Supplementary: Towards Robustness of Deep Neural Networks via Regularization

A. Theoretical analysis of our framework

The proof of Theorem 1 is adapted from the proof of Theorem 1 in [11]. Consider certain sets of joint probability distributions of three random variables $(X, U, Z) \in \mathcal{X} \times \mathcal{U} \times \mathcal{Z}$. X can be taken as the input images, U as the output of the framework, and Z as the latent codes. $P_{\boldsymbol{C},Z}(U, Z)$ represents a joint distribution of a variable pair (U, Z), where Z is first sampled from P_Z and then U from $P_{\boldsymbol{C}}(U|Z)$. $P_{\boldsymbol{C}}$ defined in (2) is the marginal distribution of U when $(U, Z) \sim P_{\boldsymbol{C},Z}$.

The joint distributions $\Gamma(X, U)$ or couplings between values of X and U can be written as $\Gamma(X, U) = \Gamma(U|X)P_X(X)$ due to the marginal constraint. $\Gamma(U|X)$ can be decomposed into an encoding distribution $\boldsymbol{Q}(Z|X)$ and the generating distribution $P_{\boldsymbol{C}}(U|Z)$, and Theorem 1 mainly shows how to factor it through Z.

In the first part, we will show that if $P_{\boldsymbol{C}}(U|Z)$ are Dirac measures, we have

$$\begin{aligned}&\inf_{\Gamma \in \mathcal{P}(X \sim P_X, U \sim P_{\boldsymbol{C}})} \mathbb{E}_{(X,U)\sim\Gamma} \left\{\ell(f(X), U)\right\} \\ &= \inf_{\Gamma \in \mathcal{P}_{X,U}} \mathbb{E}_{(X,U)\sim\Gamma} \left\{\ell(f(X), U)\right\}, \end{aligned} \tag{1}$$

where $\mathcal{P}(X \sim P_X, U \sim P_{\boldsymbol{C}})$ denotes the set of all joint distributions of (X, U) with marginals $P_X, P_{\boldsymbol{C}}$, and likewise for $\mathcal{P}(X \sim P_X, Z \sim P_Z)$. The set of all joint distributions of (X, U, Z) such that $X \sim P_X$, $(U, Z) \sim P_{\boldsymbol{C},Z}$, and $(U \perp\!\!\!\perp X)|Z$ are denoted by $\mathcal{P}_{X,U,Z}$. $\mathcal{P}_{X,U}$ and $\mathcal{P}_{X,Z}$ denote the sets of marginals on (X, U) and (X, Z) induced by $\mathcal{P}_{X,U,Z}$.

From the definition, it is clear that $\mathcal{P}_{X,U} \subseteq \mathcal{P}(P_X, P_{\boldsymbol{C}})$. Therefore, we have

$$\begin{aligned}&\inf_{\Gamma \in \mathcal{P}(X \sim P_X, U \sim P_{\boldsymbol{C}})} \mathbb{E}_{(X,U)\sim\Gamma} \left\{\ell(f(X), U)\right\} \\ &\leq \inf_{\Gamma \in \mathcal{P}_{X,U}} \mathbb{E}_{(X,U)\sim\Gamma} \left\{\ell(f(X), U)\right\}, \end{aligned} \tag{2}$$

The identity is satisfied if $P_{\boldsymbol{C}}(U|Z)$ are Dirac measures, such as $U = \boldsymbol{C}(Z)$. This is proved by the following Lemma in [11].

Lemma 1 *$\mathcal{P}_{X,U} \subseteq \mathcal{P}(P_X, P_{\boldsymbol{C}})$ with identity if $P_{\boldsymbol{C}}(U|Z = z)$ are Dirac for all $z \in \mathcal{Z}$. (see details in [11].)*

In the following part, we show that

$$\begin{aligned}&\inf_{\Gamma \in \mathcal{P}_{X,U}} \mathbb{E}_{(X,U)\sim\Gamma} \left\{\ell(f(X), U)\right\} \\ &= \inf_{\boldsymbol{Q}: \boldsymbol{Q}_Z = P_Z} \mathbb{E}_{P_X} \mathbb{E}_{\boldsymbol{Q}(Z|X)} \left\{\ell(f(X), \boldsymbol{C}(Z))\right\}. \end{aligned} \tag{3}$$

Based on the definition, $\mathcal{P}(P_X, P_{\boldsymbol{C}})$, $\mathcal{P}_{X,U,Z}$ and $\mathcal{P}_{X,U}$ depend on the choice of conditional distributions $P_{\boldsymbol{C}}(U|Z)$, but $\mathcal{P}_{X,Z}$ does not. It is also easy to check that $\mathcal{P}_{X,Z} = \mathcal{P}(X \sim P_X, Z \sim P_Z)$. The tower rule of expectation, and the conditional independence property of $\mathcal{P}_{X,U,Z}$ implies

$$\begin{aligned}&\inf_{\Gamma \in \mathcal{P}_{X,U}} \mathbb{E}_{(X,U)\sim\Gamma} \left\{\ell(f(X), U)\right\} \\ &= \inf_{\Gamma \in \mathcal{P}_{X,U,Z}} \mathbb{E}_{(X,U,Z)\sim\Gamma} \left\{\ell(f(X), U)\right\} \\ &= \inf_{\Gamma \in \mathcal{P}_{X,U,Z}} \mathbb{E}_{P_Z} \mathbb{E}_{X\sim P(X|Z)} \mathbb{E}_{U\sim P(U|Z)} \left\{\ell(f(X), U)\right\} \\ &= \inf_{\Gamma \in \mathcal{P}_{X,U,Z}} \mathbb{E}_{P_Z} \mathbb{E}_{X\sim P(X|Z)} \left\{\ell(f(X), \boldsymbol{C}(Z))\right\} \\ &= \inf_{\Gamma \in \mathcal{P}_{X,Z}} \mathbb{E}_{(X,Z)\sim\Gamma} \left\{\ell(f(X), \boldsymbol{C}(Z))\right\} \\ &= \inf_{\boldsymbol{Q}: \boldsymbol{Q}_Z = P_Z} \mathbb{E}_{P_X} \mathbb{E}_{\boldsymbol{Q}(Z|X)} \left\{\ell(f(X), \boldsymbol{C}(Z))\right\} \end{aligned} \tag{4}$$

Finally, since $Y = f(X)$, it is easy to get

$$\begin{aligned}&\inf_{\Gamma \in \mathcal{P}(Y \sim P_Y, U \sim P_{\boldsymbol{C}})} \mathbb{E}_{(Y,U)\sim\Gamma} \left\{\ell(Y, U)\right\} \\ &= \inf_{\Gamma \in \mathcal{P}(X \sim P_X, U \sim P_{\boldsymbol{C}})} \mathbb{E}_{(X,U)\sim\Gamma} \left\{\ell(f(X), U)\right\} \end{aligned} \tag{5}$$

Now (1), (3) and (5) are proved and the three together prove Theorem 1.

Our proposed framework readily applies to non-deterministic case. If the classifier part is non-deterministic, Lemma 1 provides only the inclusion of sets $\mathcal{P}_{X,U} \subseteq \mathcal{P}(P_X, P_U)$, and we can get an upper bound on the Wasserstein distance between the ground-truth and predicted label distributions:

$$\begin{aligned}&\inf_{\Gamma \in \mathcal{P}(X \sim P_X, U \sim P_{\boldsymbol{C}})} \mathbb{E}_{(X,U)\sim\Gamma} \left\{\ell(f(X), U)\right\} \\ &\leq \inf_{\Gamma \in \mathcal{P}_{X,U}} \mathbb{E}_{(X,U)\sim\Gamma} \left\{\ell(f(X), U)\right\} \\ &\leq \sum_{i=1}^{d} \sigma_i^2 + \inf_{\Gamma \in \mathcal{P}_{X\sim P_X, Z\sim P_Z}} \mathbb{E}_{(X,Z)\sim\Gamma} \left\{\|f(X) - \boldsymbol{C}(Z)\|^2\right\}, \end{aligned} \tag{6}$$

where we assume the conditional distributions $P_{\boldsymbol{C}}(U|Z = z)$ have mean values $\boldsymbol{C}(z) \in \mathbb{R}^d$ and marginal variances $\sigma_1^2, ..., \sigma_d^2 \geq 0$ for all $z \in \mathcal{Z}$, where $\boldsymbol{C} : \mathcal{Z} \to \mathcal{X}$, and $\ell(y, u) = \|y - u\|^2$. The above upper bound is derived by:

$$\begin{aligned}&\inf_{\Gamma \in \mathcal{P}_{X,U}} \mathbb{E}_{(X,U)\sim\Gamma}\left\{\|f(X) - U\|^2\right\} \\ &= \inf_{\Gamma \in \mathcal{P}_{X,U,Z}} \mathbb{E}_{P_Z}\mathbb{E}_{X\sim P(X|Z)}\mathbb{E}_{U\sim P(U|Z)}\{\|f(X) - U\|^2\}\end{aligned} \tag{7}$$

and

$$\begin{aligned}&\mathbb{E}_{U\sim P(U|Z)}\{\|f(X) - U\|^2\} \\ &= \mathbb{E}_{U\sim P(U|Z)}\{\|f(X) - \boldsymbol{C}(Z) + \boldsymbol{C}(Z) - U\|^2\} \\ &= \|f(X) - \boldsymbol{C}(Z)\|^2 + \mathbb{E}_{U\sim P(U|Z)}\{\|\boldsymbol{C}(Z)| - U\|^2\} \\ &+ \mathbb{E}_{U\sim P(U|Z)}\{< f(X) - \boldsymbol{C}(Z), \boldsymbol{C}(Z) - U >\} \\ &= \|f(X) - \boldsymbol{C}(Z)\|^2 + \sum_{i=1}^{d} \sigma_i^2.\end{aligned} \tag{8}$$

In equation (8), the second term of the second last row becomes 0 since the optimization will drive $f(X) - \boldsymbol{C}(Z)$ to zero.

B. Implementation Details

B.1. Detection Model Training Algorithm

See details of training ER-Detector in Algorithm 1.

Algorithm 1 Training the Detection System

Input: Pre-trained encoder $\boldsymbol{Q}_\phi$.
Training Procedure:
Feeding all the training set images $\{x\}_{i=1}^{n_{train}}$ into the encoder $\boldsymbol{Q}_\phi$ and get $\{z^{'}\}_{i=1}^{n_{train}}$.
for $t = 1, ..., m$ **do**, ▷ *m is the number of classes*
    Fit KDE_t based on $Z_t^{'}$, where $Z_t^{'}$ is the set of $z_i^{'}$ with label t.
Generate noisy ($\{x_i^b\}_{i=1}^{n_{setI}}$) and adversarial ($\{x_i^a\}_{i=1}^{n_{setI}}$) examples based on $\{x_i\}_{i=1}^{n_{setI}}$.
for $i = 1, ..., n_{setI}$ **do**
    $d_i = \text{KDE}_t(x_i)$, where the predicted label $\hat{y}_i = t$
    $d_i^b = \text{KDE}_t(x_i^b)$, where the predicted label $\hat{y}_i^b = t$
    $d_i^a = \text{KDE}_t(x_i^a)$, where the predicted label $\hat{y}_i^a = t$
Train Logistic Regression based on $\{d_i\}_{i=1}^{n_{setI}}$ and $\{d_i^b\}_{i=1}^{n_{setI}}$ as negative examples and $\{d_i^a\}_{i=1}^{n_{setI}}$ as positive examples.
End Procedure
Return $\boldsymbol{G}_\beta$

B.2. Kernel Density Estimation

Kernel density estimation (KDE) is used in the detector $\boldsymbol{G}_\beta$ to model the low-dimensional space of the projection system. KDE is an unsupervised technique to estimate unknown probability distribution. Suppose that $z_1, ..., z_n$ are training samples drawn from an unknown probability density $f_Z(z)$. Given z, we can use the following function to estimate the density score at z:

$$\hat{f}_Z(z) = \frac{1}{n}\sum_{i=1}^{n} K_\sigma(z, z_i),$$

where $K_\sigma(\cdot, \cdot)$ stands for kernel functions. In the experiments, one kernel density model is fitted for each class. Therefore, if z is predicted with label t, the samples $\{z\}_{i=1}^{n}$ used to do the estimation are training samples from class t.

In the experiments, we apply the Gaussian kernel with bandwidth σ:

$$K_\sigma(z_1, z_2) \sim \exp(-\|z_1 - z_2\|^2/\sigma^2).$$

The bandwidth parameter affects the "smoothness" of the resulting density. A large bandwidth leads to a very "smooth" density distribution. A small bandwidth usually leads to a "spiky" density distribution.

C. Experimental Details

C.1. Datasets

In this paper, we compare the performance of our proposed algorithm with other state-of-the-art defense methods on several benchmark datasets:

- MNIST [8]: handwritten digit dataset, which consists of $60,000$ training images and $10,000$ testing images. These are 28×28 black and white images in ten different classes.
- CIFAR10 [7]: natural image dataset, which contains $50,000$ training images and $10,000$ testing images in ten different classes. These are low resolution 32×32 color images.
- STL10 [2]: color image dataset similar to CIFAR10, but contains only $5,000$ training images and $8,000$ testing images in ten different classes. The images are of higher resolution 96×96.
- Tiny Imagenet [3]: a subset of Imagenet dataset. Tiny Imagenet has 200 classes, and each class has 500 training images, 50 testing images, making it a challenging benchmark for the defense task. The resolution of the images is 64×64.

Adversarial training parameters on different datasets are shown in Table 1. The parameters are the same for both Madry's adversarial training and ER-Classifier for fair comparison.

Table 1. Parameters of adversarial training.

Data	ϵ	Number of Iterations
MNIST	0.3	40
CIFAR10	0.03	20
STL10	0.03	20
Tiny Imagenet	0.01	10

C.2. Dimension of Embedding Space

One important hyper-parameter of ER-Classifier is the dimension of the embedding space. If the dimension is too small, important features are "collapsed" onto the same dimension, and if the dimension is too large, it will be hard to regularize the embedding and result in too much noise and instability. The maximum likelihood estimation of intrinsic dimension proposed by [9][1] is used to calculate the intrinsic dimension of each image dataset, serving as a guide for selecting the embedding dimension. The sample size used in calculating the intrinsic dimension is $1,000$, and increasing the sample size does not influence the results much. Based on the intrinsic dimension estimated by [9], we test several different values around the estimated intrinsic dimension and evaluate the models against the l_∞-PGD attack. All models are trained without min-max robust optimization, and the experimental results are shown in Figure 1.

The final embedding dimension is chosen based on robustness, number of parameters, and testing accuracy when there is no attack. The final embedding dimensions and estimated intrinsic dimensions are shown in Table 2.

Table 2. Pixel space dimension, intrinsic dimension calculated by [9], and final embedding dimension used.

Data	Data dim.	Estimated Intrinsic dim.	Embedding dim.
MNIST	$1 \times 28 \times 28$	13	4
CIFAR10	$3 \times 32 \times 32$	17	16
STL10	$3 \times 96 \times 96$	20	16
Tiny Imagenet	$3 \times 64 \times 64$	19	20

Based on Figure 1, the embedding dimension close to the estimated intrinsic dimension usually offers better results except on MNIST. One explanation may be that MNIST is a simple handwritten digit dataset, so performing classification on MNIST may not require that many dimensions.

C.3. Epsilon Selection

Epsilon (ϵ) is an important hyper-parameter for adversarial training. When doing Madry's adversarial training, we test the model robustness with different ϵ and choose the best one. The experiment results are shown in Figure 2.

Based on Figure 2, we use $\epsilon = 0.3, 0.03, 0.03$ in Madry's adversarial training on MNIST, CIFAR10 and STL10 respectively. For Tiny Imagenet, we use $\epsilon = 0.01$. To make a fair comparison, we use the same ϵ when training ER-Classifier.

[1]Code publicly available at `https://github.com/OFAI/hub-toolbox-python3`

C.4. Prior Selection

ER-Classifier does not have restrictions on the choice of prior. However, it is interesting to explore the performances of different priors.

Three different prior distributions are tested on MNIST and CIFAR10 datasets. They are standard Gaussian, Uniform$(-3, 3)$ and Cauchy$(0, 1)$, where Cauchy$(0, 1)$ has the same support as standard Gaussian but is heavy tailed and 99.7% of the standard Gaussian points lies within $[-3, 3]$. All the models are trained without min-max robust optimization, and the experimental results are shown in Figure 3. Based on the results, all three priors work well, but standard Gaussian performs best on both datasets.

Ding et al. [5] prove that adversarial robustness is sensitive to the input data distribution, and if the data is uniformly distributed in the input space, no algorithm can achieve good robustness. They also empirically show that cornered/concentrated data distributions tend to achieve better robustness. Standard Gaussian pushes the embedding space to be more concentrated, making the valid perturbation space to be smaller. This may explain why Gaussian prior performs a little bit better than two other priors.

D. More Detection Experiments

D.1. Testing Against High Confidence Adversarial Examples

In [1], the author pointed out that LID detection method is not able to detect high confidence adversarial examples generated by C&W attack. This might be a concern for all detection methods. Therefore, in this part, we apply C&W to generate high confidence adversarial examples and test the detectors against them.

We generate 100 high confidence adversarial examples for both datasets and evaluate the detectors against them. The confidences are 9 and 20 for MNIST and CIFAR10 respectively. Based on the experiment, if the confidence goes higher than those thresholds, it is difficult to generate adversarial examples on the corresponding datasets within the L_2 thresholds. The performances of the detectors are shown in Tables 3-5.

Table 3. Trained on FGSM adv.examples and tested on high confidence adv.examples generated by C&W.

Methods	MNIST		CIFAR10	
	ACC-NADV	ACC-COM	ACC-NADV	ACC-COM
KD	63.50	11.00	**86.50**	16.00
LID	57.00	11.00	43.00	16.00
ER1	80.00	**100.00**	82.50	**100.00**
ER2	**83.00**	**100.00**	77.00	16.00

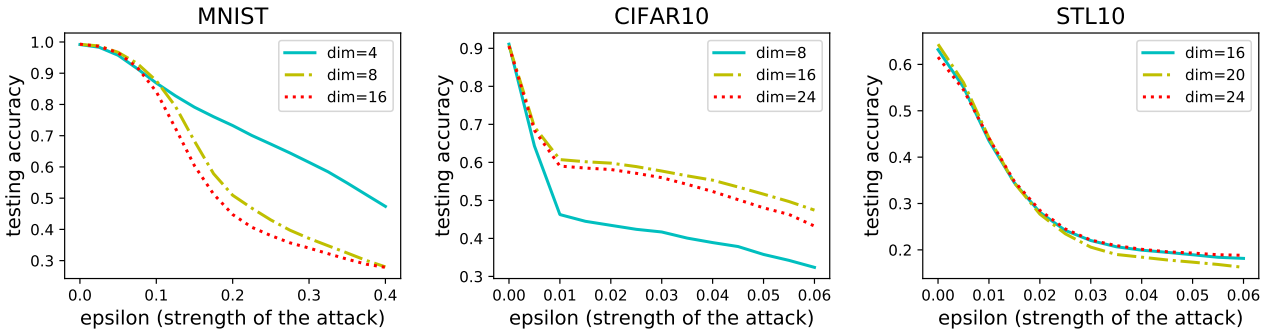

Figure 1. Testing accuracy of models with different embedding dimensions under l_∞-PGD attack.

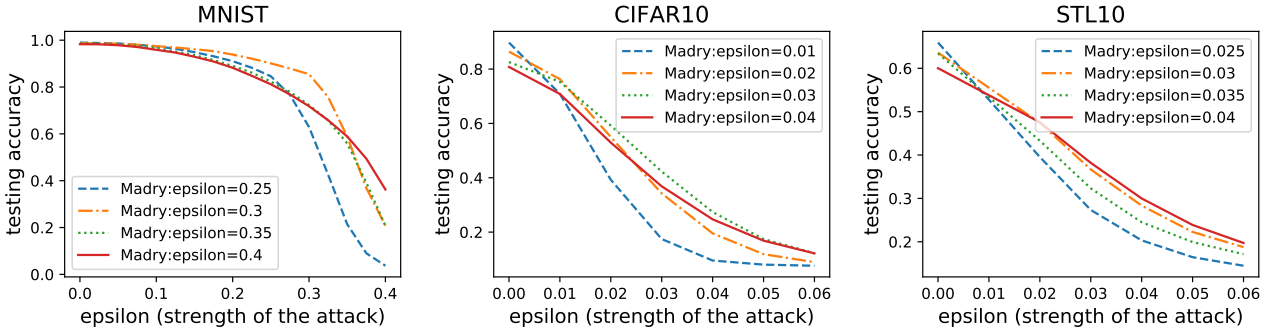

Figure 2. Testing accuracy of models with different ϵ on MNIST, CIFAR10 and STL10.

Table 4. Trained on PGD adv.examples and tested on high confidence adv.examples generated by C&W.

Methods	MNIST		CIFAR10	
	ACC-NADV	ACC-COM	ACC-NADV	ACC-COM
KD	74.00	11.00	74.50	16.00
LID	**83.00**	11.00	**94.00**	36.00
ER1	73.00	**100.00**	79.00	**100.00**
ER2	80.50	**100.00**	72.50	16.00

Table 5. Trained on C&W adv.examples and tested on high confidence adv.examples generated by C&W.

Methods	MNIST		CIFAR10	
	ACC-NADV	ACC-COM	ACC-NADV	ACC-COM
KD	69.50	11.00	74.00	16.00
LID	**98.50**	11.00	74.00	16.00
ER1	87.00	11.00	**85.00**	**100.00**
ER2	89.50	**100.00**	**83.00**	16.00

When tested against high confidence adversarial examples, ER1 can still maintain good performance while other methods are heavily influenced. When trained on FGSM adversarial examples, KD has better ACC-NADV on CIFAR10 while the ACC-COM is much lower than that of ER1. When trained on PGD adversarial examples, LID has better ACC-NADVs but in terms of ACC-COM, it performs much worse than ER1. Similarly, when trained on C&W adversarial examples, LID has better ACC-NADV on MNIST, but the corresponding ACC-COM is worse than that of ER2. Generally speaking, taking both ACC-NADV and ACC-COM into consideration, ERs perform better than the baseline methods under the high confidence setting.

D.2. Effect of Regularization

To show that regularization on the embedding space help improve the robustness of ER-Detector, we fit frameworks with only the encoder and classifier part (ER1$^-$ and ER2$^-$), where the encoder and classifier have the same structures as in ER-Detector. The results are shown in Table 6. Considering both ACC-NADV and ACC-COM, ER-Detector performs much better than structures without regularization. Instead of fitting a discriminator to regularize the embedding space, Kullback-Leibler distance is also tried. However, the KL loss and classification loss cannot converge together during the training.

E. Loss Surface Visualization

To show that ER-Classifier outperforms Madry's adversarial training is not because of weird loss surface, we visualized the loss surfaces of ER-Classifier and Madry's adversarial training on CIFAR10. Following the implementation in [6], we vary the data input along a linear space defined by the sign of the input gradient and a random Rademacher vector, where the x- and y- axes represent the magnitude of

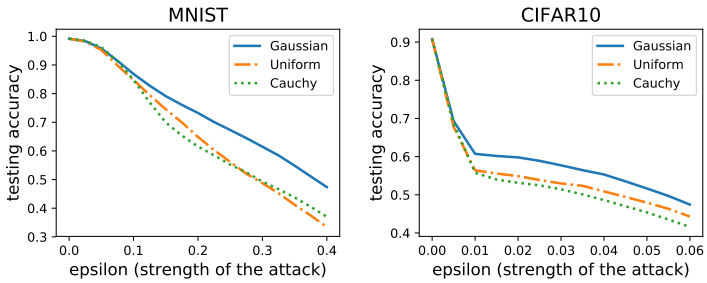

Figure 3. Testing accuracy of models with different prior distributions under l_∞-PGD attack.

Table 6. Performances of ERs$^-$ and ERs on MNIST and CIFAR10 against PGD, FGSM and C&W attacks.

	Criteria	PGD				FGSM				C&W			
		ER1$^-$	ER1	ER2$^-$	ER2	ER1$^-$	ER1	ER2$^-$	ER2	ER1$^-$	ER1	ER2$^-$	ER2
MNIST	ACC-NADV	99.95	97.41	100.00	97.94	95.97	94.82	99.52	96.49	93.42	95.97	91.55	98.66
	ACC-COM	1.11	63.81	1.41	44.13	31.54	81.15	8.67	76.70	9.61	99.91	9.75	100.00
CIFAR10	ACC-NADV	97.68	98.44	96.85	97.54	95.03	97.62	96.77	97.69	100.00	92.13	100.00	93.45
	ACC-COM	58.74	80.49	49.11	86.78	52.31	59.82	55.83	61.60	10.06	100.00	11.54	100

the perturbation added in each direction and the z-axis represents the loss. Based on the results in Figure 4. We can see that both methods have smooth loss surfaces.

F. Embedding Visualization

Larger versions of embedding visualization plots are shown in Figure 5 and Figure 6.

G. Model Structure

MNIST, CIFAR10, STL10 and TinyImagenet classifier structures used for baseline methods are shown in Table 7. Details of ER-Classifier structures on the four benchmark datasets are shown in Table 8. The discrminator architecture is the same on four datasets: four fully connected layers. See code of model in code files in supplementary code folder.

Table 7. Architectures of baseline networks.

Dataset	Architecture
MNIST [8]	4Conv. + 4FC layers
CIFAR10 [7]	VGG19 with BN [10]
STL10 [2]	6Conv. with BN and 5Max.Pool +4FC
Tiny Imagenet [4]	13Conv. with BN and 5Max.Pool +4FC

Table 8. Architectures of ER-Classifier Encoders.

Dataset	Encoder Architecture	Classifier Architecture
MNIST [8]	4Conv. with BN + 1FC	3FC with BN
CIFAR10 [7]	16Conv. + 1FC	4FC
STL10 [2]	6Conv. with BN +1FC	3FC
Tiny Imagenet [4]	13Conv. with BN and 5Max.Pool +1FC	3FC

References

[1] 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.

[2] 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.

[3] 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.

[4] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota, June 2019. Association for Computational Linguistics.

[5] Gavin Weiguang Ding, Kry Yik Chau Lui, Xiaomeng Jin, Luyu Wang, and Ruitong Huang. On the sensitivity of adversarial robustness to input data distributions. In *International Conference on Learning Representations*, 2019.

[6] Logan Engstrom, Andrew Ilyas, and Anish Athalye. Evaluating and understanding the robustness of adversarial logit pairing. *arXiv preprint arXiv:1807.10272*, 2018.

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

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

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

[10] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.

[11] Ilya Tolstikhin, Olivier Bousquet, Sylvain Gelly, and Bernhard Schoelkopf. Wasserstein auto-encoders. *International Conference on Learning Representations*, 2018.

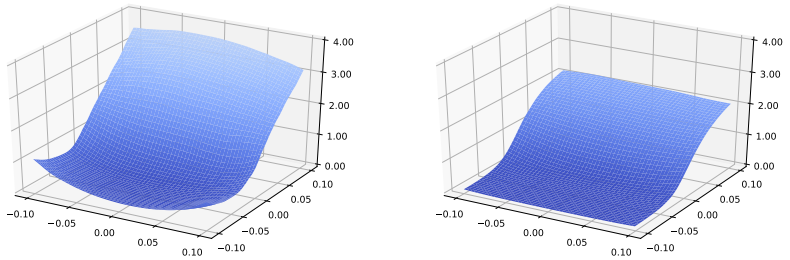

Figure 4. Loss surfaces of ER-Classifier and Madry's adversarial training. (Left: ER-Classifier, Right: Madry's adversarial training)

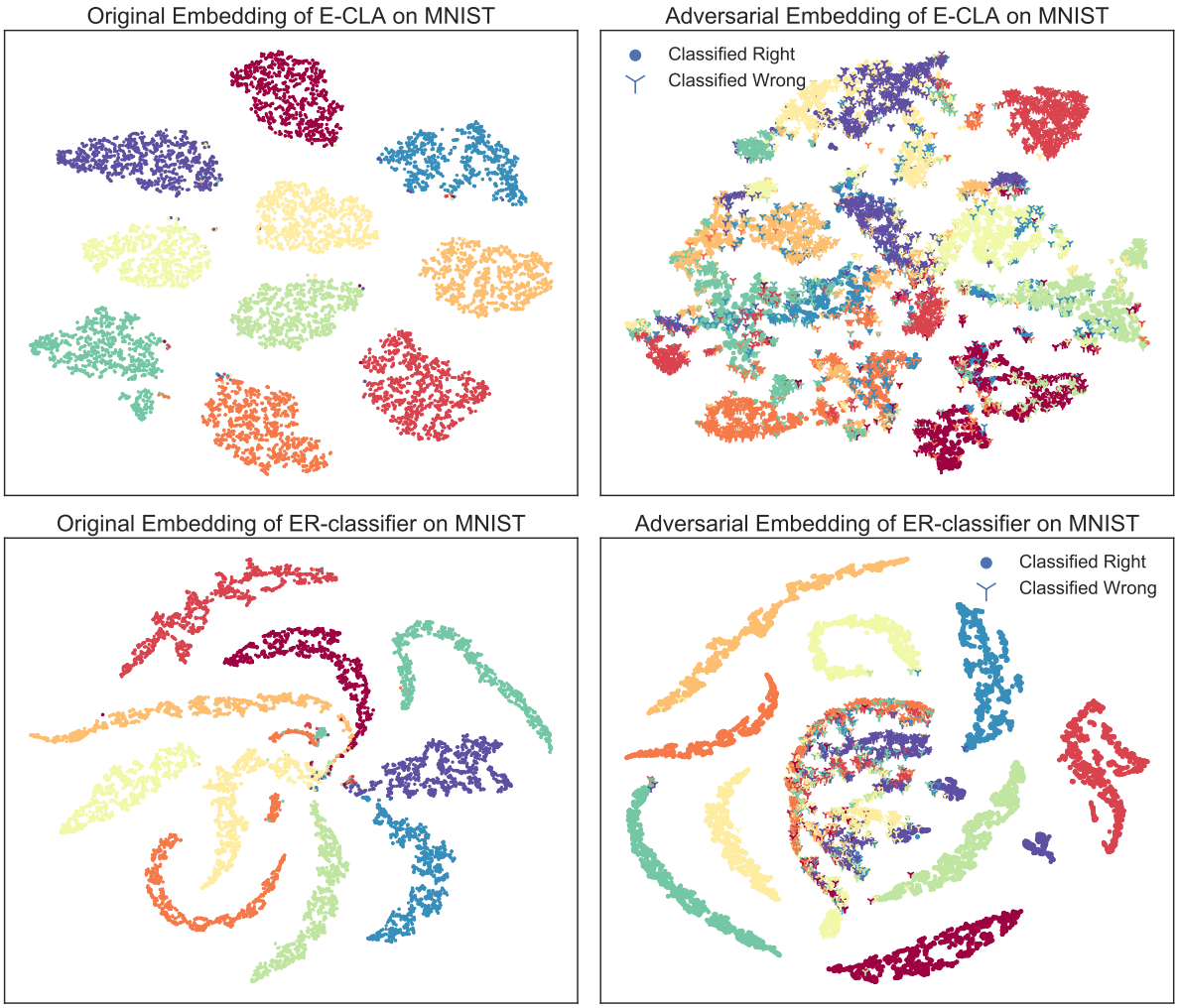

Figure 5. 2D embeddings for E-CLA and ER-Classifier on MNIST.

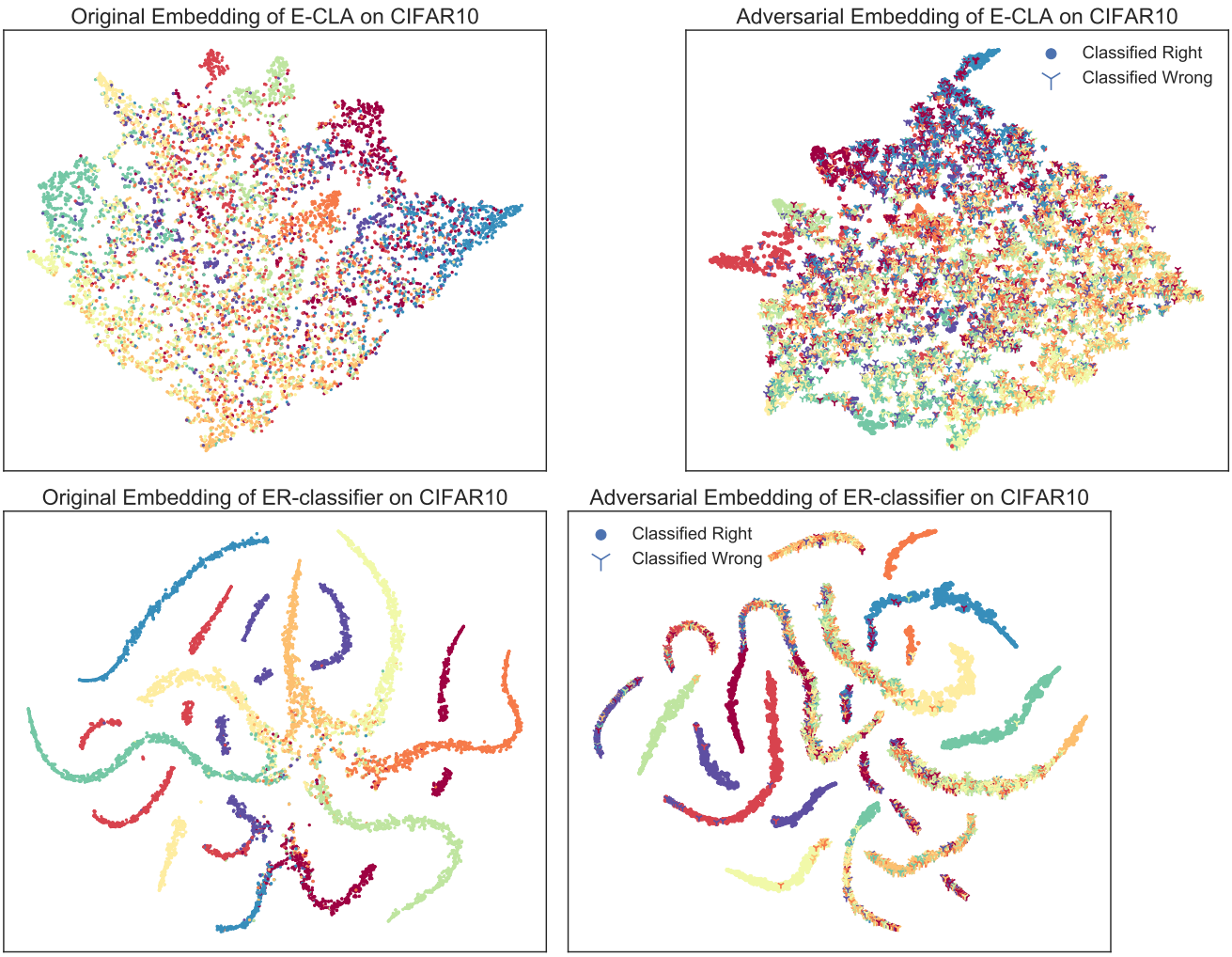

Figure 6. 2D embeddings for E-CLA and ER-Classifier on CIFAR10.