right two panels, (dark purple and dark brown) test points between the two black dotted lines can be attacked by PGD-20, i.e., their adversarial variants are misclassified by the model under PGD-20 attack. The upper two panels show that SAT-mixup introduces linear behavior in-between the two classes, and the adversarial variants of test points close to the decision boundary (dark-colored points between two

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TDSC.2022.3165889, IEEE Transactions on Dependable and Secure Computing

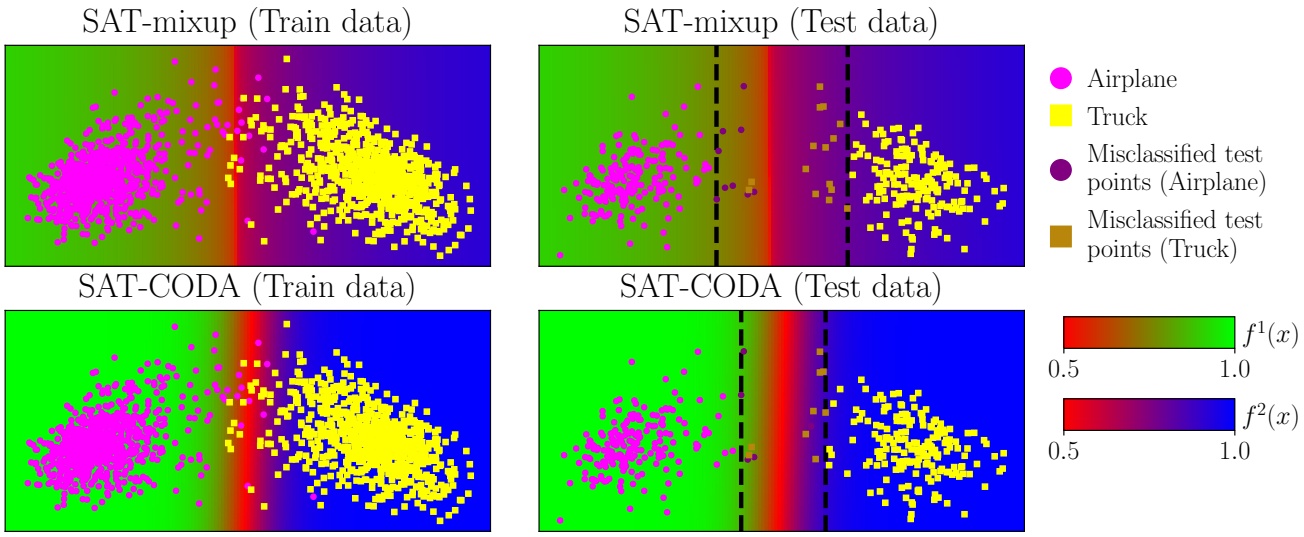


Fig. 3. Comparisons between SAT-mixup and SAT-CODA. The detailed settings are in Section 3.1.2. The green shading represents $f^1(x)$ (the prediction confidence for "airplane"), and the blue shading represents $f^2(x)$ (the prediction confidence for "truck"). The red shading is the margin for the two classes. The left two panels show the training data while the test data are demonstrated in the right two panels. In the right two panels, test points between the two black dotted lines (dark purple and dark brown points) are misclassified under PGD-20 attack. The upper two panels show that SAT-mixup introduces linear behavior in-between the two classes, and the adversarial variants of test points close to the decision boundary (dark-colored points between the two black dotted lines) can be easily misclassified. This shows that the linear behavior near the decision boundary is harmful to adversarial robustness. The lower two panels demonstrate that SAT-CODA can mitigate the linear behavior of SAT-mixup (the transition between green and blue shading is sharper), and there are fewer test points between the two black dotted lines. Therefore, SAT-CODA can increase the robust accuracy of the test points near the decision boundary, and further improve robustness.

black dotted lines) can be easily misclassified. This shows that the linear behavior near the decision boundary is harmful to adversarial robustness.

We summarize the drawbacks of vanilla mixup: (1) the interpolated data are randomly selected, which leads to less attackable data after training for several epochs; and (2) the interpolation weight λ_{mi} is randomly sampled from a Beta distribution, which leads to linear behavior. This motivates us to seek for better solutions that can obtain more attackable data, without sacrificing much the local invariance. We next present our proposed deCisiOn bounDary-aware data Augmentation framework (CODA). On the one hand, CODA can augment more attackable data during training, and the attackable ratio of the interpolated data does not decrease as training progresses. On the other hand, the training of CODA does not lead to linear behavior. As far as we know, CODA is the first solution that can obtain more attackable data while maintaining local invariance.

R2C1

3.2 Definition of attackable data

In mixup, each interpolated example is computed by two different examples (Eq. (6)), which results in a soft interpolated label $\overline{y} \in [0,1]^C$. Thus, we cannot use the definition of attackable data in other papers [22] which only defines data with hard label. To this end, we define the interpolated attackable data as follows.

Definition 1 Attackable data. Let $(\overline{x}, \overline{y})$ be an interpolated example computed by (x_i, y_i) and (x_j, y_j) (with Eq. (6)), $\tilde{x}$ be the adversarial variant of $\overline{x}$ computed with Eq. (3). The interpolated example $(\overline{x}, \overline{y})$ is said to be attackable, if

$$\arg\max_c f_\theta^c(\tilde{x}) \neq \arg\max_c y_i^c \wedge \arg\max_c f_\theta^c(\tilde{x}) \neq \arg\max_c y_j^c, \tag{7}$$

where $\wedge$ is "And" operation.

Since the interpolated example $(\overline{x}, \overline{y})$ is interpolated by two different examples (x_i, y_i) and (x_j, y_j), we say it is attackable, if the predicted label of $\tilde{x}$ (classified by the model $f_\theta(\cdot)$) is different from the label of x_i or x_j. Otherwise, we say $\overline{x}$ a guarded example, if the predicted label of $\tilde{x}$ is the same as the label of x_i or x_j.

3.3 Detail of CODA

As discussed in Section 3.1, we argue to generate more interpolated data that are attackable for training. According to Eq. (6), there are two core factors in data interpolation: 1) selecting data for interpolation; 2) setting the interpolation weight λ. We next explain each factor in more detail.

3.3.1 *Selecting data for interpolation*

In terms of selecting data for interpolation, vanilla mixup randomly picks data for interpolation. The attackable ratio of interpolated data also drops rapidly as adversarial training progresses (see the green dotted line in the left panel of Fig. 1). To alleviate this issue, our CODA only chooses attackable data for interpolation. We illustrate the interpolated data in Fig. 2. As shown in the right figure (after training for several epochs), if we choose random data (red, dark blue, yellow, and light blue-colored) for interpolation, we have a high probability to generate guarded data (green-colored data) that are far away from the decision boundary. our CODA only chooses attackable data (red and dark blue colored) for interpolation, the interpolated data (purple-colored) are more likely to be close to the decision boundary (i.e., attackable) throughout the whole training process.

3.3.2 Setting the interpolation weight λ

With regard to setting the interpolation weight λ, in vanilla mixup, λ_{mi} is sampled randomly from a Beta distribution [33], which is not suitable in adversarial training. Such λ_{mi} will encourage the model to behave linearly between classes, which is not favorable to AT. In this work, instead of random sampling from a Beta distribution, we sophisticatedly adjust λ to a fixed weight in order to mitigate the linear behavior in mixup and thus benefit AT. As shown in Fig. 3, mixup behaves linearly, due to the random λ_{mi}, while our CODA has a sharper transition, due to the fixed λ. In addition, we conduct an auxiliary experiment to further compare the differences between SAT-mixup and SAT-CODA on ResNet-18 [23] with the whole CIFAR-10 dataset in Appendix C. The value of the fixed λ is empirically selected. We further discuss the selection of the fixed λ in Section 4.6.2.

3.3.3 Interpolation function of CODA

In more detail, in this work, we interpolate data as follows:

$$\begin{aligned} \overline{x} &= \lambda x_i + (1-\lambda) x_j \\ \overline{y} &= \lambda y_i + (1-\lambda) y_j, \end{aligned} \tag{8}$$

where $(\overline{x}, \overline{y})$ is the interpolated example, (x_i, y_i) and (x_j, y_j) are two random attackable examples satisfying Eq. (4), and λ is a fixed interpolation weight. In each epoch, we utilize Eq. (8) to augment original training data and train the model with the augmented data and original training data. As a result, we can enhance the adversarial robustness of the model. Compared with vanilla mixup, CODA has two advantages: 1) CODA can obtain more attackable data; 2) CODA can mitigate the linear behavior of vanilla mixup. As a result, in CODA, the attackable ratio of interpolated data does not decrease as training progresses and can maintain local invariance.

It should be noted that CODA is motivated by GAIRAT [22], which is the first method that pays more attention to attackable data. GAIRAT assigns a higher weight to the loss of attackable data and a lower weight to the loss of guarded data. GAIRAT neglects the contribution of the guarded data and may result in catastrophic forgetting of the model [6], i.e., leading to underfitting of the model. To solve this issue, instead of assigning different weights to the loss of training data, our CODA introduces more attackable data by augmentation to enhance AT. Since CODA treats all data equally, our CODA can naturally avoid the catastrophic forgetting issue in GAIRAT.

3.3.4 Training of CODA

Our CODA is a compatible approach, which can be combined with any existing adversarial training methods. For example, we can combine SAT with CODA, and formulate a new method called SAT-CODA. The training process of SAT-CODA is demonstrated in Algorithm 1. First, at the beginning of each epoch, we obtain the interpolation set from the previous attackable set (computed in the previous epoch) (Line 4). Second, we generate the adversarial variants of data from the original dataset along with the interpolation set for updating the model (Line 8 and 15). Third, we filter out the attackable data from the original dataset and put them into the current attackable set A (Line 9-10). This attackable set A will be used to obtain the interpolation set that guides the next-epoch training. Last, we compute the loss of the adversarial variants and update the parameters θ with the gradient of the loss (Line 17). Moreover, we can also combine CODA with TRADES [10], GAIRAT [22], and FastAT [17], i.e., TRADES-CODA, GAIRAT-CODA, and FastAT-CODA, and show the algorithms of these methods in Appendix D.

Algorithm 1 Standard adversarial training combined with decision boundary-aware data augmentation framework (SAT-CODA)

Input: model f_θ, training dataset S, learning rate η, number of epochs T, original batch size m, interpolation batch size m', number of batches M, perturbation bound ϵ, step size α, PGD step number K
Output: adversarially robust network f_θ

1: $A_0 \leftarrow S$
2: **for** epoch $t = 1, 2, ..., T$ **do**
3: $\quad A_t = \emptyset$
4: $\quad$ Compute interpolation set $\overline{S}_t$ by A_{t-1} and Eq. (8)
5: $\quad$ **for** mini-batch $= 1, ..., M$ **do**
6: $\quad\quad$ Randomly sample $\{(x_i, y_i)\}_{i=1}^{m}$ from S
7: $\quad\quad$ **for** $i = 1, \ldots, m$ **do in parallel**
8: $\quad\quad\quad$ Generate adversarial data $\widetilde{x}_i$ of x_i by Eq. (3)
9: $\quad\quad\quad$ **if** $\arg\max_c f_\theta^c(\widetilde{x}_i) \neq \arg\max_c y_i^c$ **then**
10: $\quad\quad\quad\quad A_t \leftarrow A_t \cup \{(x_i, y_i)\}$
11: $\quad\quad\quad$ **end if**
12: $\quad\quad$ **end for**
13: $\quad\quad$ Randomly sample $\{(\overline{x}_i, \overline{y}_i)\}_{i=m+1}^{m+m'}$ from $\overline{S}_t$
14: $\quad\quad$ **for** $i = m+1, ..., m+m'$ **do in parallel**
15: $\quad\quad\quad$ Generate adversarial data $\widetilde{x}_i$ of $\overline{x}_i$ by Eq. (3)
16: $\quad\quad$ **end for**
17: $\quad\quad$ $\theta \leftarrow \theta - \eta \cdot \frac{1}{m+m'} \sum_{i=1}^{m+m'} \nabla_\theta \ell(f_\theta(\widetilde{x}_i), y_i)$
18: $\quad$ **end for**
19: **end for**

4 Experiments

4.1 Datasets, model and hardware

In this section, we empirically justify the efficacy of our proposed CODA on three representative image datasets: CIFAR-10 [24], CIFAR-100 [24], MNIST [54], and SVHN [55]. The CIFAR-10 dataset consists of 60,000 32x32 color images in 10 classes. The CIFAR-100 dataset is similar to CIFAR-10 dataset, except it has 100 classes. The MNIST dataset consists of handwritten digits, with 60,000 training examples and 10,000 test examples. SVHN is a real-world image dataset, which consists of 73,257 training data and 26,032 testing data in 10 classes.

All models are trained with ResNet-18 [23] or Wide ResNet (WRN) [56]. All experiments are run on the same machine with Intel Xeon Gold 5218 CPU, 250GB RAM, and six NVIDIA Tesla V100 SXM2 GPU. All methods are implemented with PyTorch 1.7.0 [57].

4.2 Baselines

Our experiments are conducted on the following baselines: standard adversarial training (SAT) [5], interpolated adversarial training (IAT) [36], SAT-noise [42], SAT-random [44],

KD+SWA [58], geometry-aware instance-reweighted adversarial training (GAIRAT) [22], fast adversarial training (FastAT) [17], and TRADES [10]. SAT is a standard adversarial training method that used projected gradient descent for training. IAT, SAT-noise, and SAT-random are 3 recent data augmentation methods. IAT utilized mixup to interpolate data and trained on the interpolations of adversarial data along with the interpolations of natural data. SAT-noise augmented the training data by injecting random Gaussian noises. SAT-random randomly resized the input data and randomly padded the resized data. KD+SWA enhanced AT by smoothing the logits via self-training and smoothing the weights via stochastic weight averaging. GAIRAT treated data unequally and paid more attention to attackable data. Specifically, GAIRAT assigned a higher weight to the losses of attackable data during training. FastAT combined FGSM [3] training with random initialization. TRADES proposed a new loss that consists of two terms: an empirical risk minimization to maximize the natural accuracy, and a regularization term to push the decision boundary away from the data.

To test the effectiveness and compatibility of our proposed CODA, we modify SAT, GAIRAT, FastAT, and TRADES to their CODA versions, i.e., SAT-CODA, GAIRAT-CODA, FastAT-CODA, and TRADES-CODA. The algorithm of SAT-CODA is demonstrated in Algorithm 1, and the algorithms of GAIRAT-CODA, FastAT-CODA, and TRADES-CODA are shown in Appendix D. To further compare CODA with mixup, we also modify SAT, GAIRAT, FastAT, and TRADES to their mixup versions, i.e., SAT-mixup, GAIRAT-mixup, FastAT-mixup, and TRADES-mixup.

4.3 Experimental settings

We first introduce the settings for CIFAR-10 and CIFAR-100 dataset. In our experiments, we consider $||\tilde{x} - x||_\infty < \epsilon$ with the same ϵ in both training and evaluations. All images of CIFAR-10 and CIFAR-100 are normalized into $[0, 1]$. To generate the most adversarial data to update the model, we set the perturbation bound $\epsilon = 8/255$; PGD step number $K = 10$; and PGD step size $\alpha = 2/255$, which keeps the same as [18]. We train the model using SGD with momentum$= 0.9$ for 60 epochs with weight decay $5e-4$. It has been proved that piecewise constant learning rate scheduling achieves the best performance [5]. So we follow the setting of [5] and set the initial learning rate $\eta = 0.1$, which is then divided by 10 at Epoch 30 and 45, respectively. For evaluations, we report standard test accuracy for natural test data and robust test accuracy for adversarial test data. The adversarial test data are generated by PGD-20 (PGD with 20 steps) [5], CW-30 (CW with 30 steps) [50], and AA (auto attack) [59] with the same perturbation bound $\epsilon = 8/255$. The step size α for PGD-20 attack and CW-30 attack is $2/255$.

For SVHN dataset, we set the perturbation bound $\epsilon = 4/255$, PGD step size $\alpha = 1/255$, and initial learning rate $\eta = 0.01$. For evaluations, the adversarial test data are generated by PGD-20 and CW-30 with the same perturbation bound $\epsilon = 4/255$. The step size α for PGD-20 and CW is $1/255$. Other settings in SVHN dataset are the same as CIFAR-10 and CIFAR-100.

For MNIST dataset, we utilize a CNN with 4 convolutional layers, followed by 3 fully-connected layers. We set the perturbation bound $\epsilon = 0.3$, PGD step number $K = 40$, PGD step size $\alpha = 0.1$, weight decay 0, and initial learning rate $\eta = 0.01$. During test phase, the adversarial test data are generated by PGD-40 and CW-40 attack with the same perturbation bound $\epsilon = 0.3$ and step size $\alpha = 0.1$. Other settings in MNIST dataset are the same as CIFAR-10 and CIFAR-100.

TABLE 1
Test accuracies of SAT-CODA under different interpolation weight λ on the validation set on CIFAR-10 dataset. The best robustness is in bold.

Defense	Natural	PGD-20	CW-30
SAT-CODA($\lambda = 0.1$)	84.21	48.13	46.92
SAT-CODA($\lambda = 0.2$)	83.98	49.92	48.64
SAT-CODA($\lambda = 0.3$)	83.82	52.06	48.91
SAT-CODA($\lambda = 0.4$)	83.27	52.75	49.19
SAT-CODA($\lambda = 0.5$)	81.62	**53.74**	**50.12**

For CODA-based and mixup-based methods, 50% of the training data are sampled from the interpolation set $\overline{S}$, and we set the original batch size $m = 64$ and the interpolation batch size $m' = 64$. To make sure that all methods train on the same amount of data, instead of sampling interpolated data in CODA-based (or mixup-based) methods, all baseline methods train all the data twice in each epoch. We set the original batch size $m = 128$ for all baseline methods. We set the interpolation weight $\lambda = 0.5$ for all CODA-based methods. In addition, all CODA-based methods (e.g., SAT-CODA, TRADES-CODA, GAIRAT-CODA, and FastAT-CODA) involve a burn-in period. During the initial period of the training epochs (first 30 epochs), instead of generating data with CODA, we utilize the original data for the training. It should be noted we did not involve the burn-in period for the models trained in Fig. 1 and Fig. 3, in order to better illustrate the differences between SAT-CODA, SAT-mixup, and SAT.

Following the original setting of mixup [33], the interpolation weight of mixup-based methods $\lambda_{mi} \sim Beta(1, 1)$, i.e., λ_{mi} is uniformly distributed between 0 and 1. For fair comparisons, our CODA-based and mixup-based methods keep the same settings as their original versions. In TRADES, TRADES-mixup, and TRADES-CODA, we set KL loss weight $\beta = 6$, weight of Gaussian noise $\gamma = 0.001$, and training epoch $T = 75$. The initial learning rate $\eta = 0.1$ and divided by 10 at 37-th and 56-th epochs, respectively.

4.4 Selection of the fixed interpolation weight λ

One of the most important hyperparameter is the fixed interpolation weight λ. For CIFAR-10 dataset, we randomly sample 20% of the training data as the validation set and use the rest of training data for training. To explore the best λ for AT, we we test the performance of SAT-CODA using the validation set under different interpolation weight $\lambda \in \{0.1, 0.2, 0.3, 0.4, 0.5\}$ on ResNet-18. Note that fixing $\lambda \in \{0.1, 0.2, 0.3, 0.4\}$ is equivalent to fixing $\lambda \in \{0.6, 0.7, 0.8, 0.9\}$, due to that λ and $1-\lambda$ are symmetric in CODA (Eq. (8)). Thus we do not report the results of $\lambda > 0.5$. TABLE 1 shows that the robustness increases as λ becomes larger, and fixing $\lambda = 0.5$ achieves the best robustness. All these results imply that $\lambda = 0.5$ is an optimal

TABLE 2
Test accuracies of the best epochs of different methods on the CIFAR-10 dataset. Diff. are the differences between the CODA-based (mixup-based) methods and their original versions. The best robustness of each network is in bold.

Network	Defense	Natural		PGD-20		CW-30		AA	
		Acc.	Diff.	Acc.	Diff.	Acc.	Diff.	Acc.	Diff.
ResNet-18	IAT [36]	90.61	-	41.11	-	41.93	-	36.43	-
	SAT-noise [42]	70.24	-	41.38	-	39.92	-	37.91	-
	SAT-random [44]	75.91	-	44.07	-	41.85	-	39.45	-
	KD+SWA [58]	85.16	-	46.85	-	46.73	-	44.74	-
	SAT [5]	81.72	-	51.05	-	49.86	-	47.25	-
	SAT-mixup	82.33	+0.61	52.23	+1.18	49.75	-0.11	47.52	+0.27
	SAT-CODA	81.59	-0.13	53.57	+2.52	**50.07**	+0.21	47.91	+0.66
	GAIRAT [22]	82.91	-	55.93	-	42.74	-	40.03	-
	GAIRAT-mixup	81.06	-1.85	57.50	+1.57	40.20	-2.54	38.69	-1.34
	GAIRAT-CODA	81.29	-1.62	**60.57**	+4.64	43.06	+0.32	40.36	+0.33
	FastAT [17]	86.33	-	43.81	-	45.12	-	41.03	-
	FastAT-mixup	79.31	-7.02	40.47	-3.34	39.42	-5.70	36.91	-4.12
	FastAT-CODA	85.45	-0.88	47.49	+3.68	45.88	+0.76	42.88	+1.85
	TRADES [10]	81.28	-	52.05	-	49.63	-	48.53	-
	TRADES-mixup	82.12	+0.84	52.24	+0.19	49.11	-0.52	48.26	-0.27
	TRADES-CODA	81.54	+0.26	53.04	+1.01	49.84	+0.21	**48.96**	+0.43
WRN-32-10	IAT	95.79	-	50.85	-	49.53	-	41.51	-
	SAT-noise	81.57	-	41.12	-	41.61	-	39.17	-
	SAT-random	82.61	-	47.80	-	46.56	-	44.92	-
	KD+SWA	84.46	-	52.73	-	52.53	-	50.85	-
	SAT	86.73	-	53.66	-	53.66	-	50.96	-
	SAT-mixup	86.63	-0.10	54.58	+0.92	53.45	-0.21	51.22	+0.26
	SAT-CODA	86.49	-0.24	55.92	+2.26	**53.99**	+0.33	**51.92**	+0.96
	GAIRAT	86.31	-	56.69	-	44.63	-	42.26	-
	GAIRAT-mixup	85.29	-1.02	60.11	+3.42	43.68	-0.95	41.37	-0.89
	GAIRAT-CODA	84.59	-1.72	**64.97**	+8.28	46.71	+2.08	44.84	+2.58
	FastAT	88.38	-	46.14	-	47.31	-	43.70	-
	FastAT-mixup	87.31	-1.07	44.27	-1.87	46.18	-1.13	40.68	-3.02
	FastAT-CODA	88.83	+0.45	48.42	+2.28	48.26	+0.95	41.91	-1.79
WRN-34-10	TRADES	84.92	-	55.33	-	53.81	-	52.54	-
	TRADES-mixup	83.27	-1.65	56.19	+0.86	53.99	+0.18	52.23	-0.31
	TRADES-CODA	83.64	-1.28	**57.45**	+2.12	**54.14**	+0.33	**53.37**	+0.83

choice for improving the robustness of AT. Besides CIFAR-10 dataset, we also report the results on CIFAR-100, MNIST, and SVHN datasets in Appendix E. As shown in the appendix, $\lambda = 0.5$ achieve the best robustness. Thus, we set $\lambda = 0.5$ for all datasets.

4.5 Robustness evaluation

In this section, we first demonstrate the effectiveness of involving more attackable data. Second, we evaluate the robustness of our proposed CODA-based methods on CIFAR-10 dataset. Last, we demonstrate the effectiveness of CODA on CIFAR-100, MNIST, and SVHN dataset.

4.5.1 Effectiveness of involving more attackable data

In Fig. 1, we train three models with SAT, SAT-mixup, and SAT-CODA, respectively, with ResNet-18 [23] on CIFAR-10 dataset [24]. These models do not involve the burn-in period. In the left panel, we plot the ratio of attackable data in different epochs. The green dotted line and red dotted line represent the ratio of the attackable data of the interpolated data of mixup and CODA, respectively. The blue solid line, green solid line, and red solid line represent the ratio of the attackable data of the final training data (including original data and interpolated data) of SAT, SAT-mixup, and SAT-CODA, respectively. In the right panel, we report the robust test accuracies of these models against PGD-20 [5] attack.

From the left panel of Fig. 1, we can see that the attackable data ratio of SAT (blue solid line) decreases rapidly as training progresses. Mixup (green dotted line) can increase the ratio of attackable data but still cannot keep a high ratio of attackable data after 30-th epoch. CODA (red dotted line) can always generate a high ratio of attackable data. The right panel demonstrates that all three methods have similar robust test accuracies before 30-th epoch, due to that all three methods have a high ratio (more than 60%) of attackable data. After 30-th epoch (with a reduced learning rate), the robust test accuracy of SAT drops significantly, which is strongly related to the low ratio of attackable data. After 45-th epoch (with a further reduction of learning rate), SAT-mixup has a lower performance than SAT-CODA, due to the drop of the ratio of attackable data. Compared to SAT (with robust test accuracy 51.05%), SAT-mixup (with robust test accuracy

TABLE 3
Test accuracies of the last epochs of different methods on CIFAR-10 dataset. Diff. are the differences between the CODA-based (mixup-based) methods and their original versions. The best robustness of each network is in bold.

Network	Defense	Natural		PGD-20		CW-30		AA	
		Acc.	Diff.	Acc.	Diff.	Acc.	Diff.	Acc.	Diff.
ResNet-18	IAT	90.53	-	39.50	-	40.43	-	35.10	-
	SAT-noise	77.61	-	33.88	-	34.29	-	32.95	-
	SAT-random	80.76	-	36.51	-	36.75	-	34.16	-
	KD+SWA	85.63	-	45.45	-	46.12	-	43.70	-
	SAT	84.77	-	45.47	-	45.90	-	43.28	-
	SAT-mixup	85.33	+0.56	50.19	+4.72	47.46	+1.56	45.19	+1.91
	SAT-CODA	83.15	-1.62	52.06	+6.59	48.21	+2.31	45.59	+2.31
	GAIRAT	83.03	-	50.94	-	38.04	-	35.17	-
	GAIRAT-mixup	80.72	-2.31	57.24	+6.30	38.76	+0.72	35.29	+0.12
	GAIRAT-CODA	81.07	-1.96	**60.08**	+9.14	40.09	+2.05	37.43	+2.26
	FastAT	89.26	-	1.44	-	2.72	-	0.00	-
	FastAT-mixup	91.83	+2.57	5.67	+4.23	9.70	+6.98	0.05	+0.05
	FastAT-CODA	89.13	-0.13	1.23	-0.21	2.35	-0.37	0.00	+0.00
	TRADES	81.17	-	51.81	-	49.35	-	48.41	-
	TRADES-mixup	82.04	+0.87	51.13	-0.68	48.80	-0.55	47.79	-0.62
	TRADES-CODA	82.11	+0.94	52.05	+0.24	**49.51**	+0.16	**48.45**	+0.04
WRN-32-10	IAT	96.55	-	47.37	-	48.50	-	38.70	-
	SAT-noise	80.90	-	39.15	-	39.40	-	36.61	-
	SAT-random	84.19	-	39.08	-	39.48	-	36.79	-
	KD+SWA	86.53	-	50.34	-	**50.29**	-	**48.07**	-
	SAT	86.88	-	47.94	-	48.12	-	45.82	-
	SAT-mixup	86.99	+0.11	50.17	+2.23	48.42	+0.30	46.42	+0.60
	SAT-CODA	86.14	-0.74	51.42	+3.48	49.17	+1.05	46.70	+0.88
	GAIRAT	85.43	-	52.50	-	43.71	-	41.14	-
	GAIRAT-mixup	85.19	-0.24	53.92	+1.42	43.97	+0.26	41.55	+0.41
	GAIRAT-CODA	82.18	-3.25	**55.11**	+2.61	45.65	+1.94	43.24	+2.10
	FastAT	89.56	-	6.79	-	6.01	-	0.03	-
	FastAT-mixup	91.08	+1.52	7.51	+0.72	5.41	-0.60	0.02	-0.01
	FastAT-CODA	89.93	+0.37	0.39	-6.40	0.34	-5.67	0.00	-0.03
WRN-34-10	TRADES	85.10	-	53.16	-	51.12	-	50.09	-
	TRADES-mixup	84.01	-1.09	53.52	+0.36	50.80	-0.32	49.77	-0.32
	TRADES-CODA	84.25	-0.85	**56.91**	+3.75	**53.91**	+2.79	**53.08**	+2.99

52.01%) can improve the best robust test accuracy by 0.96%. SAT-CODA (with robust test accuracy 53.60%) can further increase the best test accuracy of SAT by 2.55%.

4.5.2 Robustness evaluation on CIFAR-10

To show the efficiency of the CODA on CIFAR-10 dataset, we conduct experiments on the baseline methods (SAT, IAT, KD+SWA, GAIRAT, FastAT, and TRADES), the mixup-based methods (SAT-mixup, GAIRAT-mixup, FastAT-mixup, and TRADES-mixup), and our proposed CODA-based methods (SAT-CODA, GAIRAT-CODA, FastAT-CODA, and TRADES-CODA). In TABLE 2 and TABLE 3, we report the best accuracy across all epochs and the accuracy of the last epoch, respectively, for each model. Following the setting of [5], all models (except TRADES, TRADES-mixup, and TRADES-CODA) are trained with ResNet-18 or WRN-32-10. To keep the same setting as [10], TRADES, TRADES-mixup and TRADES-CODA are trained with ResNet-18 and WRN-34-10.

GAIRAT-CODA achieves the highest robust test accuracy under PGD-20 attack (64.97%), and TRADES-CODA achieves the best robust test accuracy under CW-30 attack (54.14%) and AA (53.37%). Our proposed CODA-based AT methods generally outperform their original versions and other baseline methods. For example, compared with GAIRAT, GAIRAT-CODA increases the robust test accuracy of the best epoch by 8.28% on WRN-32-10 under PGD-20 attack. These results demonstrate that our proposed CODA is capable of improving adversarial robustness. Moreover, CODA is a compatible approach, which can be combined with any existing adversarial training methods and improve their robust test accuracies.

For SAT, the robust test accuracy of the best epoch under PGD-20 attack on ResNet-18 is 51.05%, but drops to 45.47% at the last epoch. This indicates that SAT suffers from overfitting. By contrast, SAT-CODA increases the robust test accuracy of best epoch of SAT by 2.52% and the robust test accuracy of last epoch by 6.59%. Therefore, by incorporating CODA with SAT, we can not only improve the robust test accuracy of the best epoch but also alleviate the overfitting issue.

Although mixup can increase the ratio of attackable data, it can be harmful to AT. For example, FastAT-mixup

TABLE 4
Test accuracies of SAT and SAT-CODA on CIFAR-100 dataset. The best robustness is in bold.

Defense	Natural	PGD-20	CW-30
SAT	57.34	27.78	26.94
SAT-CODA	56.79	**29.90**	**27.47**

TABLE 5
Test accuracies of SAT and SAT-CODA on MNIST dataset. The best robustness is in bold.

Defense	Natural	PGD-40	CW-40
SAT	99.43	96.54	96.51
SAT-CODA	99.11	**97.69**	**97.53**

TABLE 6
Test accuracies of SAT and SAT-CODA on SVHN dataset. The best robustness is in bold.

Defense	Natural	PGD-20	CW-30
SAT	95.67	74.52	74.86
SAT-CODA	94.98	**77.98**	**75.10**

TABLE 7
Test accuracies of SAT-CODA with different adversarial data generating methods on CIFAR-10 dataset. The best robustness is in bold.

Defense	Natural	PGD-20	CW-30
SAT-CODA (PGD)	81.59	53.57	**50.07**
SAT-CODA (CW)	81.19	**59.65**	47.79

underperforms FastAT by 5.70% under CW attack. We hypothesise that the reason is the linear behavior is harmful to AT, and thus decreases the performance. The results show that directly utilizing mixup in AT is not suitable. Our CODA outperforms vallina mixup across all AT methods, due to that CODA can increase the ratio of attackable data and mitigate the linear behavior of mixup.

Among all the results, GAIRAT-CODA achieves the best performance under PGD-20 attack, which implies that paying more attention to attackable data can indeed improve robustness. However, GAIRAT, GAIRAT-mixup, and GAIRAT-CODA have (almost) the worst defense under CW-30 attack and AA, due to that GAIRAT neglects the effect of guarded data during training and result in catastrophic forgetting of the model [6]. CODA involves more attackable data for training but does not pay less attention to guarded data. Thus, CODA-based methods are able to improve the robustness under PGD-20 attack, CW-30 attack, and AA at the same time.

The CODA versions of the adversarial training methods have a slightly lower natural test accuracies than their original versions, due to the trade-off between natural accuracy and robust accuracy [60], [61]. The failure of FastAT, Fast-mixup, and FastAT-CODA in the last epoch is due to catastrophic overfitting [17].

4.5.3 Robustness evaluation on CIFAR-100, MNIST, and SVHN

We report the performance of SAT-CODA compared with SAT on CIFAR-100, MNIST, and SVHN dataset in TABLE 4, TABLE 5, and TABLE 6, respectively. SAT-CODA has better robust test accuracies on all datasets. For example, SAT-CODA improves the test accuracy of SAT by 3.46% on SVHN dataset under PGD-20 attack. This further manifests that our CODA can indeed improve adversarial robustness.

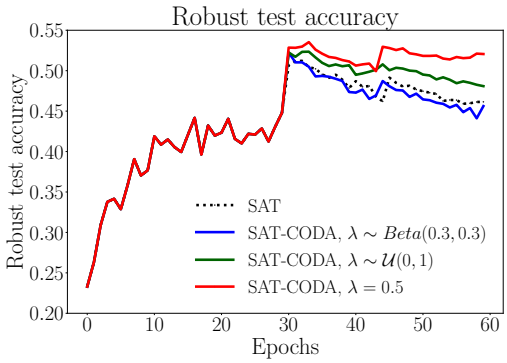


Fig. 4. Robust test accuracies of SAT-CODA with different λ under PGD-20 attack.

4.6 Ablation studies

In this section, we conduct a series of ablation studies to further understand the effects of different components in SAT-CODA, including: (1) the adversarial training data generated with CW attack; (2) the impact of interpolation weight λ; (3) the effect of the burn-in period; (4) the varying ratios of the original data and the interpolated data; (5) the impact of attackable ratio; (6) training time of SAT-CODA. Note that all ablation studies are conducted on CIFAR-10 dataset.

4.6.1 Generating adversarial training data with different methods

Besides PGD-10 attack, we also utilize the CW-10 attack [50] (CW attack with 10 steps) to generate adversarial training data (Eq. (3)) on CIFAR-10 dataset. TABLE 7 reports the test accuracies of generating adversarial training data with PGD-10 (SAT-CODA (PGD)) and CW-10 (SAT-CODA (CW)). The model is trained with ResNet-18 on CIFAR-10 dataset. SAT-CODA (CW) increases the robust test accuracy by 6.08% under PGD-20 attack but has a lower performance under CW-30 attack. SAT-CODA (CW) achieves a higher test accuracy than all baseline methods (e.g, SAT, IAT, KD+SWA, GAIRAT, FastAT, and TRADES, shown in TABLE 2) under PGD-20 attack, which shows that generating adversarial training data with CW attack is also promising.

4.6.2 The impact of the interpolation weight λ

In addition to the fixed $\lambda = 0.5$, we also randomly sample λ from a uniform distribution or a beta distribution, i.e., $\lambda \sim \mathcal{U}(0, 1)$ or $\lambda \sim Beta(0.3, 0.3)$. The hyperparameters for uniform distribution and beta distribution are set according to [33]. As shown in Fig. 4, when $\lambda = 0.5$ (red line), the model achieves the best performance. The model with $\lambda \sim \mathcal{U}(0, 1)$ (green line) and $\lambda \sim Beta(0.3, 0.3)$ (blue line) underperform our SAT-CODA with $\lambda = 0.5$. We hypothesis this is due to that the randomly sampled λ (from uniform/Beta distributions) leads to an undesirable linear behavior. Note that all methods have the same performance before 30-th epoch, as we utilize the burn-in period.

TABLE 8
Test accuracies of SAT-CODA with and without the burn-in period on CIFAR-10 dataset. The best robustness is in bold.

Defense	Natural	PGD-20	CW-30
SAT-CODA (w/o burn-in)	82.35	**53.60**	49.13
SAT-CODA (burn-in)	81.59	53.57	**50.07**

TABLE 9
Test accuracies of SAT-CODA with different ratios of original data and interpolated data on ResNet-18. The best robustness is in bold.

Defense	Natural	PGD-20	CW-30
SAT-CODA (1:1)	81.59	**53.57**	**50.07**
SAT-CODA (1:2)	81.90	53.05	49.33
SAT-CODA (2:1)	82.88	51.98	49.47

TABLE 10
Test accuracies of SAT-CODA with different attackable ratio (attrat) on ResNet-18. The best robustness is in bold.

Attackable ratio	Natural	PGD-20	CW
attrat=0.3	81.08	49.25	48.12
attrat=0.4	81.41	49.79	48.33
attrat=0.5	83.17	51.71	48.69
attrat=0.6	83.66	52.41	48.72
attrat=0.7	83.63	52.52	48.93
attrat=0.8	83.84	**53.88**	**49.27**

TABLE 11
Training time of SAT and SAT-CODA on different networks. Diff. is the extra percentage of time that SAT-CODA needs compared to SAT.

Defense	ResNet-18	Diff.	WRN-32-10	Diff.
SAT	562s	-	3772s	-
SAT-CODA	591s	+5.16%	3836s	+1.70%

4.6.3 The effect of burn-in period

To show the effect of burn-in period, we conduct an experiment on ResNet-18. TABLE 8 compares the performance of SAT-CODA with and without the burn-in period. SAT-CODA (burn-in) first trains on the original data without the interpolated data in the first 30 epochs, then starts to train on both the original data and the interpolated data after 30-th epoch. In contrast, SAT-CODA (w/o burn-in) utilizes the original data and the interpolated data for adversarial training from the first epoch. All the results in TABLE 8 suggest that the burn-in period can slightly improve the robustness under CW-30 attack, but has a slightly lower test accuracy under PGD-20 attack. Another advantage of utilizing the burn-in period is that we can reuse the original model and increase training efficiency.

4.6.4 Different ratios of original data and interpolated data

We train SAT-CODA on the original dataset and the interpolation set with different ratios (e.g., 1:1, 1:2, and 2:1). To make sure these methods are trained on the same amount of data in one mini-batch, in SAT-CODA (1:1), we set original batch size $m = 64$ and interpolation batch size $m' = 64$; in SAT-CODA (1:2), we set $m = 43$ and $m' = 85$; and in SAT-CODA (2:1), we set $m = 85$ and $m' = 43$. We report test accuracies of these models in TABLE 9. As can be observed, SAT-CODA (1:1) has a better performance than SAT-CODA (1:2) and SAT-CODA (2:1). These results imply that the original data is also very useful in AT and we cannot neglect them. This also indicates that selecting an appropriate ratio of the original data and the interpolated data is important for robustness enhancement.

4.6.5 The impact of the attackable ratio

As we claimed in the previous sections, adversarial robustness is highly correlated with the attackable ratio. To verify this, we conducted experiments by varying the attackable ratio. In particular, we modify the interpolated data and fix the attackable ratio of the training data to be the same in each epoch. We report the performance of SAT-CODA across attrat=$\{0.3, 0.4, 0.5, 0.6, 0.7, 0.8\}$ on CIFAR-10 dataset in TABLE 10, where attrat is the fixed attackable ratio in each epoch. Not surprisingly, the robustness of SAT-CODA increases as attrat increases, and SAT-CODA achieves the best performance when attrat=0.8, which verifies that a higher attackable ratio can indeed improve robustness. Mover

4.6.6 Training time of CODA

We also report the training time of SAT and SAT-CODA in TABLE 11. SAT-CODA only needs 5.16% and 1.70% more training time compared to SAT, which shows that the training efficiency of CODA is slightly lower than SAT. This is because compared to SAT, our SAT-CODA only needs to conduct one more operation that augments the interpolated data. In fact, finding the most adversarial data (which needs to be done by every AT method) consumes most computation resources, and thus overwhelm the training time of the data augmentation operation. This shows that CODA is able to be applied in most adversarial training scenarios and increases the robustness of the model at a low cost.

4.7 Discussion

In Section 4.5.2, we show that GAIRAT and GAIRAT-CODA achieve the best robustness under PGD-20 attack but (almost) the worst robustness under CW-30 attack and AA. This is due to the different impacts of attackable data and guarded data in adversarial training. Although we argue attackable data is more helpful to improve AT, the guarded data still plays an important role in AT. The experimental results in Section 4.6.4 also show that using only the attackable data for training might not be suitable and we also need guarded data in AT. It would be interesting to further investigate the contributions of attackable data and guarded data in AT.

5 Conclusion and Future Work

This paper proposed a novel data augmentation framework, i.e., deCisiOn bounDary-aware data Augmentation framework (CODA), for adversarial training. CODA utilizes the meta information of the previous epoch to guide the data interpolation process. CODA is the first solution that can obtain a high ratio of attackable data by interpolation without significantly sacrificing the local invariance, thus enhancing adversarial robustness. Extensive experiments on three real-world datasets show that our CODA boosts the robustness of adversarial training.

We believe our work can be inspiring for the research in adversarial training. A number of avenues for further work are appealing. In particular, we would like to investigate the different impacts of attackable data and guarded data, and how these data affect the decision boundary in AT. We would also like to develop more advanced techniques for data augmentation in AT.

ACKNOWLEDGMENTS

This work is supported by the Key Research and Development Program of Zhejiang Province of China (No. 2020C01024).

REFERENCES

[1] B. Biggio, I. Corona, D. Maiorca, B. Nelson, N. Šrndić, P. Laskov, G. Giacinto, and F. Roli, "Evasion attacks against machine learning at test time," in *Joint European conference on machine learning and knowledge discovery in databases*. Springer, 2013, pp. 387–402.

[2] C. Szegedy, W. Zaremba, I. Sutskever, J. Bruna, D. Erhan, I. Goodfellow, and R. Fergus, "Intriguing properties of neural networks," in *ICLR*, 2014.

[3] I. J. Goodfellow, J. Shlens, and C. Szegedy, "Explaining and harnessing adversarial examples," in *ICLR*, 2015.

[4] A. Nguyen, J. Yosinski, and J. Clune, "Deep neural networks are easily fooled: High confidence predictions for unrecognizable images," in *CVPR*, 2015, pp. 427–436.

[5] A. Madry, A. Makelov, L. Schmidt, D. Tsipras, and A. Vladu, "Towards deep learning models resistant to adversarial attacks," in *ICLR*, 2018.

[6] Q. Cai, C. Liu, and D. Song, "Curriculum adversarial training," in *IJCAI*, 2018.

[7] Y. Wang, X. Ma, J. Bailey, J. Yi, B. Zhou, and Q. Gu, "On the convergence and robustness of adversarial training," in *ICML*, 2019.

[8] Y. Wang, D. Zou, J. Yi, J. Bailey, X. Ma, and Q. Gu, "Improving adversarial robustness requires revisiting misclassified examples," in *ICLR*, 2020.

[9] R. Wang, K. Xu, S. Liu, P.-Y. Chen, T.-W. Weng, C. Gan, and M. Wang, "On fast adversarial robustness adaptation in model-agnostic meta-learning," in *ICLR*, 2021.

[10] H. Zhang, Y. Yu, J. Jiao, E. Xing, L. El Ghaoui, and M. I. Jordan, "Theoretically principled trade-off between robustness and accuracy," in *ICML*, 2019.

[11] J. Zhang, X. Xu, B. Han, G. Niu, L. Cui, M. Sugiyama, and M. Kankanhalli, "Attacks which do not kill training make adversarial learning stronger," in *ICML*, 2020.

[12] T. Pang, K. Xu, C. Du, N. Chen, and J. Zhu, "Improving adversarial robustness via promoting ensemble diversity," in *ICML*, 2019.

[13] T. Pang, K. Xu, Y. Dong, C. Du, N. Chen, and J. Zhu, "Rethinking softmax cross-entropy loss for adversarial robustness," in *ICLR*, 2020.

[14] T. Pang, X. Yang, Y. Dong, H. Su, and J. Zhu, "Bag of tricks for adversarial training," in *ICLR*, 2021.

[15] D. Wu, S.-T. Xia, and Y. Wang, "Adversarial weight perturbation helps robust generalization," *NeurIPS*, vol. 33, 2020.

[16] V. B.S. and R. V. Babu, "Single-step adversarial training with dropout scheduling," in *CVPR*, 2020.

[17] E. Wong, L. Rice, and J. Z. Kolter, "Fast is better than free: Revisiting adversarial training," in *ICLR*, 2020.

[18] L. Rice, E. Wong, and J. Z. Kolter, "Overfitting in adversarially robust deep learning," in *ICML*, 2020.

[19] G. Sriramanan, S. Addepalli, A. Baburaj, and R. V. Babu, "Gama: Guided adversarial margin attack," in *NeurIPS*, 2020.

[20] Y. Bai, Y. Zeng, Y. Jiang, S.-T. Xia, X. Ma, and Y. Wang, "Improving adversarial robustness via channel-wise activation suppressing," in *ICLR*, 2021.

[21] G. W. Ding, Y. Sharma, K. Y. C. Lui, and R. Huang, "Mma training: Direct input space margin maximization through adversarial training," in *ICLR*, 2020.

[22] J. Zhang, J. Zhu, G. Niu, B. Han, M. Sugiyama, and M. Kankanhalli, "Geometry-aware instance-reweighted adversarial training," in *ICLR*, 2021.

[23] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *CVPR*, 2016.

[24] A. Krizhevsky, G. Hinton *et al.*, "Learning multiple layers of features from tiny images," in *Technical report*, 2009.

[25] L. Schmidt, S. Santurkar, D. Tsipras, K. Talwar, and A. Madry, "Adversarially robust generalization requires more data," in *NeurIPS*, 2018, pp. 5014–5026.

[26] D. Hendrycks, K. Lee, and M. Mazeika, "Using pre-training can improve model robustness and uncertainty," in *ICML*, 2019.

[27] Y. Carmon, A. Raghunathan, L. Schmidt, P. Liang, and J. C. Duchi, "Unlabeled data improves adversarial robustness," in *NeurIPS*, 2019.

[28] J. Alayrac, J. Uesato, P. Huang, A. Fawzi, R. Stanforth, and P. Kohli, "Are labels required for improving adversarial robustness?" in *NeurIPS*, 2019.

[29] A. Najafi, S. Maeda, M. Koyama, and T. Miyato, "Robustness to adversarial perturbations in learning from incomplete data," in *NeurIPS*, 2019.

[30] V. H. Buch, I. Ahmed, and M. Maruthappu, "Artificial intelligence in medicine: current trends and future possibilities," *Br J Gen Pract.*, vol. 68, no. 668, pp. 143–144, 2018.

[31] E. A. Abbe, A. E. Khandani, and A. W. Lo, "Privacy-preserving methods for sharing financial risk exposures," *American Economic Review*, vol. 102, no. 3, pp. 65–70, 2012.

[32] P. Voigt and A. v. d. Bussche, *The EU General Data Protection Regulation (GDPR): A Practical Guide*, 1st ed. Springer Publishing Company, Incorporated, 2017.

[33] H. Zhang, M. Cisse, Y. N. Dauphin, and D. Lopez-Paz, "mixup: Beyond empirical risk minimization," in *ICLR*, 2018.

[34] L. Zhang, Z. Deng, K. Kawaguchi, A. Ghorbani, and J. Zou, "How does mixup help with robustness and generalization?" in *ICLR*, 2021.

[35] N. Papernot, P. McDaniel, A. Sinha, and M. Wellman, "Towards the science of security and privacy in machine learning," *arXiv:1611.03814*, 2016.

[36] A. Lamb, V. Verma, J. Kannala, and Y. Bengio, "Interpolated adversarial training: Achieving robust neural networks without sacrificing too much accuracy," in *Proceedings of the 12th ACM Workshop on Artificial Intelligence and Security*, 2019, pp. 95–103.

[37] A. Laugros, A. Caplier, and M. Ospici, "Addressing neural network robustness with mixup and targeted labeling adversarial training," *arXiv preprint arXiv:2008.08384*, 2020.

[38] T. Pang, K. Xu, and J. Zhu, "Mixup inference: Better exploiting mixup to defend adversarial attacks," in *ICLR*, 2019.

[39] A. N. Bhagoji, D. Cullina, C. Sitawarin, and P. Mittal, "Enhancing robustness of machine learning systems via data transformations," in *2018 52nd Annual Conference on Information Sciences and Systems (CISS)*. IEEE, 2018, pp. 1–5.

[40] W. Xu, D. Evans, and Y. Qi, "Feature squeezing: Detecting adversarial examples in deep neural networks," *arXiv preprint arXiv:1704.01155*, 2017.

[41] C. Guo, M. Rana, M. Cisse, and L. van der Maaten, "Countering adversarial images using input transformations," in *International Conference on Learning Representations*, 2018.

[42] E. Raff, J. Sylvester, S. Forsyth, and M. McLean, "Barrage of random transforms for adversarially robust defense," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019, pp. 6528–6537.

[43] P. Tabacof and E. Valle, "Exploring the space of adversarial images," in *2016 International Joint Conference on Neural Networks (IJCNN)*. IEEE, 2016, pp. 426–433.

[44] C. Xie, J. Wang, Z. Zhang, Z. Ren, and A. Yuille, "Mitigating adversarial effects through randomization," in *International Conference on Learning Representations*, 2018.

[45] M. Cheng, Q. Lei, P.-Y. Chen, I. Dhillon, and C.-J. Hsieh, "Cat: Customized adversarial training for improved robustness," *arXiv:2002.06789*, 2020.

[46] S. Thulasidasan, G. Chennupati, J. A. Bilmes, T. Bhattacharya, and S. Michalak, "On mixup training: Improved calibration and predictive uncertainty for deep neural networks," in *NeurIPS*, 2019, pp. 13 888–13 899.

[47] D. Berthelot, N. Carlini, I. Goodfellow, N. Papernot, A. Oliver, and C. A. Raffel, "Mixmatch: A holistic approach to semi-supervised learning," in *NeurIPS*, 2019, pp. 5050–5060.

[48] D. Berthelot, N. Carlini, E. D. Cubuk, A. Kurakin, K. Sohn, H. Zhang, and C. Raffel, "Remixmatch: Semi-supervised learning with distribution matching and augmentation anchoring," in *ICLR*, 2020.

[49] J. Kim, W. Choo, H. Jeong, and H. O. Song, "Co-mixup: Saliency guided joint mixup with supermodular diversity," in *ICLR*, 2021.
[50] N. Carlini and D. A. Wagner, "Towards evaluating the robustness of neural networks," in *Symposium on Security and Privacy (SP)*, 2017.
[51] C. Cortes and V. Vapnik, "Support-vector networks," *Machine learning*, vol. 20, no. 3, pp. 273–297, 1995.
[52] G. Elsayed, D. Krishnan, H. Mobahi, K. Regan, and S. Bengio, "Large margin deep networks for classification," *Advances in neural information processing systems*, vol. 31, 2018.
[53] H. Abdi and L. J. Williams, "Principal component analysis," *Wiley interdisciplinary reviews: computational statistics*, vol. 2, no. 4, pp. 433–459, 2010.
[54] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278–2324, 1998.
[55] Y. Netzer, T. Wang, A. Coates, A. Bissacco, B. Wu, and A. Y. Ng, "Reading digits in natural images with unsupervised feature learning," in *NeurIPS Workshop on Deep Learning and Unsupervised Feature Learning*, 2011.
[56] S. Zagoruyko and N. Komodakis, "Wide residual networks," *arXiv:1605.07146*, 2016.
[57] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga *et al.*, "Pytorch: An imperative style, high-performance deep learning library," *NeurIPS*, pp. 8026–8037, 2019.
[58] T. Chen, Z. Zhang, S. Liu, S. Chang, and Z. Wang, "Robust overfitting may be mitigated by properly learned smoothening," in *ICLR*, 2021.
[59] F. Croce and M. Hein, "Reliable evaluation of adversarial robustness with an ensemble of diverse parameter-free attacks," in *ICML*, 2020.
[60] D. Tsipras, S. Santurkar, L. Engstrom, A. Turner, and A. Madry, "Robustness may be at odds with accuracy," in *ICLR*, 2019.
[61] J. Zhang, X. Xu, B. Han, G. Niu, L. Cui, M. Sugiyama, and M. S. Kankanhalli, "Attacks which do not kill training make adversarial learning stronger," in *ICML*, 2020.

Chen Chen is currently working toward the Ph.D. degree in the College of Computer Science, Zhejiang University, China. He received the BS degree in computer science from Zhejiang University in 2017. His research interests include federated learning, adversarial training, multi-label learning, and recommendation systems.

Jingfeng Zhang is currently a postdoctoral researcher at RIKEN-AIP, Tokyo. He obtained his Bachelor degree in computer science at Taishan College in Shandong University in 2016. He obtained his Ph.D. degree in computer science at the School of Computing in the National University of Singapore in 2020. He has worked extensively in machine learning security, specifically in adversarial machine learning. His long-term research interest is making artificial intelligence safe for human beings.

Xilie Xu is currently attending the Ph.D. programme offered by the School of Computing, Department of Computer Science, National University of Singapore. He received the BS degree in computer science from Taishan College, Shandong University in 2021. He mainly focuses on adversarial machine learning.

Lingjuan Lyu is currently a senior research scientist and team leader in Sony AI. She received Ph.D.from the University of Melbourne. She was a winner of the IBM Ph.D. Fellowship Worldwide. Her current research interest is trustworthy machine learning. She had published over 50 papers in top conferences and journals, including NeurIPS, ICLR, etc.

Chaochao Chen obtained his PhD degree in computer science from Zhejiang University, China, in 2016, and he was a visiting scholar in University of Illinois at Urbana-Champaign, during 2014-2015. He is currently a Distinguished Research Fellow at Zhejiang University. Before that, he was a Staff Algorithm Engineer at Ant Group. His research mainly focuses on privacy preserving machine learning, federated learning, and graph machine learning. He has published more than 50 papers in peer reviewed journals and conferences.

Tianhei Hu is currently an associate professor at the College of Computer Science of Zhejiang University, Hangzhou China. He received the Ph.D. degree in computer science from Zhejiang University, Hangzhou, China. His research interests include large-scale data management and mining technologies supporting massive Internet users.

Gang Chen is currently the Dean of the College of Computer Science, Zhejiang University, where he is the professor of computer science. He received the Ph.D. degree in computer science from Zhejiang University, China. He is the Director of Key Laboratory of Intelligent Computing Based Big Data of Zhejiang Province. His research interests include database management technology, intelligent computing based big data and massive internet systems.