Towards Robustness of Deep Neural Networks via Regularization

Yao Li[1] Martin Renqiang Min[2] Thomas Lee[3]
Wenchao Yu[2] Erik Kruus[2] Wei Wang[4] Cho-Jui Hsieh[4]
[1]University of North Carolina, Chapel Hill [2]NEC Labs America, Princeton
[3]University of California, Davis [4]University of California, Los Angeles

Abstract

Recent studies have demonstrated the vulnerability of deep neural networks against adversarial examples. Inspired by the observation that adversarial examples often lie outside the natural image data manifold and the intrinsic dimension of image data is much smaller than its pixel space dimension, we propose to embed high-dimensional input images into a low-dimensional space and apply regularization on the embedding space to push the adversarial examples back to the manifold. The proposed framework is called Embedding Regularized Classifier (ER-Classifier), which improves the adversarial robustness of the classifier through embedding regularization. Besides improving classification accuracy against adversarial examples, the framework can be combined with detection methods to detect adversarial examples. Experimental results on several benchmark datasets show that, our proposed framework achieves good performance against strong adversarial attack methods.

1. Introduction

It has been shown that Deep Neural Networks (DNNs) are vulnerable to adversarial examples that are generated by adding carefully crafted perturbations to original images [44, 27]. This phenomenon brings out security concerns for practical applications of deep learning. Despite many adversarial training methods have been proposed for defending against adversarial examples [35, 50, 38, 39], we propose a novel defense and detection method from a different point of view.

Recent work showed that adversarial examples often lie outside the natural image data manifolds [24, 45]. Since the model was trained using data in the manifold, the model naturally mis-classifies on examples outside the manifold. Therefore, applying regularization to push the adversarial examples back to the natural image data manifold may help improve the robustness of neural networks (see Figure 1). Furthermore, a consensus in the high-dimensional data analysis community is that, a method working well on the high-dimensional data is because the data is not really of high-dimension [30]. Inspired by the observation that the intrinsic dimension of image data is actually much smaller than its pixel space dimension [30] and adversarial examples lie outside the natural image data manifold [24, 45], we propose a defense framework called Embedding Regularized Classifier (ER-Classifier) shown in Figure 2.

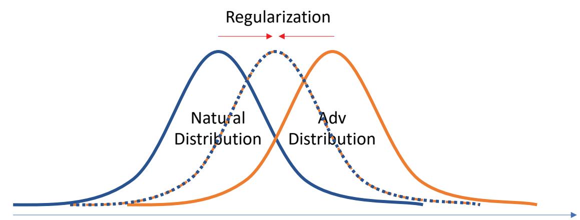

Figure 1. Natural and adversarial images are from different distributions

The difference between ER-Classifier and general deep classifier is that the extracted feature is regularized by a discriminator part in ER-Classifier. To be more specific, we introduce a discriminator in the latent space which tries to separate the **generated code vectors** (output of the encoder network) from the **ideal code vectors** (sampled from a prior distribution, i.e., a standard Gaussian distribution). Employing a similar powerful competitive mechanism as demonstrated by Generative Adversarial Networks [17], the discriminator enforces the embedding space of the model to follow the prior distribution. This regularization process can help remove the effect of adversarial distortion and push the adversarial example back to the natural data manifold. Another difference is that the embedding space dimension of ER-classifier is much smaller, which makes it easier for ER-Classifier to apply regularization.

We compare ER-Classifier with other state-of-the-art defense methods on MNIST, CIFAR10, STL10 and Tiny Imagenet. Experimental results demonstrate that our proposed ER-Classifier outperforms other methods by a large margin. To sum up, this paper makes the following four main contributions:

- A novel unified end-to-end robust deep neural net-

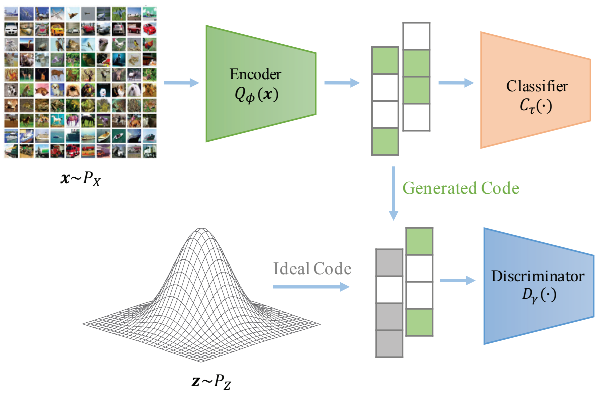

Figure 2. Overview of ER-Classifier framework

work framework against adversarial attacks is proposed, where embedding space regularization is applied to remove the adversarial effect.

- An objective is induced to minimize the optimal transport cost between the true class distribution and the framework output distribution, guiding the framework to project the input image to a low-dimensional space without losing important features for classification.
- A detection process is proposed to further improve the robustness of the framework.
- Extensive experiments demonstrate the robustness of our proposed ER-Classifier framework under the white-box attacks, and show that ER-Classifier outperforms other state-of-the-art approaches on several benchmark image datasets.

2. Related Work

In this section, we summarize related work into two categories: attack methods and defense mechanisms.

Attack Methods. We first discuss different adversarial attack methods [35, 6, 21, 13, 18, 5, 9, 40, 12, 21, 41, 53, 51]. Two main types of attack settings have been considered in previous research: black-box and white-box settings. Under the white-box setting, attackers have all information about the targeted neural network, including network structure and gradients. Many other white-box attack methods have been proposed [35, 6, 21, 13], and among them C&W [6] and PGD [35] attacks have been widely used to test the robustness of machine learning models. Recently, a new attack method called Autoattack [13] was developed, which is parameter-free, computationally affordable and user-independent, to test adversarial robustness. In this paper, we mainly use l_∞-PGD untargeted attack [35] and Autoattack [13] to evaluate the effectiveness of the defense method under the white-box setting. In the black-box setting, the detailed model information, such as gradient and model structure, are not available to the attackers. Some attack methods rely on the predicted scores, such as class probabilities or logits, of the model to craft adversarial examples [9, 22]. Some attacks are more agnostic and only rely on the final decision of the model [3, 10, 20, 8, 32, 7].

Defense Mechanisms. Many works have been done to improve the robustness of deep neural networks [50, 35, 37, 18, 31, 47, 15, 48, 49, 25, 42, 19, 4, 29, 52]. We select five representative methods to compare with the proposed framework. Madry's adversarial training [35] proposed a min-max formulation against adversarial attacks. The proposed model is trained on adversarial examples generated during the training process within the ϵ-ball of input images. A new loss function was introduced in [50] to trade adversarial robustness off classification accuracy. The loss function is used to replace the loss function used in Mary's adversarial training to further improve the robustness of adversarial training. Another effective defense method under the white-box setting is RSE [31]. The authors proposed a "noise layer", which fuses output of each layer with Gaussian noise. Due to the difficulty of defense, some works attempted to detect adversarial examples as alternative solutions. In [16], the author proposed a kernel density based detection framework, in which one kernel density model is fitted for each class on the final layer output. A Logistic Regression model is trained on the kernel density scores to detect adversarial example. Local Intrinsic Dimensionality (LID) characterizes the space-filling capability of the region surrounding a sample, based on the distance to its nearest neighbors within the sample. The author of [33] proposed to use LID as a feature to facilitate the detection of adversarial examples.

Notations. In this paper, we use l_∞ and l_2 distortion metrics to measure similarity. We report l_∞ distance in the normalized $[0, 1]$ space, so that a distortion of 0.031 corresponds to $8/256$, and l_2 distance as the total root-mean-square distortion normalized by the total number of pixels. We use calligraphic letters for sets (i.e., $\mathcal{X}$), capital letters for random variables (i.e., X), and lower case letters for their values (i.e., x). The probability distributions are denoted with capital letters (i.e., P_X) and corresponding densities with lower case letters (i.e., p_X).

3. Embedding Regularized Classifier

3.1. Framework Details

We propose a novel defense framework, ER-Classifier, which aims at improving the robustness of deep neural networks through embedding regularization by training a discriminator to push the embedding space distribution towards a prior distribution. An overview of the framework is shown in Figure 2.

Mathematically, input images $X \in \mathcal{X} = \mathbb{R}^d$ from P_X are projected to a low-dimensional embedding vector $Z \in \mathcal{Z} = \mathbb{R}^k$ through the encoder $\boldsymbol{Q}_\phi$, where the embedding space dimension k is much smaller than the pixel space dimension d. The discriminator $\boldsymbol{D}_\gamma$ discriminates between the generated code $\tilde{Z} \sim \boldsymbol{Q}_\phi(Z|X)$ and the ideal code $Z \sim P_Z$. In this paper we apply the standard Gaussian $\mathcal{N}(\mathbf{0}, \mathbf{1})$ as our prior distribution P_Z, but other priors may be used for different cases. The classifier $\boldsymbol{C}_\tau$ performs classification based on the generated code $\tilde{Z}$, producing output $U \in \mathcal{U} = \mathbb{R}^m$, where m is the number of classes. The label of X is denoted as $Y \in \mathcal{U}$.

The training objective of ER-Classifier is:

$$\inf_{\boldsymbol{Q}(Z|X)\in\mathcal{Q}} \underbrace{\mathbb{E}_{P_X}\mathbb{E}_{\boldsymbol{Q}(Z|X)}\left\{\ell(Y, \boldsymbol{C}(Z))\right\}}_{\text{Classification}} + \underbrace{\lambda\mathcal{D}(\boldsymbol{Q}_Z, P_Z)}_{\text{Regularization}}. \quad (1)$$

The first part of the objective function represents classification loss, where the encoder $\mathcal{Q}$ maps the input image to the embedding vector Z, then the classifier $\boldsymbol{C}$ takes Z as an input and performs prediction. Y is the true label of the input image and ℓ is the cost function. The second part represents the regularization loss, where $\lambda > 0$ is a hyper-parameter controls the trade-off between regularization and classification tasks, and $\mathcal{D}$ can be any arbitrary divergence between the marginal distribution of Z ($\boldsymbol{Q}_Z$) and prior distribution (P_Z).

To estimate the divergence between $\boldsymbol{Q}_Z$ and P_Z, we apply a GAN-based framework, fitting a discriminator to estimate the 1-Wasserstein distance between $\boldsymbol{Q}_Z$ and P_Z:

$$W(\boldsymbol{Q}_Z, P_Z) = \inf_{\Gamma\in\mathcal{P}(\tilde{Z}\sim\boldsymbol{Q}_Z, Z\sim P_Z)} \mathbb{E}_{(\tilde{Z},Z)\sim\Gamma}\|\tilde{Z} - Z\|.$$

When training the framework, the weight clipping method proposed in Wasserstein GAN [1] is applied to help stabilize the training of discriminator $\boldsymbol{D}_\gamma$. Details of the training algorithm is summarized in Algorithm 1.

At training stage, the encoder $\boldsymbol{Q}_\phi$ first maps the input x to a low-dimensional space, resulting in generated code ($\tilde{z}$). Another ideal code (z) is sampled from the prior distribution, and the discriminator $\boldsymbol{D}_\gamma$ discriminates between the ideal code and the generated code. The classifier ($\boldsymbol{C}_\tau$) predicts the image label based on the generated code ($\tilde{z}$). The main goal of ER-Classifier is leveraging embedding space regularization to push adversarial examples back to the natural data manifold, removing adversarial perturbation. Therefore, other defense methods can also benefit from this property. Our framework is trained with min-max robust optimization [35], first searching for adversarial examples with projected gradient descent (PGD) method then optimizing the framework over the adversarial examples.

At inference time, only the encoder $\boldsymbol{Q}_\phi$ and the classifier $\boldsymbol{C}_\tau$ are used. The input image x is first mapped to a low-dimensional space by the encoder ($\tilde{z} = \boldsymbol{Q}_\phi(x)$), then the latent code $\tilde{z}$ is fed into the classifier to predict the label.

3.2. Justifications of the Framework

In the framework, the classifier ($P_{\boldsymbol{C}}(U|Z)$) maps a latent code Z sampled from a fixed distribution (P_Z) in a latent space $\mathcal{Z}$, to the output $U \in \mathcal{U} = \mathbb{R}^m$. The density of ER-Classifier output is defined as follow:

$$p_{\boldsymbol{C}}(u) := \int_{\mathcal{Z}} p_{\boldsymbol{C}}(u|z)p_Z(z)dz, \;\; \forall u \in \mathcal{U}. \quad (2)$$

If the divergence between the distribution of the true class (P_Y) and the distribution of the framework output ($P_{\boldsymbol{C}}$) is minimized, the framework will be doing classification task well, which is the most important goal of the framework.

There are various ways to define the distance or divergence between P_Y and $P_{\boldsymbol{C}}$. In this paper, we turn to the optimal transport theory [46], because it imposes a weak distance between distributions making it easier for a sequence of distributions to converge [1]. Kantorovich's distance induced by the optimal transport problem is given by

$$W_c(P_Y, P_{\boldsymbol{C}}) := \inf_{\Gamma\in\mathcal{P}(Y\sim P_Y, U\sim P_{\boldsymbol{C}})} \mathbb{E}_{(Y,U)\sim\Gamma}\left\{c(Y, U)\right\},$$

where $\mathcal{P}(Y \sim P_Y, U \sim P_{\boldsymbol{C}})$ is the set of all joint distributions of (Y, U) with marginals P_Y and $P_{\boldsymbol{C}}$, and $c(y, u) : \mathcal{U}\times\mathcal{U} \mapsto \mathbb{R}_+$ is any measurable cost function. $W_c(P_Y, P_{\boldsymbol{C}})$ measures the divergence between probability distributions P_Y and $P_{\boldsymbol{C}}$. When the probability measures are on a metric space, the p-th root of W_c is called the p-Wasserstein distance.

Algorithm 1 Training ER-Classifier

1: **Input:** Regularization coefficient $\lambda > 0$, l_∞ distortion ϵ, step-size for PGD attack α.
2: **Note:** ℓ stands for the cross-entropy loss. $\Pi_\epsilon(\cdot, x^o)$ stands for projection to the set $\{x| \|x - x^o\|_\infty \leq \epsilon\}$
3: **while** (ϕ, γ, τ) not converged **do**
4: Sample $\{(x_1^o, y_1^o), ..., (x_n^o, y_n^o)\}$ from the training set. Generate adversarial examples by running PGD for multiple steps, and get $\{(x_1, y_1), ..., (x_n, y_n)\}$
5: Sample $\{z_1, ..., z_n\}$ from the prior P_Z
6: Obtain $\tilde{z}_i$ from $\boldsymbol{Q}_\phi(Z|x_i)$ for $i = 1, ..., n$
7: Update $\boldsymbol{D}_\gamma$ by ascending:

$$\frac{\lambda}{n}\sum_{i=1}^{n} \boldsymbol{D}_\gamma(z_i) - \boldsymbol{D}_\gamma(\tilde{z}_i)$$

8: Update $\boldsymbol{Q}_\phi$ and $\boldsymbol{C}_\tau$ by descending:

$$\frac{1}{n}\sum_{i=1}^{n} \ell(\boldsymbol{C}_\tau(\boldsymbol{Q}_\phi(x_i)), y_i)$$

9: Update $\boldsymbol{Q}_\phi$ by ascending the following objective by 1-step Adam:

$$\frac{\lambda}{n}\sum_{i=1}^{n} \boldsymbol{D}_\gamma(\boldsymbol{Q}_\phi(x_i))$$

Theorem 1 *For $P_{\boldsymbol{C}}$ as defined above with a deterministic $P_{\boldsymbol{C}}(U|Z)$ and any function $\boldsymbol{C} : \mathcal{Z} \mapsto \mathcal{U}$,*

$$W_c(P_Y, P_{\boldsymbol{C}}) = \inf_{\boldsymbol{Q}:\boldsymbol{Q}_Z = P_Z} \mathbb{E}_{P_X} \mathbb{E}_{\boldsymbol{Q}(Z|X)} \left\{ c(Y, \boldsymbol{C}(Z)) \right\},$$

where $c(y,u) : \mathcal{U} \times \mathcal{U} \mapsto \mathbb{R}_+$ is any measurable cost function. $\boldsymbol{Q}_Z$ is the marginal distribution of Z when $X \sim P_X$ and $Z \sim \boldsymbol{Q}(Z|X)$. The proof is presented in Appendix A.

It is obvious that the objective function (1) is relaxed from the r.h.s of Theorem 1, by converting the constraint on $\boldsymbol{Q}_Z$ to a penalty term and using cross-entropy loss as the cost function. Therefore, optimizing over the objective function (1) is equivalent to minimizing the discrepancy between the true class distribution (P_Y) and the output distribution $P_{\boldsymbol{C}}$. A summary of Theorem 1 is that if we are doing optimization well on the objective used, we will be doing classification task well. Theorem 1 requires a deterministic $P_{\boldsymbol{C}}(U|Z)$. However, our proposed framework readily applies to the non-deterministic case. We derived an upper bound on the distance between the two distributions P_Y and $P_{\boldsymbol{C}}$ for the non-deterministic case. See details in Appendix A.

3.3. Detecting Adversarial Examples

Studies showed that adversarial samples come from a distribution that is different from the natural data distribution, that is, adversarial samples do not lie on the data manifold, and DNNs perform correctly only near the manifold of training data [24, 45]. Embedding regularization helps push adversarial examples back to the data manifold but there might be some adversarial examples that are hard to regularize. Therefore, we propose a detection framework, ER-Detector, to filter out these adversarial examples before classification. After training the ER-Classifier framework, all samples from the training set are fed into the encoder to generate the combined code $Z^{'}$, where $Z^{'}$ is generated by concatenating the hidden layer output means of encoder $\boldsymbol{Q}_\phi$ and encoder direct output $\tilde{Z}$. Combined code is used to provide more clues to the detector. Then, one kernel density model is fitted for each class based on $Z^{'}$.

With the kernel density model, we can get density score for any input. A logistic regression model ($\boldsymbol{G}_\beta$) is trained on the density scores to detect adversarial examples, with scores of adversarial examples as positive examples and scores of natural examples as negative examples. Details of the detector training process are shown in Algorithm 2 in Appendix B.

At the inference time, only the encoder $\boldsymbol{Q}_\phi$, classifier $\boldsymbol{C}_\tau$ and detector $\boldsymbol{G}_\beta$ are used. The input image X is first mapped to a low-dimensional space ($\boldsymbol{Q}_\phi(X) \rightarrow \tilde{Z}, Z^{'}$). Then the embedding based detector $\boldsymbol{G}_\beta$ will judge whether the image is an adversarial example. If it is marked as adversarial, the classifier does not need to deal with it, otherwise the classifier will predict the label based on the latent code. The process is designed to detect adversarial examples that manage to "escape" the regularization process and further improve the robustness of the framework.

4. Experiments

For the first set of experiments, we assume the classifier needs to predict a label for each test sample and compare the performance of the proposed algorithm (ER-Classifier) with other state-of-the-art adversarial defense methods. We consider the following datasets: MNIST [28], CIFAR10 [26], STL10 [11] and Tiny Imagenet [14]. See details of data in Appendix C.

Various defense methods have been proposed to improve the robustness of deep neural networks. In Section 4.1, we compare our algorithm with state-of-the-art methods that are robust in the white-box setting. Madry's adversarial training (**Madry's Adv**) has been recognized as one of the most successful defense methods in the white-box setting, as shown by [2]. Random Self-Ensemble (**RSE**) method introduced by [31] adds stochastic components in the neural network, achieving similar performance to Madry's adversarial training algorithm. **Trades** introduced in [50] won first place in the NeurIPS 2018 Adversarial Vision Challenge and outperformed the runner-up apparoach by 11.41% in terms of mean l_2 perturbation distance.

Since the main goal of ER-Classifier is using embedding regularization to improve adversarial robustness, other defense methods can also benefit from this property. The proposed **ER-Classifier** is trained with min-max robust optimization [35]. **ER-Trades** is a variant that combines the proposed framework with the loss function of **Trades** [50]. To demonstrate the regularization effect of ER-Classifier, we include a variant **ER-Classifier$^-$** which trains ER-Classifier without min-max robust optimization. Code for reproduction is available in supplementary material and will be made available at Github later, and network architecture details are included in Appendix G.

In Section 4.5, **ER-Detector**, ER-Classifier combined with detection method described in Section 3.3, is compared with **KD**-Detection [16] and **LID** Detection [33] on MNIST [28] and CIFAR10 [26].

4.1. Evaluate Models Under White-box Attack

In this section, we first evaluate the defense methods against l_∞-PGD untargeted attack [35]. The methods that perform best among the baselines are then evaluated by Autoattack [13]. When tested against PGD attack, defense methods are evaluated under different distortion levels (ϵ), and the larger the distortion the stronger the attack. Depending on the image scale and type, different datasets are sensitive to different strengths of the attack. Models on MNIST are evaluated under distortion level from 0 to 0.4 by

Table 1. Testing accuracy (%) of two defense methods under Autoattack [13] with l_∞ norm. VGG19 is used as the base architecture on CIFAR10. Architectures on STL10 and Tiny Imagenet are similar to VGG19. See details of architectures in Appendix G.

Method	MNIST	CIFAR10	STL10	Tiny
Madry's Adv	69.0	35.6	21.0	9.70
ER-Classifier	79.0	47.2	25.3	11.1

Table 2. Testing accuracy (%) of two defense methods under C&W attack with $l_2 \leq 0.005$.

Method	Testing Accuracy
Defense-GAN	55.0
ER-Classifier	99.1

0.025. Models on CIFAR10 and STL10 are evaluated under $\epsilon \in [0, 0.06, 0.005]$. Models on Tiny Imagenet are evaluated under $\epsilon \in [0, 0.02, 0.002]$. When evaluated by Autoattack, the distortion levels on MNIST, CIFAR10, STL10 and Tiny Imagenet are $0.3, 0.03, 0.03$ and 0.01 respectively. As mentioned in the notation part, all the distortion levels are reported in the normalized $[0, 1]$ space. All the methods are trained for 30 epochs on the datasets. The experimental results against l_∞-PGD are shown in Figure 3.

Based on Figure 3, we can see that ER-Classifier is the most robust one on MNIST and Tiny Imagenet. On STL10, ER-Trades outperforms all the other baselines. On CIFAR10, ER-Classifier- performs better than other baselines when the attack is strong and other baselines except that Trades performs similarly when distortion level (ϵ) is small. The good performances of ER-Classifier and ER-Trades show that the proposed framework can be combined with state-of-the-art defense methods to further improve robustness against adversarial examples.

ER-Classifier without min-max robust optimization can also improve the robustness of deep neural networks. Compare the performance of ER-Classifier$^-$ with the performance of the model without defense method (No Defense), we can see that ER-Classifier$^-$ is much more robust than the model with no defense method on all benchmark datasets. Besides, when the distortion level (ϵ) is large, ER-Classifier$^-$ tends to perform better than some state-of-the-art defense methods on MNIST, CIFAR10 and Tiny Imagenet. This phenomenon is obvious on CIFAR10 and it even performs better than ER-Classifier when the attack strength is strong. The reason might be that without min-max robust optimization, it is easier to regularize the embedding space.

Madry's adversarial training performs best among all the baselines except proposed frameworks. Therefore, we evaluate Madry's adversarial training and ER-Classifier against a stronger white-box attack, Autoattack [13]. Since Autoattack takes a long time to generate adversarial examples, the two methods are compared on $1,000$ random samples from each benchmark dataset. The results are shown in Table 1.

Table 3. Testing accuracy (%) of ER-Classifier under RayS black-box attack [7] with l_∞ norm.

Method	MNIST	CIFAR10	STL10	Tiny
ER-Classifier	75.8	51.0	34.4	21.7

We also compare Defense-GAN [43] with our method ER-Classifier on MNIST. Although Defense-GAN was shown to be partly broken by [2, 23], both ER-Classifier and Defense-GAN leverage the power of generative models to improve adversarial robustness, and comparing to Defense-GAN is important to demonstrate the advantage of our novel Wassserstein distance regularization. Please note that Defense-GAN is not our major comparison baseline in this paper.

Both ER-Classifier and Defense-GAN are evaluated against the l_2-C&W untargeted attack, one of the strongest white-box attacks proposed by [6]. Defense-GAN is evaluated using the method proposed by [2], and the code is available on github[1]. ER-Classifier is evaluated against l_2-C&W untargeted attack with the same hyper-parameter values as those used in the evaluation of Defense-GAN. The results under $l_2 \leq 0.005$ threshold are shown in Table 2. Based on Table 2, ER-Classifier is much more robust than Defense-GAN under the $l_2 \leq 0.005$ threshold. Since [43] did not evaluate Defense-GAN on CIFAR10, STL10 and Tiny Imagenet, without details of GAN structure, we can not compare with Defense-GAN on these datasets.

4.2. Evaluate Models Under Black-box Attack

We evaluate ER-Classifier against a recently proposed black-box attack method called RayS [7] on four benchmark datasets to test the performance of the proposed framework under black-box setting. RayS is an adversarial attack that only requires the target model's prediction. RayS is performed on $1,000$ random samples from each benchmark dataset since the attack process takes a long time. In the experiment, the maximum number of queries is set to be $10,000$[2]. We report the robust accuracy in Table 3.

4.3. Evaluate the Effect of Discriminator

The ER-Classifier framework consists of three parts, where the classification task is done by the encoder $\boldsymbol{Q}_\phi$ and classifier $\boldsymbol{C}_\tau$, and the regularization task is done by the discriminator $\boldsymbol{D}_\gamma$. The encoder and classifier are not different from the general deep neural network classifier. To show that it is embedding space regularization improves the robustness, we fit a framework with only the encoder and classifier part (E-CLA), where the encoder and classifier have the same structures as in ER-Classifier, and com-

[1]Publicly available at `https://github.com/anishathalye/obfuscated-gradients/tree/master/defensegan`

[2]Code available at `https://github.com/uclaml/RayS`

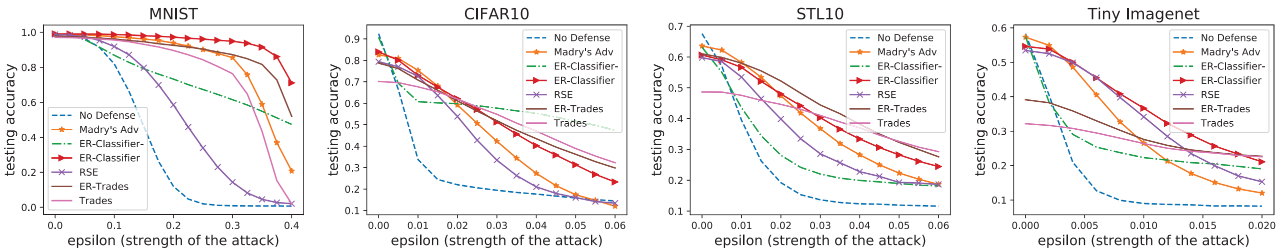

Figure 3. Testing accuracy under l_∞-PGD attack on four different datasets: MNIST, CIFAR10, STL10 and Tiny Imagenet.

pare E-CLA with ER-Classifier framework. For a fair comparison, both structures are trained without min-max robust optimization. The results are shown in Figure 4.

Based on Figure 4, we can observe that ER-Classifier is much more robust than E-CLA structure on MNIST, CIFAR10 and Tiny Imagenet. It is also more robust on STL10 but not that much. The reason might be that there are only $5,000$ training images in STL10 and the resolution is 96×96. Therefore, it is harder to learn a good embedding with a limited amount of images. However, even when the number of training images is limited, ER-Classifier is still much more robust than the E-CLA structure. This observation demonstrates that regularization on the embedding space helps improve the adversarial robustness. Notice that the performance of the E-CLA structure is similar to the performance of the model without defense method on CIFAR10, STL10 and Tiny Imagenet, and worse on MNIST, which means the robustness of ER-Classifier does not come from the structure design.

Variational auto-encoder can project the images to low-dimensional space and use Kullback–Leibler divergence loss to regularize the embedding distribution, which does not need discriminator structure. Therefore, we also tried VAE-CLA, which applies Variational auto-encoder structure to do the projection and regularization. The experimental results in Figure 4 show that VAE-CLA does not perform as well as ER-Classifier. Based on the observation of the Kullback–Leibler loss and classification loss during the training process, it seems difficult for VAE-CLA to balance between the two tasks.

4.4. Embedding Visualization

In this section, we compare the embedding learned by Encoder+Classifier structure (E-CLA) and the embedding learned by ER-Classifier without min-max robust optimization on several datasets. We first generate embedding of testing data using the encoder ($\tilde{z} = \boldsymbol{Q}_\phi(x)$), then project the embedding ($\tilde{z}$) to 2-D space by tSNE [34]. Adversarial images (x_{adv}) are generated using l_∞-PGD attack. The adversarial embedding is generated by feeding the adversarial images into the encoder ($\tilde{z}_{adv} = \boldsymbol{Q}_\phi(x_{adv})$). Finally, we project the adversarial embedding ($\tilde{z}_{adv}$) to 2-D space. The results are shown in Figures 5 and 6. The first two plots are embedding visualization for E-CLA, and the last two plots are the embedding visualization for ER-Classifier. In adversarial embedding visualization plots, the mis-classified point is marked as “down triangle”, which means the PGD attack successfully changed the prediction, and the correctly classified point is marked as “point”, which means the attack fails.

We can see that E-CLA can learn a good embedding on natural images of MNIST. In the first plot of Figure 5, embedding for different classes are well separated on the 2D space, but under adversarial attack (second plot of Figure 5), some points of different classes are mixed together. However, ER-Classifier can generate good separated embedding on both natural and adversarial images (last two plots of Figure 5). On CIFAR10, the E-CLA can not generate good separated embedding on either natural or adversarial images (first two plots of Figure 6), while ER-Classifier can generate good separated embedding for both (last two plots of Figure 6).

4.5. ER-Detector

In this section, we compare the performance of the proposed detection method (ER-Detector) with baseline adversarial example detection methods on MNIST and CIFAR10 to show that the ER-Classifier framework can be combined with a detection method to further improve the adversarial robustness. There are two versions of ER-Detector: **ER1** performs classification on the direct output $\tilde{Z}$ and **ER2** performs classification on the combined code $Z^{'}$.

4.5.1 Setup and Criteria

For both MNIST and CIFAR10, deep neural networks without any defense methods are used as baseline nets for KD-Detection and LID detection methods. The networks are trained on the designated training set. The designated testing set is split into *set I* (20%) and *set II* (80%). The detectors for KD, LID and ER-Detector, are trained on *set I*, then evaluated on *set II* (80%).

The detection methods are evaluated against FGSM, PGD and C&W attacks, which are frequently used to benchmark the detection methods. For FGSM and PGD, the l_∞ distortion levels (ϵ) are 0.3 on MNIST and 0.03 on CIFAR10. Those are standard values used in many previous

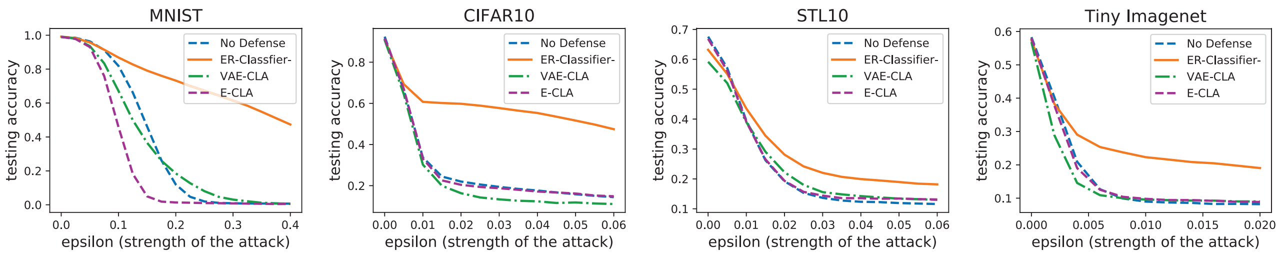

Figure 4. Testing accuracy of E-CLA, VAE-CLA and ER-Classsifier^{-} under l_∞-PGD attack on four different datasets: MNIST, CIFAR10, STL10 and Tiny Imagenet.

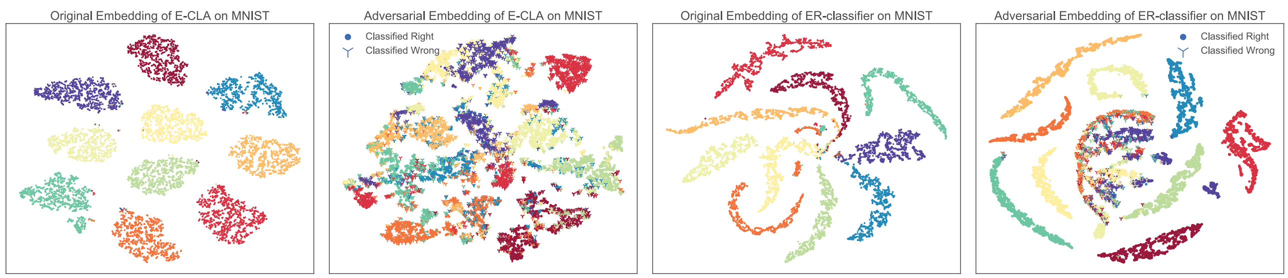

Figure 5. 2D embeddings for E-CLA and ER-Classifier on MNIST. Larger visualization figures are available in Appendix E.

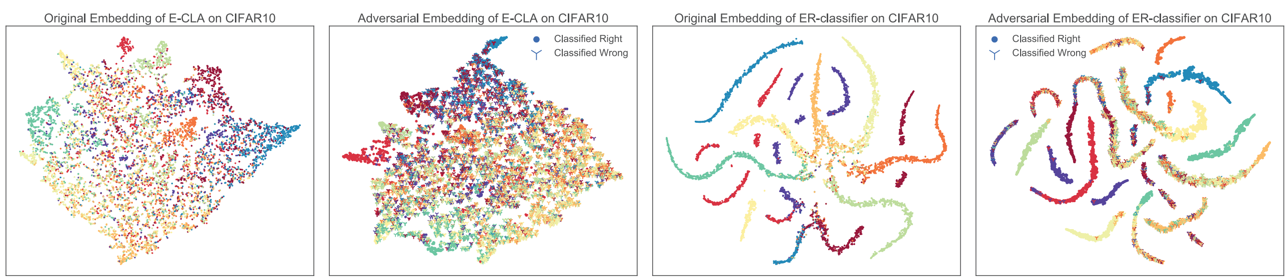

Figure 6. 2D embeddings for E-CLA and ER-Classifier on CIFAR10. Larger visualization figures are available in Appendix E.

Table 4. Criteria for evaluating the detection methods

Criteria	Description	Formula
ACC-ADV	detecting accuracy for adversarial examples	ACC-ADV= $\frac{n_{adv}}{N}$
ACC-NADV	detecting accuracy for non-adversarial examples	ACC-NADV= $\frac{m_{nadv}}{M}$
ACC-CLA	classification accuracy for adversarial examples not detected	ACC-CLA= $\frac{n_{cla}}{N-n_{adv}}$
ACC-COM	combined accuracy for adversarial example	ACC-COM= $\frac{n_{cla}+n_{adv}}{N}$

papers [2, 36, 33]. As for C&W attacks, the l_2 distortions are bounded by 0.007 for MNIST and 0.004 for CIFAR10.

Assume that there are N adversarial examples, and M natural examples. n_{adv} represents the number of adversarial examples correctly detected by the detection method. m_{nadv} represents the number of non-adversarial examples correctly found by the detection method. Out of the undetected adversarial examples, n_{cla} is the number of them that are correctly classified. Table 4 lists the criteria used to evaluate the performance of the detection methods.

The detection method should not detect the non-adversarial examples as adversarial ones, i.e. the false positive rate should not be high. The higher the ACC-NADV, the lower the false positive rate. Besides, the detection method should also recognize the adversarial examples correctly, and if not detected, it should classify them correctly. Since ACC-ADV only considers the detection accuracy of adversarial examples, it does not fully represent the "ability" of detectors in handling adversarial examples. Instead, ACC-COM represents the accuracy for detecting and classifying adversarial examples. Therefore, **ACC-NADV** and **ACC-COM** are two important criteria reflecting the overall performance of the detectors.

4.5.2 Detectors Against "Familiar" Adversary

In this part, we test the performance of the detection methods against the adversary that have been "seen" by the detectors, which means that the detectors are first trained with the adversarial examples generated by an attack method, and then tested with adversarial examples crafted by the same attack method. The performance of the detection methods are shown in Table 5. The numbers in the table are percentage (%), and the best ACC-NADV and ACC-COM are marked in **bold**.

Based on Table 5, ERs perform better than KD and LID on MNIST, especially when evaluated against the PGD attacks. KD also performs well when tested against the C&W attack, but the accuracy for detecting non-adversarial examples is lower than those of ERs. ERs also perform better

Table 5. Performance on MNIST and CIFAR10 against PGD, FGSM and C&W attacks.

Data	Metric	**PGD**				**FGSM**				**C&W**			
		KD	LID	ER1	ER2	KD	LID	ER1	ER2	KD	LID	ER1	ER2
MNIST	ACC-NADV	90.71	91.24	97.41	**97.94**	92.99	89.78	94.82	**96.49**	86.39	88.15	95.97	**98.66**
	ACC-COM	29.87	28.12	**63.81**	44.13	78.46	58.33	**81.15**	76.70	**100.00**	52.38	99.91	**100.00**
CIFAR10	ACC-NADV	94.56	94.86	**98.44**	97.54	93.01	92.74	97.62	**97.69**	88.76	91.86	92.13	**93.45**
	ACC-COM	29.33	79.30	80.49	**86.78**	50.04	58.17	59.82	**61.60**	**100.00**	82.03	**100.00**	**100.00**

Table 6. Trained on FGSM adv.examples and tested by adv.examples generated from PGD and C&W.

Data	Metric	**PGD**				**C&W**			
		KD	LID	ER1	ER2	KD	LID	ER1	ER2
MNIST	ACC-NADV	81.41	89.56	**99.91**	99.66	80.28	57.48	93.62	**94.34**
	ACC-COM	51.23	28.58	**60.09**	41.87	**100.00**	11.18	**100.00**	**100.00**
CIFAR10	ACC-NADV	98.71	90.36	**99.98**	99.91	**94.53**	46.06	88.79	88.48
	ACC-COM	23.71	60.90	63.86	**72.20**	10.06	87.65	**100.00**	**100.00**

Table 7. Trained on PGD adv.examples and tested by adv.examples generated from FGSM and C&W.

Data	Metric	**FGSM**				**C&W**			
		KD	LID	ER1	ER2	KD	LID	ER1	ER2
MNIST	ACC-NADV	**97.42**	91.18	77.98	92.09	90.58	81.33	88.99	**91.12**
	ACC-COM	57.05	47.28	**99.10**	82.31	10.05	27.01	**100.00**	**100.00**
CIFAR10	ACC-NADV	97.89	94.97	**98.19**	97.54	**89.18**	84.41	81.70	79.07
	ACC-COM	40.37	36.63	59.13	**60.86**	**100.00**	10.06	**100.00**	**100.00**

Table 8. Trained on C&W adv.examples and tested by adv.examples generated from FGSM and PGD.

Data	Metric	**FGSM**				**PGD**			
		KD	LID	ER1	ER2	KD	LID	ER1	ER2
MNIST	ACC-NADV	95.95	78.82	97.78	**98.84**	87.12	88.31	**99.99**	99.94
	ACC-COM	67.63	21.87	**76.65**	56.94	39.21	13.99	**58.89**	40.07
CIFAR10	ACC-NADV	85.73	86.86	**100.00**	99.84	94.22	87.54	**100.00**	99.98
	ACC-COM	**60.89**	52.07	56.31	59.91	29.72	19.37	53.05	**68.18**

than KD and LID on CIFAR10 against all three attacks. The ACC-COMs of ERs are better than those of KD and LID, which means ERs do well in detecting and classifying adversarial examples. Overall, ERs perform better than the baseline methods in the "familiar" adversary setting.

4.5.3 Cross Attack

We now evaluate the performances of the detectors under the setting that the detectors are trained on adversarial examples generated by one attack method, and tested on adversarial examples crafted by another attack method. This is a more realistic setting since the detection system cannot predict what kind of attack strategy will be used by the adversary. Therefore, the performances of detection methods under the cross-attack setting are important. The experimental results are shown in Tables 6-8 ("adv.examples" stands for "adversarial examples".)

Based on Tables 6-8, ERs perform better than KD and LID on MNIST in cross-attack setting, especially when trained on PGD adversarial examples and attacked by C&W. When trained on PGD adversarial examples and attacked by FGSM, KD performs better in terms of ACC-NADV. But the corresponding ACC-COM of KD is much worse than those of ERs. Taking both ACC-NADV and ACC-COM into consideration, ER2 performs best when trained on PGD adversarial examples and attacked by FGSM.

In general, ERs perform better than KD and LID on CIFAR10 in cross-attack setting. Based on Table 6, when trained on FGSM adversarial examples and attacked by PGD on CIFAR10, ERs perform better than the baseline methods. When attacked by C&W, the ACC-NADV of KD is higher than those of ERs. However, the corresponding ACC-COM of KD is much lower than those of ERs. The ACC-COM of KD is only 10.06% while the ACC-COMs of ERs are 100%. Taking both criteria into consideration, ERs perform better than KD and LID when trained on FGSM adversarial examples and attacked by other methods on CIFAR10. Based on Table 7, when trained on PGD adversarial examples and attacked by FGSM on CIFAR10, ERs perform better than KD and LID. However, when attacked by C&W, KD performs the best and ERs perform slightly worse in terms of ACC-NADV. Based on Table 8, when trained on C&W adversarial examples and attacked by PGD on CIFAR10, ERs perform better than other baseline methods. When attacked by FGSM, ACC-COM of KD is slightly better than those of ERs. The difference is not significant as the ACC-COM of ER2 is 59.91% and that of KD is 60.89%. However, ACC-NADV of KD is much worse than those of ERs. The ACC-NADV of KD is 85.73% and those of ERs are 100% and 99.84%. Therefore, taking both criteria into consideration, ER2 performs the best.

See more experimental results on adversarial detection in Appendix D. ER-Detector also performs well against high-confidence adversarial examples. Due to the page limit, the results of comparing detectors against high confidence adversarial examples and the results of evaluating the effect of regularization on detection performance are moved to Appendix D.

5. Conclusion

In this paper, we propose a new defense framework, ER-Classifier, which improves the robustness of deep neural networks through embedding regularization. A discriminator is trained to minimize the discrepancy between the embedding space distribution and the prior distribution. Theoretical analysis shows that our framework is not distracted from the main goal of the model, to do classification well. We empirically show that ER-Classifier is more robust than other state-of-the-art defense methods on several benchmark datasets. Future work will include further exploration of the low-dimensional space to improve the robustness of deep neural networks.

Acknowledgements This work is partially supported by NSF under CCF-1934568, DMS-1916125, DMS-2113605, IIS-1901527, IIS-2008173 and IIS-2048280.

References

[1] Martin Arjovsky, Soumith Chintala, and Léon Bottou. Wasserstein generative adversarial networks. In Doina Precup and Yee Whye Teh, editors, *Proceedings of the 34th International Conference on Machine Learning*, volume 70 of *Proceedings of Machine Learning Research*, pages 214–223, International Convention Centre, Sydney, Australia, 06–11 Aug 2017. PMLR.

[2] 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*, pages 274–283. PMLR, 2018.

[3] Wieland Brendel, Jonas Rauber, and Matthias Bethge. Decision-based adversarial attacks: Reliable attacks against black-box machine learning models. *International Conference on Learning Representations*, 2018.

[4] Vivek B.S. and R. Venkatesh Babu. Single-step adversarial training with dropout scheduling. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2020.

[5] Nicholas Carlini and David Wagner. Adversarial examples are not easily detected: Bypassing ten detection methods. In *Proceedings of the 10th ACM Workshop on Artificial Intelligence and Security*, pages 3–14. ACM, 2017.

[6] Nicholas Carlini and David Wagner. Towards evaluating the robustness of neural networks. In *Security and Privacy (SP), 2017 IEEE Symposium on*, pages 39–57. IEEE, 2017.

[7] Jinghui Chen and Quanquan Gu. Rays: A ray searching method for hard-label adversarial attack. In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pages 1739–1747, 2020.

[8] Jianbo Chen, Michael I Jordan, and Martin J Wainwright. Hopskipjumpattack: A query-efficient decision-based attack. In *2020 ieee symposium on security and privacy (sp)*, pages 1277–1294. IEEE, 2020.

[9] Pin-Yu Chen, Huan Zhang, Yash Sharma, Jinfeng Yi, and Cho-Jui Hsieh. Zoo: Zeroth order optimization based black-box attacks to deep neural networks without training substitute models. In *Proceedings of the 10th ACM Workshop on Artificial Intelligence and Security*, pages 15–26. ACM, 2017.

[10] Minhao Cheng, Simranjit Singh, Patrick H. Chen, Pin-Yu Chen, Sijia Liu, and Cho-Jui Hsieh. Sign-opt: A query-efficient hard-label adversarial attack. In *International Conference on Learning Representations*, 2020.

[11] Adam Coates, Andrew Ng, and Honglak Lee. An analysis of single-layer networks in unsupervised feature learning. In *Proceedings of the fourteenth international conference on artificial intelligence and statistics*, pages 215–223, 2011.

[12] Francesco Croce and Matthias Hein. Sparse and imperceivable adversarial attacks. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 4724–4732, 2019.

[13] Francesco Croce and Matthias Hein. Reliable evaluation of adversarial robustness with an ensemble of diverse parameter-free attacks. In *International Conference on Machine Learning*, pages 2206–2216. PMLR, 2020.

[14] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on*, pages 248–255. Ieee, 2009.

[15] Guneet S Dhillon, Kamyar Azizzadenesheli, Zachary C Lipton, Jeremy Bernstein, Jean Kossaifi, Aran Khanna, and Anima Anandkumar. Stochastic activation pruning for robust adversarial defense. *International Conference on Learning Representations*, 2018.

[16] Reuben Feinman, Ryan R Curtin, Saurabh Shintre, and Andrew B Gardner. Detecting adversarial samples from artifacts. *arXiv preprint arXiv:1703.00410*, 2017.

[17] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In *Advances in neural information processing systems*, pages 2672–2680, 2014.

[18] Ian Goodfellow, Jonathon Shlens, and Christian Szegedy. Explaining and harnessing adversarial examples. In *International Conference on Learning Representations*, 2015.

[19] Sven Gowal, Chongli Qin, Po-Sen Huang, Taylan Cemgil, Krishnamurthy Dvijotham, Timothy Mann, and Pushmeet Kohli. Achieving robustness in the wild via adversarial mixing with disentangled representations. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2020.

[20] Yiwen Guo, Ziang Yan, and Changshui Zhang. Subspace attack: Exploiting promising subspaces for query-efficient black-box attacks. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019.

[21] Qian Huang, Isay Katsman, Horace He, Zeqi Gu, Serge Belongie, and Ser-Nam Lim. Enhancing adversarial example transferability with an intermediate level attack. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 4733–4742, 2019.

[22] Andrew Ilyas, Logan Engstrom, Anish Athalye, and Jessy Lin. Black-box adversarial attacks with limited queries and information. In *International Conference on Machine Learning*, pages 2137–2146. PMLR, 2018.

[23] Andrew Ilyas, Ajil Jalal, Eirini Asteri, Constantinos Daskalakis, and Alexandros G Dimakis. The robust manifold defense: Adversarial training using generative models. *arXiv preprint arXiv:1712.09196*, 2017.

[24] Susmit Jha, Uyeong Jang, Somesh Jha, and Brian Jalaian. Detecting adversarial examples using data manifolds. In *MILCOM 2018-2018 IEEE Military Communications Conference (MILCOM)*, pages 547–552. IEEE, 2018.

[25] Christoph Kamann and Carsten Rother. Increasing the robustness of semantic segmentation models with painting-by-numbers. In *European Conference on Computer Vision*, pages 369–387. Springer, 2020.

[26] Alex Krizhevsky and Geoffrey Hinton. Learning multiple layers of features from tiny images. Technical report, Citeseer, 2009.

[27] Alexey Kurakin, Ian Goodfellow, Samy Bengio, et al. Adversarial examples in the physical world, 2016.

[28] Yann LeCun. The mnist database of handwritten digits. *http://yann. lecun. com/exdb/mnist/*, 1998.

[29] Saehyung Lee, Hyungyu Lee, and Sungroh Yoon. Adversarial vertex mixup: Toward better adversarially robust generalization. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2020.

[30] Elizaveta Levina and Peter J Bickel. Maximum likelihood estimation of intrinsic dimension. In *Advances in neural information processing systems*, pages 777–784, 2005.

[31] Xuanqing Liu, Minhao Cheng, Huan Zhang, and Cho-Jui Hsieh. Towards robust neural networks via random self-ensemble. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 369–385, 2018.

[32] Yujia Liu, Seyed-Mohsen Moosavi-Dezfooli, and Pascal Frossard. A geometry-inspired decision-based attack. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, October 2019.

[33] Xingjun Ma, Bo Li, Yisen Wang, Sarah M Erfani, Sudanthi Wijewickrema, Grant Schoenebeck, Dawn Song, Michael E Houle, and James Bailey. Characterizing adversarial subspaces using local intrinsic dimensionality. *International Conference on Learning Representations*, 2018.

[34] Laurens van der Maaten and Geoffrey Hinton. Visualizing data using t-sne. *Journal of machine learning research*, 9(Nov):2579–2605, 2008.

[35] Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu. Towards deep learning models resistant to adversarial attacks. In *International Conference on Learning Representations*, 2018.

[36] Jan Hendrik Metzen, Tim Genewein, Volker Fischer, and Bastian Bischoff. On detecting adversarial perturbations. *International Conference on Learning Representations*, 2017.

[37] Taesik Na, Jong Hwan Ko, and Saibal Mukhopadhyay. Cascade adversarial machine learning regularized with a unified embedding. In *International Conference on Learning Representations*, 2018.

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

[39] Tianyu Pang, Xiao Yang, Yinpeng Dong, Kun Xu, Jun Zhu, and Hang Su. Boosting adversarial training with hypersphere embedding. *arXiv preprint arXiv:2002.08619*, 2020.

[40] Nicolas Papernot, Patrick McDaniel, Ian Goodfellow, Somesh Jha, Z Berkay Celik, and Ananthram Swami. Practical black-box attacks against machine learning. In *Proceedings of the 2017 ACM on Asia Conference on Computer and Communications Security*, pages 506–519. ACM, 2017.

[41] Haonan Qiu, Chaowei Xiao, Lei Yang, Xinchen Yan, Honglak Lee, and Bo Li. Semanticadv: Generating adversarial examples via attribute-conditioned image editing. In *European Conference on Computer Vision*, pages 19–37. Springer, 2020.

[42] Arash Rahnama, Andre T. Nguyen, and Edward Raff. Robust design of deep neural networks against adversarial attacks based on lyapunov theory. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2020.

[43] Pouya Samangouei, Maya Kabkab, and Rama Chellappa. Defense-GAN: Protecting classifiers against adversarial attacks using generative models. In *International Conference on Learning Representations*, 2018.

[44] Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian Goodfellow, and Rob Fergus. Intriguing properties of neural networks. In *International Conference on Learning Representations*, 2014.

[45] Thomas Tanay and Lewis Griffin. A boundary tilting persepective on the phenomenon of adversarial examples. *arXiv preprint arXiv:1608.07690*, 2016.

[46] Cédric Villani. *Optimal transport: old and new*, volume 338. Springer Science & Business Media, 2008.

[47] Cihang Xie, Jianyu Wang, Zhishuai Zhang, Zhou Ren, and Alan Yuille. Mitigating adversarial effects through randomization. In *International Conference on Learning Representations*, 2018.

[48] Cihang Xie, Jianyu Wang, Zhishuai Zhang, Yuyin Zhou, Lingxi Xie, and Alan Yuille. Adversarial examples for semantic segmentation and object detection. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 1369–1378, 2017.

[49] Junfeng Yang and Carl Vondrick. Multitask learning strengthens adversarial robustness. 2020.

[50] Hongyang Zhang, Yaodong Yu, Jiantao Jiao, Eric Xing, Laurent El Ghaoui, and Michael Jordan. Theoretically principled trade-off between robustness and accuracy. In *International Conference on Machine Learning*, pages 7472–7482. PMLR, 2019.

[51] Zhengyu Zhao, Zhuoran Liu, and Martha Larson. Towards large yet imperceptible adversarial image perturbations with perceptual color distance. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2020.

[52] Haizhong Zheng, Ziqi Zhang, Juncheng Gu, Honglak Lee, and Atul Prakash. Efficient adversarial training with transferable adversarial examples. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2020.

[53] Junhua Zou, Zhisong Pan, Junyang Qiu, Xin Liu, Ting Rui, and Wei Li. Improving the transferability of adversarial examples with resized-diverse-inputs, diversity-ensemble and region fitting. In *European Conference on Computer Vision*, pages 563–579. Springer, 2020.