Appendix: Two Coupled Rejection Metrics Can Tell Adversarial Examples Apart

Tianyu Pang[1], Huishuai Zhang[2], Di He[2], Yinpeng Dong[1], Hang Su[1], Wei Chen[3], Jun Zhu[1*], Tie-Yan Liu[2]
[1]Dept. of Comp. Sci. and Tech., Institute for AI, BNRist Center, THBI Lab, Tsinghua University [2]MSRA [3]ICT, CAS
{pty17, dyp17}@mails.tsinghua.edu.cn, {suhangss, dcszj}@tsinghua.edu.cn, {huishuai.zhang, dihe, wche, tyliu}@microsoft.com

A. Proof

In this section, we provide proofs for the proposed Theorem 1, and Theorem 2.

A.1. Proof of Theorem 1

Proof. The conditions in Theorem 1 can be written as $f_\theta(x_1)[y_1^m] > \frac{1}{2-\xi}$, $y_1^m = y_1$ and $f_\theta(x_2)[y_2^m] > \frac{1}{2-\xi}$, $y_2^m \neq y_2$, where $\xi \in [0,1)$. Since $A_\phi(x)$ is ξ-error at x_1 and x_2, according to Definition 1, at least one of the bounds holds for x_1 and x_2, respectively:

$$\text{Bound (i): } \left|\log\left(\frac{A_\phi(x)}{A_\phi^*(x)}\right)\right| \leq \log\left(\frac{2}{2-\xi}\right);$$

$$\text{Bound (ii): } \left|A_\phi(x) - A_\phi^*(x)\right| \leq \frac{\xi}{2}.$$

For x_1, there is $A_\phi^*(x_1) = 1$. Then if bound (i) holds, we can obtain

$$\begin{aligned}\text{R-Con}(x_1) &= f_\theta(x_1)[y_1^m]\cdot A_\phi(x_1)\\ &> f_\theta(x_1)[y_1^m]\cdot\frac{2-\xi}{2}\\ &> \frac{1}{2-\xi}\cdot\frac{2-\xi}{2} = \frac{1}{2},\end{aligned}$$

and if bound (ii) holds, we can obtain

$$\begin{aligned}\text{R-Con}(x_1) &= f_\theta(x_1)[y_1^m]\cdot A_\phi(x_1)\\ &> f_\theta(x_1)[y_1^m]\cdot\left(1-\frac{\xi}{2}\right)\\ &> \frac{1}{2-\xi}\cdot\frac{2-\xi}{2} = \frac{1}{2}.\end{aligned}$$

Similarly for x_2, there is $f_\theta(x_2)[y_2^m]\cdot A_\phi^*(x_2) = f_\theta(x_2)[y_2]$. Then if bound (i) holds, we can obtain

$$\begin{aligned}\text{R-Con}(x_2) &= f_\theta(x_2)[y_2^m]\cdot A_\phi(x_2)\\ &= f_\theta(x_2)[y_2^m]\cdot A_\phi^*(x_2)\cdot\frac{A_\phi(x_2)}{A_\phi^*(x_2)}\\ &< f_\theta(x_2)[y_2]\cdot\frac{2}{2-\xi}\\ &< \left(1-\frac{1}{2-\xi}\right)\cdot\frac{2}{2-\xi}\\ &= \frac{2-2\xi}{(2-\xi)^2} < \frac{1}{2},\end{aligned}$$

where it is easy to verify that $\frac{2-2\xi}{(2-\xi)^2}$ is monotone decreasing in the interval of $\xi \in [0,1)$. If bound (ii) holds for x_2, we can obtain

$$\begin{aligned}&\text{R-Con}(x_2)\\ &= f_\theta(x_2)[y_2^m]\cdot A_\phi(x_2)\\ &< f_\theta(x_2)[y_2^m]\cdot\left(\frac{f_\theta(x_2)[y_2]}{f_\theta(x_2)[y_2^m]}+\frac{\xi}{2}\right)\\ &= f_\theta(x_2)[y_2] + f_\theta(x_2)[y_2^m]\cdot\frac{\xi}{2}\\ &= f_\theta(x_2)[y_2]\cdot\left(1-\frac{\xi}{2}\right) + \left(f_\theta(x_2)[y_2]+f_\theta(x_2)[y_2^m]\right)\cdot\frac{\xi}{2}\\ &< \left(1-\frac{1}{2-\xi}\right)\cdot\left(1-\frac{\xi}{2}\right)+\frac{\xi}{2} = \frac{1}{2}.\end{aligned}$$

Thus we have proven $\text{R-Con}(x_1) > \frac{1}{2} > \text{R-Con}(x_2)$. □

A.2. Proof of Theorem 2

Proof. Since $A_\phi^*(x)$ is naturally bounded in $[0,1]$ for any input x, and $A_\phi(x)$ is bounded in $[0,1]$ by model design, we denote $\{B_0, B_1, \cdots, B_S\}$ as $S+1$ points in $[0,1]$, where $B_0 = 0$ and $B_s = 1$. These $S+1$ points induce S bins or intervals, i.e., $I_s = [B_{s-1}, B_s]$ for $s = 1, \cdots, S$. When $A_\phi(x)$ is ξ-error at x, we consider the cases of bound (i) and bound (ii) hold, respectively, as detailed below:

Bound (i) holds. We construct the bins in a geometric manner, where $B_s = \frac{2}{2-\xi}\cdot B_{s-1}$ and we set $B_1 = \rho$ be a

[*]Corresponding author.

rounding error. Note that we have

$$\rho \cdot \left(\frac{2}{2-\xi}\right)^{S-2} < 1 \leq \rho \cdot \left(\frac{2}{2-\xi}\right)^{S-1},$$

thus we can derive that

$$S = \left\lceil \frac{\log \rho^{-1}}{\log\left(\frac{2}{2-\xi}\right)} \right\rceil + 1.$$

It is easy to find that if $A_\phi(x)$ and $A^*_\phi(x)$ locate in the same bin, then bound (i) holds. Therefore, this regression task can be substituted by a classification task of classes $N_1 = \left\lceil \frac{\log \rho^{-1}}{\log\left(\frac{2}{2-\xi}\right)} \right\rceil + 1$.

Bound (ii) holds. In this case, we construct the bins in an arithmetic manner, where $B_s = B_{s-1} + \frac{\xi}{2}$. Then we have

$$(S-1) \cdot \frac{\xi}{2} < 1 \leq S \cdot \frac{\xi}{2},$$

thus we can derive that

$$S = \left\lceil \frac{2}{\xi} \right\rceil.$$

It is easy to find that if $A_\phi(x)$ and $A^*_\phi(x)$ locate in the same bin, then bound (ii) holds. So this regression task can be substituted by a classification task of classes $N_2 = \left\lceil \frac{2}{\xi} \right\rceil$. $\square$

B. More backgrounds

Adversarial training. In recent years, adversarial training (AT) has become the critical ingredient for the state-of-the-art robust models [10, 16, 18]. Many variants of AT have been proposed via adopting the techniques like ensemble learning [45, 59, 67], metric learning [31, 41], generative modeling [28, 61], curriculum learning [5], semi-supervised learning [2, 8], and self-supervised learning [11, 12, 27, 43]. Other efforts include tuning AT mechanisms by universal perturbations [47, 52], reweighting misclassified samples [63, 73] or multiple threat models [40, 58]. Accelerating the training procedure of AT is another popular research routine, where recent progresses involve reusing the computations [51, 71], adaptive adversarial steps [62, 72] or one-step training [3, 30, 32, 64].

Adversarial detection. Instead of correctly classifying adversarial inputs, another complementary research routine aims to detect / reject them [15, 25, 34, 35, 42, 49, 70]. Previous detection methods mainly fall into two camps, i.e., statistic-based and model-based. Statistic-based methods stem from the features learned by standardly trained models. These statistics include density ratio [22], kernel density [20, 44], prediction variation [66], mutual information [53, 54], Fisher information [74], local intrinsic dimension [38], activation invariance [37], and feature attributions [57, 68]. As for the model-based methods, the auxiliary detector could be a sub-network [9, 13, 55], a Gaussian mixture model [1, 29, 36], or an additional generative model [4, 19, 50].

C. More analyses

In this section, we provide implementation details of the BCE loss, toy examples to intuitively illustrate the effects of temperature tuning, and analyze the role of T-Con in randomized classifiers.

C.1. Implementation of the BCE loss

For notation simplicity, we generally denote the BCE objective as

$$\text{BCE}(f \parallel g) = -g_{\dagger} \cdot \log f - (1-g_{\dagger}) \cdot \log(1-f), \quad (1)$$

where the subscript $\dagger$ indicates stopping gradients, an operation usually used to stabilize the training processes [24]. We show that the stopping-gradient operations can lead to unbiased optimal solution for the classifier. Specifically, taking PGD-AT+RR as an example, the training objective is minimizing

$$\mathbb{E}_{p(x,y)}\left[\mathcal{L}_{\text{CE}}\left(f_\theta(x), y\right) + \text{BCE}\left(f_\theta(x)[y^m] \cdot A_\phi(x) || f_\theta(x)[y]\right)\right]$$

w.r.t. ϕ and θ, where we use $p(x, y)$ to represent adversarial data distribution. Note that the optimal solution of minimizing $\mathcal{L}_{\text{CE}}\left(f_\theta(x), y\right)$ is $f_\theta(x)[y] = p(y|x)$, but if we do not stop gradients of $f_\theta(x)[y]$ in the RR term (BCE loss), then the optimal θ of the entire PGD-AT+RR objective no longer satisfies $f_\theta(x)[y] = p(y|x)$, i.e., in this case RR will introduce bias on the optimal solution of classifier. Thus, stopping gradients on $f_\theta(x)[y]$ in the RR term can avoid affecting the training of classifier.

C.2. Toy examples on temperature tuning

Assume that there are three classes, and the confidence / T-Con on x_1 and x_2 are

$$\mathcal{M}(x_1; \tau) = \frac{e^{\frac{a_1}{\tau}}}{e^{\frac{a_1}{\tau}} + e^{\frac{b_1}{\tau}} + e^{\frac{c_1}{\tau}}}; \mathcal{M}(x_2; \tau) = \frac{e^{\frac{a_2}{\tau}}}{e^{\frac{a_2}{\tau}} + e^{\frac{b_2}{\tau}} + e^{\frac{c_2}{\tau}}}.$$

Let $a_1 = a_2 = 0$, $b_1 = 3$, $c_1 = -1000$, $b_2 = c_2 = 2$, it is easy to numerically compute that

$$\mathcal{M}(x_1; \tau = 1) < \mathcal{M}(x_2; \tau = 1);$$

$$\mathcal{M}(x_1; \tau = 2) > \mathcal{M}(x_2; \tau = 2).$$

This mimics the case of T-Con for misclassified inputs. We can simply choose $a_1 = a_2 = 0$, $b_1 = -1$, $c_1 = -1000$, $b_2 = c_2 = -2$ to mimic the case of confidence.

Table 1. Results of different hyperparameters for the KD and LID methods on CIFAR-10, under $(\ell_\infty, 8/255)$ threat model. For KD, we restore the features on $1,000$ correctly classified training samples in each class. For LID, we restore the features on totally $10,000$ correctly classified training samples.

Method	Hyperparameters	ROC-AUC Clean	ROC-AUC PGD-10
KD	$\sigma = 10^{-1}$	0.562	0.545
	$\sigma = 10^{-2}$	0.609	0.581
	$\sigma = 10^{-3}$	**0.618**	**0.587**
LID	$K = 100$	0.686	0.622
	$K = 200$	0.699	0.638
	$K = 300$	0.706	0.648
	$K = 400$	0.710	0.654
	$K = 500$	**0.712**	0.658
	$K = 600$	0.711	**0.661**
	$K = 700$	0.709	0.661
	$K = 800$	0.706	0.660
	$K = 1000$	0.695	0.653
	$K = 2000$	0.603	0.590

Table 2. Results of different hyperparameters for the KD and LID methods on CIFAR-100. The basic settings are the same as in Table 1, except that for KD, we restore 100 correctly classified training features in each class.

Method	Hyperparameters	ROC-AUC Clean	ROC-AUC PGD-10
KD	$\sigma = 10^{1}$	0.522	0.517
	$\sigma = 1$	**0.549**	**0.532**
	$\sigma = 10^{-1}$	0.500	0.479
	$\sigma = 10^{-2}$	0.473	0.453
	$\sigma = 10^{-3}$	0.477	0.457
LID	$K = 10$	0.662	0.652
	$K = 20$	**0.674**	**0.668**
	$K = 40$	0.672	0.667
	$K = 60$	0.668	0.661
	$K = 80$	0.659	0.652
	$K = 100$	0.652	0.644
	$K = 200$	0.615	0.607
	$K = 300$	0.584	0.578
	$K = 400$	0.559	0.551
	$K = 500$	0.537	0.529

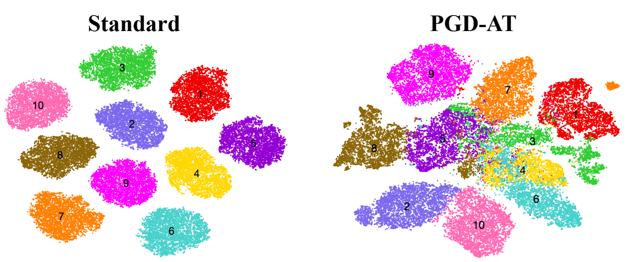

Figure 1. t-SNE visualization of the learned features on CIFAR-10. The irregular distributions of adversarially learned features make previous statistic-based detection methods less effective.

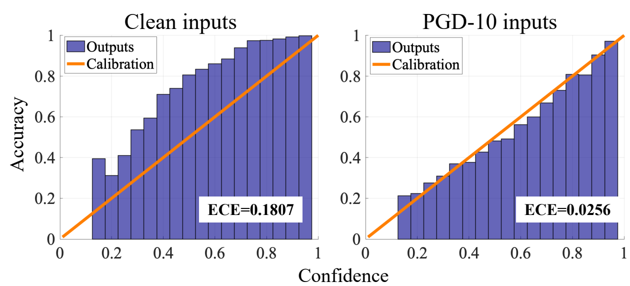

Figure 2. Reliability diagrams for an adversarially trained ResNet-18 on CIFAR-10, and the expected calibration error (ECE) [26]. The model outputs are well calibrated.

C.3. The role of T-Con in randomized classifiers

It has been shown that randomized classifiers like Bayesian neural networks (BNNs) [34, 48] and DNNs with randomized smoothing [14] can benefit adversarial robustness. In practice, these methods are usually implemented by a Monte-Carlo ensemble with finite sampled weights or inputs. We construct an abstract classification process that involves both deterministic and randomized classifiers.

Specifically, the returned label y^s is sampled from a categorical distribution as $p(y^s = l) = f_\theta(x)[l]$, where in this case, $f_\theta(x)$ is a deterministic mapping either explicitly (e.g., for DNNs) or implicitly (e.g., for BNNs) defined. For example, considering a BNN $g_\omega(x)$ where $\omega \sim q_\theta(\omega)$, the induced $f_\theta(x)$ can be written as

$$f_\theta(x)[l] = p\left(l = \arg\max_{y_s} \sum_{n=1}^{N} g_{\omega_n}(y_s|x)\right), \quad (2)$$

which is the probability measure that the returned label is l from the Bayes ensemble $\sum_{n=1}^{N} g_{\omega_n}(y_s|x)$, under the distributions of $\omega_n \sim q_\theta(\omega), n \in \{1, \cdots, N\}$. In practice, we can obtain empirical estimations on these implicitly defined $f_\theta(x)$ by sampling.

By presetting the temperature τ, the expected accuracy of the returned labels can be written as

$$\mathrm{A}_\tau = \mathbb{E}_{p(x,y)}\mathbb{E}_{y^s}\left[\mathbf{1}_{y^s=y}\right] = \mathbb{E}_{p(x,y)}\left[f_\theta(x)[y]\right], \quad (3)$$

Table 3. Classification accuracy (%) and the ROC-AUC scores on CIFAR-100 under PGD-10 attacks. For KD, we restore the features on 100 correctly classified training samples in each class and use $\sigma = 1$. For LID, we restore the features on totally $10,000$ correctly classified training samples and use $K = 20$. For SNet, the $\lambda = 8$ and coverage is 0.7. For EBD, there is $m_{in} = 6$ and $m_{out} = 3$.

Rejector	Clean		ℓ_∞, $8/255$		ℓ_∞, $16/255$		ℓ_2, $128/255$	
	TPR-95	AUC	TPR-95	AUC	TPR-95	AUC	TPR-95	AUC
Architecture backbone: ResNet-18								
KD	58.20	0.549	30.23	0.532	16.39	0.510	40.67	0.539
LID	59.49	0.674	31.60	0.668	16.86	0.661	42.01	0.658
GDA	57.06	0.416	29.67	0.412	16.17	0.410	39.83	0.416
GDA*	58.98	0.599	31.40	0.593	17.04	0.588	42.10	0.596
GMM	58.06	0.518	30.48	0.505	16.69	0.508	40.68	0.511
SNet	59.68	0.729	33.12	0.743	19.48	0.759	42.72	0.726
EBD	61.44	0.795	34.56	0.776	**20.50**	0.762	44.22	0.777
RR	**64.44**	**0.837**	**35.52**	**0.782**	19.89	**0.767**	**47.03**	**0.802**
Architecture backbone: WRN-34-10								
KD	62.04	0.602	32.59	0.573	18.19	0.559	41.66	0.575
LID	63.17	0.705	33.27	0.672	18.97	0.652	42.97	0.672
GDA	60.12	0.436	31.64	0.426	17.75	0.421	40.52	0.423
GDA*	62.71	0.601	33.79	0.605	18.65	0.575	42.91	0.602
GMM	61.80	0.519	33.33	0.520	18.95	0.529	42.27	0.513
SNet	64.09	0.727	36.14	0.738	22.02	0.753	44.32	0.713
EBD	66.83	0.810	37.76	0.775	21.80	0.743	46.80	0.789
RR	**70.14**	**0.853**	**38.81**	**0.790**	**22.20**	**0.765**	**48.26**	**0.801**

where $\mathbf{1}_{y^s=y}$ is the indicator function, which equals to one if $y^s = y$ and zero otherwise. In the limiting case of $\tau \to 0$, the returned labels are deterministic, and the expected accuracy is $\mathrm{A}_0 = \mathbb{E}_{p(x,y)}[\mathbf{1}_{y^m=y}]$, which degenerates to the traditional definition of accuracy. Note that in the adversarial setting, the Bayes optimal classifier, i.e., $\tau = 0$ may not be an empirically optimal choice. For example, in the cases of $A_0 = 0$, we can still have $A_\tau > 0$ for the non-deterministic classifiers.

D. More technical details and results

In this section, we provide more technical details and results. Our methods are implemented by Pytorch [46], and run on GeForce RTX 2080 Ti GPU workers. The experiments of ResNet-18 are run by single GPU, while those on WRN-34-10 are run by two GPUs in parallel.

D.1. The MLP architecture of $A_\phi(x)$

In our experiments, $A_\phi(x)$ is implemented by the MLP as

$$A_\phi(x) = W_2(\mathbf{ReLU}(\mathbf{BN}(W_1 z + b_1))) + b_2, \qquad (4)$$

where z is the feature vector shared with the classification branch, **BN** is an 1-D batch normalization operation, W_1, b_1 are the parameters of the first linear layer, and W_2, b_2 are the parameters of the second linear layer. For ResNet-18, there is $z \in \mathbb{R}^{512}$, $W_1 \in \mathbb{R}^{256\times512}$, $b_1 \in \mathbb{R}^{256}$, $W_2 \in \mathbb{R}^{1\times256}$, $b_2 \in \mathbb{R}^1$. For WRN-34-10, there is $z \in \mathbb{R}^{640}$, $W_1 \in \mathbb{R}^{320\times640}$, $b_1 \in \mathbb{R}^{320}$, $W_2 \in \mathbb{R}^{1\times320}$, $b_2 \in \mathbb{R}^1$.

Empirically, on ResNet-18, the average running time for PGD-AT is about 316 seconds per epoch, and it for PGD-AT+RR is about 320 seconds per epoch. As to the parameter sizes, saving a ResNet-18 model without/with RR branch uses 44.74 MB/45.27 MB, saving a WRN-34-10 model without/with RR branch uses 184.77 MB/185.59 MB.

D.2. Hyperparameters for baselines

For KD, we restore $1,000$ correctly classified training features in each class and use $\sigma = 10^{-3}$. For LID, we restore a total of $10,000$ correctly classified training features and use $K = 600$. We calculate the mean and covariance matrix on all correctly classified training samples for GDA and GMM. For SelectiveNet, the $\lambda = 8$ and coverage is 0.7. For EBD, there is $m_{in} = 6$ and $m_{out} = 3$.

Kernel density (KD). In [20], KD applies a Gaussian kernel $K(z_1, z_2) = \exp(-\|z_1 - z_2\|_2^2/\sigma^2)$ to compute the similarity between two features z_1 and z_2. There is a hyperparameter σ controlling the bandwidth of the kernel, i.e., the smoothness of the density estimation. In Table 1 and Table 2,

Table 4. Results of different hyperparameters for the SelectiveNet and EBD methods on CIFAR-10. The AT framework is PGD-AT, and the evaluated PGD-10 adversarial inputs are crafted with $\epsilon = 8$.

Method	Hyperparameters	Accuracy (%)		ROC-AUC	
		Clean	PGD-10	Clean	PGD-10
SelectiveNet	$\lambda = 8, c = 0.7$	80.57	53.43	**0.796**	**0.730**
	$\lambda = 8, c = 0.8$	82.16	53.90	0.768	0.716
	$\lambda = 8, c = 0.9$	81.33	53.82	0.757	0.694
	$\lambda = 16, c = 0.7$	81.08	53.62	0.792	0.725
	$\lambda = 16, c = 0.8$	81.72	53.90	0.782	0.722
	$\lambda = 16, c = 0.9$	82.21	54.08	0.751	0.701
	$\lambda = 32, c = 0.7$	79.98	53.52	0.793	0.716
	$\lambda = 32, c = 0.8$	80.60	53.71	0.774	0.711
	$\lambda = 32, c = 0.9$	82.48	53.86	0.750	0.704
EBD	$m_{in} = -5, m_{out} = -23$	overflow			
	$m_{in} = 6, m_{out} = 0$	80.71	52.55	0.831	0.768
	$m_{in} = 6, m_{out} = 3$	81.98	53.89	0.832	0.763

Table 5. Classification accuracy (%) and the ROC-AUC scores on CIFAR-10. The AT framework is PGD-AT and the model architecture is WRN-34-10. For KD, we restore $1,000$ correctly classified training features in each class and use $\sigma = 10^{-3}$. For LID, we restore totally $10,000$ correctly classified training features and use $K = 600$. We calculate mean and covariance matrix on all correctly classified training samples for GDA and GMM. For SNet, the $\lambda = 8$ and coverage is 0.7. For EBD, there is $m_{in} = 6$ and $m_{out} = 3$.

Rejector	Clean		ℓ_∞, 8/255		ℓ_∞, 16/255		ℓ_2, 128/255	
	TPR-95	AUC	TPR-95	AUC	TPR-95	AUC	TPR-95	AUC
KD	85.51	0.759	57.26	0.674	34.87	0.605	67.55	0.695
LID	86.94	0.760	58.53	0.690	35.54	0.642	68.62	0.699
GDA	85.10	0.512	56.47	0.506	34.22	0.482	66.79	0.503
GDA*	87.16	0.694	57.62	0.627	34.66	0.561	68.23	0.637
GMM	88.36	0.747	57.98	0.650	34.79	0.568	68.87	0.667
SNet	88.30	0.803	60.07	0.733	**37.63**	0.695	70.14	0.730
EBD	89.63	0.860	60.96	0.778	36.92	**0.712**	70.97	0.792
RR	**90.74**	**0.897**	**61.48**	**0.783**	36.52	0.698	**72.00**	**0.809**

we report the ROC-AUC scores under different values of σ, where we restore the features of $1,000/100$ correctly classified training samples in each class on CIFAR-10/CIFAR-100, respectively.

Local intrinsic dimensionality (LID). In [38], LID applies K nearest neighbors to approximate the dimension of local data distribution. Instead of computing LID in each mini-batch, we allow the detector to use a total of $10,000$ correctly classified training data points, and treat the number of K as a hyperparameter, as tuned in Table 1 and Table 2.

SelectiveNet (SNet). In [21], the training objective consists of three parts, i.e., the prediction head, the selection head, and the auxiliary head. There are two hyperparameters in SelectiveNet, one is the coverage c, which is the expected value of selection outputs, another one is λ controlling the relative importance of the coverage constraint. In the standard setting, [21] suggest $\lambda = 32$ and $c = 0.8$, while we investigate a wider range of λ and c when incorporating SelectiveNet with the PGD-AT framework, as reported in Table 4.

Energy-based detection (EBD). In [33], the discriminative classifier is implicitly treated as an energy-based model, which returns unnormalized density estimation. The two hyperparameters in EBD are m_{in} and m_{out}, controlling the upper and lower clipping bounds for correctly and wrongly classified inputs, respectively. In Table 4, we tried the setting of $m_{in} = -5, m_{out} = -23$ as used in the original paper, which overflows on ATMs.

D.3. Details on attacking parameters

For **PGD attacks** [39], we use the step size of $2/255$ under ℓ_∞ threat model, and the step size of $16/255$ under ℓ_2 threat model. We apply untargeted mode with 10 restarts. For **CW attacks** [7], we set the binary search steps to be 9 with the initial $c = 0.01$. The iteration steps for each c are $1,000$ with the learning rate of 0.005. Let x, x^* be the clean and adversarial inputs with the pixels scaled to $[0, 1]$. The values reported for CW-ℓ_∞ are $\|x - x^*\|_\infty \times 255$, while those for CW-$\ell_2$ are $\|x - x^*\|_2^2$. The default settings of **AutoAttack** [17] involve 100-steps APGD-CE/APGD-DLR with 5 restarts, 100-steps FAB with 5 restarts, $5,000$ query times for the square attack. For **multi-target attacks** [23], we use 100 iterations and 20 restarts for each of the 9 targeted class, thus the number of total iteration steps on each data point is $100 \times 20 \times 9 = 18,000$. For **GAMA attacks**, we follow the default settings used in the offical code[1].

D.4. More results of WRN-34-10 and CIFAR-100

In Table 5, we use the larger model architecture of WRN-34-10 [69]. We evaluate under PGD-10 ($\ell_\infty, \epsilon = 8/255$) which is seen during training, and unseen attacks with different perturbation constraint ($\epsilon = 16/255$), threat model (ℓ_2). As to the baselines, we choose SNet and EBD since they perform well in the cases of training ResNet-18. In Table 3, we experiment on CIFAR-100, and similarly evaluate under different variants of PGD-10 attacks. We report the results using both ResNet-18 and WRN-34-10 model architectures.

Moreover, to exclude gradient obstruction [6], we do a sanity check by running PGD-10 against PGD-AT+**RR** on CIFAR-10 under $\epsilon = \{8, 16, 32, 64, 128\}/255$, where the model architecture is ResNet-18. The ALL accuracy (%) before rejection is $\{54.40, 33.56, 19.80, 6.71, 0.95\}$, which converges to zero.

D.5. Visualization of adversarially learned features

Although statistic-based detection methods like KD, LID, GDA, and GMM have achieved good performance on STMs against *non-adaptive* or *oblivious* attacks [6], they perform much worse when combined with ATMs. To explain this phenomenon, we plot the t-SNE visualization [60] in Fig. 1 on the standardly and adversarially learned features. As seen, ATMs have much more irregular feature distributions compared to STMs, while this fact breaks the statistic assumptions and rationale of previous statistic-based detection methods. For example, GDA applying a tied covariance matrix becomes unreasonable for ATMs, and this is why after using the conditional covariance matrix, GDA* performs better than GDA.

In Fig. 2, we also plot the reliability diagrams for an adversarially trained ResNet-18 on CIFAR-10, and we report the expected calibration error (ECE) [26]. We can observe that the model trained by PGD-AT is well-calibrated, at least on the seen attack PGD-10, which is consistent with previous observations [56, 65].

[1] https://github.com/val-iisc/GAMA-GAT

References

[1] Nilesh A Ahuja, Ibrahima Ndiour, Trushant Kalyanpur, and Omesh Tickoo. Probabilistic modeling of deep features for out-of-distribution and adversarial detection. *arXiv preprint arXiv:1909.11786*, 2019. 2

[2] Jean-Baptiste Alayrac, Jonathan Uesato, Po-Sen Huang, Alhussein Fawzi, Robert Stanforth, and Pushmeet Kohli. Are labels required for improving adversarial robustness? In *Advances in Neural Information Processing Systems (NeurIPS)*, pages 12192–12202, 2019. 2

[3] Maksym Andriushchenko and Nicolas Flammarion. Understanding and improving fast adversarial training. In *Advances in neural information processing systems (NeurIPS)*, 2020. 2

[4] Rushil Anirudh, Jayaraman J Thiagarajan, Bhavya Kailkhura, and Peer-Timo Bremer. Mimicgan: Robust projection onto image manifolds with corruption mimicking. *International Journal of Computer Vision (IJCV)*, pages 1–19, 2020. 2

[5] Qi-Zhi Cai, Chang Liu, and Dawn Song. Curriculum adversarial training. In *International Joint Conference on Artificial Intelligence (IJCAI)*, pages 3740–3747, 2018. 2

[6] Nicholas Carlini, Anish Athalye, Nicolas Papernot, Wieland Brendel, Jonas Rauber, Dimitris Tsipras, Ian Goodfellow, Aleksander Madry, and Alexey Kurakin. On evaluating adversarial robustness. *arXiv preprint arXiv:1902.06705*, 2019. 6

[7] Nicholas Carlini and David Wagner. Towards evaluating the robustness of neural networks. In *IEEE Symposium on Security and Privacy (S&P)*, 2017. 6

[8] Yair Carmon, Aditi Raghunathan, Ludwig Schmidt, Percy Liang, and John C Duchi. Unlabeled data improves adversarial robustness. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2019. 2

[9] Fabio Carrara, Rudy Becarelli, Roberto Caldelli, Fabrizio Falchi, and Giuseppe Amato. Adversarial examples detection in features distance spaces. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 2018. 2

[10] Jinghui Chen and Quanquan Gu. Rays: A ray searching method for hard-label adversarial attack. In *International Conference on Knowledge Discovery & Data Mining (KDD)*, 2020. 2

[11] Kejiang Chen, Yuefeng Chen, Hang Zhou, Xiaofeng Mao, Yuhong Li, Yuan He, Hui Xue, Weiming Zhang, and Nenghai Yu. Self-supervised adversarial training. In *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 2218–2222. IEEE, 2020. 2

[12] Tianlong Chen, Sijia Liu, Shiyu Chang, Yu Cheng, Lisa Amini, and Zhangyang Wang. Adversarial robustness: From self-supervised pre-training to fine-tuning. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 699–708, 2020. 2

[13] Gilad Cohen, Guillermo Sapiro, and Raja Giryes. Detecting adversarial samples using influence functions and nearest neighbors. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020. 2

[14] Jeremy M Cohen, Elan Rosenfeld, and J Zico Kolter. Certified adversarial robustness via randomized smoothing. In *International Conference on Machine Learning (ICML)*, 2019. 3

[15] Francesco Crecchi, Marco Melis, Angelo Sotgiu, Davide Bacciu, and Battista Biggio. Fader: Fast adversarial example rejection. *arXiv preprint arXiv:2010.09119*, 2020. 2

[16] Francesco Croce, Maksym Andriushchenko, Vikash Sehwag, Nicolas Flammarion, Mung Chiang, Prateek Mittal, and Matthias Hein. Robustbench: a standardized adversarial robustness benchmark. *arXiv preprint arXiv:2010.09670*, 2020. 2

[17] Francesco Croce and Matthias Hein. Reliable evaluation of adversarial robustness with an ensemble of diverse parameter-free attacks. In *International Conference on Machine Learning (ICML)*, 2020. 6

[18] Yinpeng Dong, Qi-An Fu, Xiao Yang, Tianyu Pang, Hang Su, Zihao Xiao, and Jun Zhu. Benchmarking adversarial robustness. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020. 2

[19] Abhimanyu Dubey, Laurens van der Maaten, Zeki Yalniz, Yixuan Li, and Dhruv Mahajan. Defense against adversarial images using web-scale nearest-neighbor search. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 8767–8776, 2019. 2

[20] Reuben Feinman, Ryan R Curtin, Saurabh Shintre, and Andrew B Gardner. Detecting adversarial samples from artifacts. *arXiv preprint arXiv:1703.00410*, 2017. 2, 4

[21] Yonatan Geifman and Ran El-Yaniv. Selectivenet: A deep neural network with an integrated reject option. In *International Conference on Machine Learning (ICML)*, 2019. 5

[22] Lovedeep Gondara. Detecting adversarial samples using density ratio estimates. *arXiv preprint arXiv:1705.02224*, 2017. 2

[23] Sven Gowal, Jonathan Uesato, Chongli Qin, Po-Sen Huang, Timothy Mann, and Pushmeet Kohli. An alternative surrogate loss for pgd-based adversarial testing. *arXiv preprint arXiv:1910.09338*, 2019. 6

[24] Jean-Bastien Grill, Florian Strub, Florent Altché, Corentin Tallec, Pierre H Richemond, Elena Buchatskaya, Carl Doersch, Bernardo Avila Pires, Zhaohan Daniel Guo, Mohammad Gheshlaghi Azar, et al. Bootstrap your own latent: A new approach to self-supervised learning. In *Advances in neural information processing systems (NeurIPS)*, 2020. 2

[25] Kathrin Grosse, Praveen Manoharan, Nicolas Papernot, Michael Backes, and Patrick McDaniel. On the (statistical) detection of adversarial examples. *arXiv preprint arXiv:1702.06280*, 2017. 2

[26] Chuan Guo, Geoff Pleiss, Yu Sun, and Kilian Q Weinberger. On calibration of modern neural networks. In *International Conference on Machine Learning (ICML)*, 2017. 3, 6

[27] Dan Hendrycks, Kimin Lee, and Mantas Mazeika. Using pre-training can improve model robustness and uncertainty. In *International Conference on Machine Learning (ICML)*, 2019. 2

[28] Haoming Jiang, Zhehui Chen, Yuyang Shi, Bo Dai, and Tuo Zhao. Learning to defense by learning to attack. *arXiv preprint arXiv:1811.01213*, 2018. 2

[29] Kimin Lee, Kibok Lee, Honglak Lee, and Jinwoo Shin. A simple unified framework for detecting out-of-distribution samples and adversarial attacks. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2018. 2

[30] Bai Li, Shiqi Wang, Suman Jana, and Lawrence Carin. Towards understanding fast adversarial training. *arXiv preprint arXiv:2006.03089*, 2020. 2

[31] Pengcheng Li, Jinfeng Yi, Bowen Zhou, and Lijun Zhang. Improving the robustness of deep neural networks via adversarial training with triplet loss. In *International Joint Conference on Artificial Intelligence (IJCAI)*, 2019. 2

[32] Guanxiong Liu, Issa Khalil, and Abdallah Khreishah. Using single-step adversarial training to defend iterative adversarial examples. *arXiv preprint arXiv:2002.09632*, 2020. 2

[33] Weitang Liu, Xiaoyun Wang, John Owens, and Sharon Yixuan Li. Energy-based out-of-distribution detection. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2020. 5

[34] Xuanqing Liu, Yao Li, Chongruo Wu, and Cho-Jui Hsieh. Adv-bnn: Improved adversarial defense through robust bayesian neural network. In *International Conference on Learning Representations (ICLR)*, 2019. 2, 3

[35] Jiajun Lu, Theerasit Issaranon, and David Forsyth. Safetynet: Detecting and rejecting adversarial examples robustly. In *International Conference on Computer Vision (ICCV)*, pages 446–454, 2017. 2

[36] Chengcheng Ma, Baoyuan Wu, Shibiao Xu, Yanbo Fan, Yong Zhang, Xiaopeng Zhang, and Zhifeng Li. Effective and robust detection of adversarial examples via benford-fourier coefficients. *arXiv preprint arXiv:2005.05552*, 2020. 2

[37] Shiqing Ma and Yingqi Liu. Nic: Detecting adversarial samples with neural network invariant checking. In *Proceedings of the 26th Network and Distributed System Security Symposium (NDSS 2019)*, 2019. 2

[38] 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. In *International Conference on Learning Representations (ICLR)*, 2018. 2, 5

[39] 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 (ICLR)*, 2018. 6

[40] Pratyush Maini, Eric Wong, and Zico Kolter. Adversarial robustness against the union of multiple perturbation models. In *International Conference on Machine Learning (ICML)*, pages 6640–6650. PMLR, 2020. 2

[41] Chengzhi Mao, Ziyuan Zhong, Junfeng Yang, Carl Vondrick, and Baishakhi Ray. Metric learning for adversarial robustness. In *Advances in Neural Information Processing Systems (NeurIPS)*, pages 478–489, 2019. 2

[42] Jan Hendrik Metzen, Tim Genewein, Volker Fischer, and Bastian Bischoff. On detecting adversarial perturbations. In *International Conference on Learning Representations (ICLR)*, 2017. 2

[43] Muzammal Naseer, Salman Khan, Munawar Hayat, Fahad Shahbaz Khan, and Fatih Porikli. A self-supervised approach for adversarial robustness. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 262–271, 2020. 2

[44] Tianyu Pang, Chao Du, Yinpeng Dong, and Jun Zhu. Towards robust detection of adversarial examples. In *Advances in Neural Information Processing Systems (NeurIPS)*, pages 4579–4589, 2018. 2

[45] Tianyu Pang, Kun Xu, Chao Du, Ning Chen, and Jun Zhu. Improving adversarial robustness via promoting ensemble diversity. In *International Conference on Machine Learning (ICML)*, 2019. 2

[46] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. Pytorch: An imperative style, high-performance deep learning library. In *Advances in Neural Information Processing Systems (NeurIPS)*, pages 8024–8035, 2019. 4

[47] Julien Perolat, Mateusz Malinowski, Bilal Piot, and Olivier Pietquin. Playing the game of universal adversarial perturbations. *arXiv preprint arXiv:1809.07802*, 2018. 2

[48] Ambrish Rawat, Martin Wistuba, and Maria-Irina Nicolae. Adversarial phenomenon in the eyes of bayesian deep learning. *arXiv preprint arXiv:1711.08244*, 2017. 3

[49] Kevin Roth, Yannic Kilcher, and Thomas Hofmann. The odds are odd: A statistical test for detecting adversarial examples. In *International Conference on Machine Learning (ICML)*, 2019. 2

[50] Pouya Samangouei, Maya Kabkab, and Rama Chellappa. Defense-gan: Protecting classifiers against adversarial attacks using generative models. In *International Conference on Learning Representations (ICLR)*, 2018. 2

[51] Ali Shafahi, Mahyar Najibi, Amin Ghiasi, Zheng Xu, John Dickerson, Christoph Studer, Larry S Davis, Gavin Taylor, and Tom Goldstein. Adversarial training for free! In *Advances in Neural Information Processing Systems (NeurIPS)*, 2019. 2

[52] Ali Shafahi, Mahyar Najibi, Zheng Xu, John P Dickerson, Larry S Davis, and Tom Goldstein. Universal adversarial training. In *AAAI Conference on Artificial Intelligence (AAAI)*, pages 5636–5643, 2020. 2

[53] Fatemeh Sheikholeslami, Swayambhoo Jain, and Georgios B Giannakis. Minimum uncertainty based detection of adversaries in deep neural networks. *arXiv preprint arXiv:1904.02841*, 2019. 2

[54] Lewis Smith and Yarin Gal. Understanding measures of uncertainty for adversarial example detection. In *Conference on Uncertainty in Artificial Intelligence (UAI)*, 2018. 2

[55] Philip Sperl, Ching-Yu Kao, Peng Chen, and Konstantin Böttinger. Dla: Dense-layer-analysis for adversarial example detection. In *IEEE European Symposium on Security and Privacy (EuroS&P)*, 2020. 2

[56] David Stutz, Matthias Hein, and Bernt Schiele. Confidence-calibrated adversarial training: General-

izing to unseen attacks. In *International Conference on Machine Learning (ICML)*, 2020. 6

[57] Guanhong Tao, Shiqing Ma, Yingqi Liu, and Xiangyu Zhang. Attacks meet interpretability: Attribute-steered detection of adversarial samples. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2018. 2

[58] Florian Tramèr and Dan Boneh. Adversarial training and robustness for multiple perturbations. In *Advances in Neural Information Processing Systems (NeurIPS)*, pages 5858–5868, 2019. 2

[59] Florian Tramèr, Alexey Kurakin, Nicolas Papernot, Dan Boneh, and Patrick McDaniel. Ensemble adversarial training: Attacks and defenses. In *International Conference on Learning Representations (ICLR)*, 2018. 2

[60] Laurens Van der Maaten and Geoffrey Hinton. Visualizing data using t-sne. *Journal of machine learning research (JMLD)*, 9(11), 2008. 6

[61] Huaxia Wang and Chun-Nam Yu. A direct approach to robust deep learning using adversarial networks. In *International Conference on Learning Representations (ICLR)*, 2019. 2

[62] Yisen Wang, Xingjun Ma, James Bailey, Jinfeng Yi, Bowen Zhou, and Quanquan Gu. On the convergence and robustness of adversarial training. In *International Conference on Machine Learning (ICML)*, pages 6586–6595, 2019. 2

[63] Yisen Wang, Difan Zou, Jinfeng Yi, James Bailey, Xingjun Ma, and Quanquan Gu. Improving adversarial robustness requires revisiting misclassified examples. In *International Conference on Learning Representations (ICLR)*, 2019. 2

[64] Eric Wong, Leslie Rice, and J. Zico Kolter. Fast is better than free: Revisiting adversarial training. In *International Conference on Learning Representations (ICLR)*, 2020. 2

[65] Xi Wu, Uyeong Jang, Jiefeng Chen, Lingjiao Chen, and Somesh Jha. Reinforcing adversarial robustness using model confidence induced by adversarial training. In *International Conference on Machine Learning (ICML)*, pages 5334–5342. PMLR, 2018. 6

[66] Weilin Xu, David Evans, and Yanjun Qi. Feature squeezing: Detecting adversarial examples in deep neural networks. *arXiv preprint arXiv:1704.01155*, 2017. 2

[67] Huanrui Yang, Jingyang Zhang, Hongliang Dong, Nathan Inkawhich, Andrew Gardner, Andrew Touchet, Wesley Wilkes, Heath Berry, and Hai Li. Dverge: Diversifying vulnerabilities for enhanced robust generation of ensembles. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2020. 2

[68] Puyudi Yang, Jianbo Chen, Cho-Jui Hsieh, Jane-Ling Wang, and Michael I Jordan. Ml-loo: Detecting adversarial examples with feature attribution. In *Thirty-First AAAI Conference on Artificial Intelligence (AAAI)*, 2020. 2

[69] Sergey Zagoruyko and Nikos Komodakis. Wide residual networks. In *The British Machine Vision Conference (BMVC)*, 2016. 6

[70] Chiliang Zhang, Zuochang Ye, Yan Wang, and Zhimou Yang. Detecting adversarial perturbations with saliency. In *2018 IEEE 3rd International Conference on Signal and Image Processing (ICSIP)*, pages 271–275. IEEE, 2018. 2

[71] Dinghuai Zhang, Tianyuan Zhang, Yiping Lu, Zhanxing Zhu, and Bin Dong. You only propagate once: Accelerating adversarial training via maximal principle. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2019. 2

[72] Jingfeng Zhang, Xilie Xu, Bo Han, Gang Niu, Lizhen Cui, Masashi Sugiyama, and Mohan Kankanhalli. Attacks which do not kill training make adversarial learning stronger. In *International Conference on Machine Learning (ICML)*, 2020. 2

[73] Jingfeng Zhang, Jianing Zhu, Gang Niu, Bo Han, Masashi Sugiyama, and Mohan Kankanhalli. Geometry-aware instance-reweighted adversarial training. In *International Conference on Learning Representations (ICLR)*, 2021. 2

[74] Chenxiao Zhao, P Thomas Fletcher, Mixue Yu, Yaxin Peng, Guixu Zhang, and Chaomin Shen. The adversarial attack and detection under the fisher information metric. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 5869–5876, 2019. 2