CONFIDENCE-CALIBRATED ADVERSARIAL TRAINING: TOWARDS ROBUST MODELS GENERALIZING BEYOND THE ATTACK USED DURING TRAINING

Anonymous authors
Paper under double-blind review

ABSTRACT

Adversarial training is the standard to train models robust against adversarial examples. However, especially for complex datasets, adversarial training incurs a significant loss in accuracy and is known to generalize poorly to stronger attacks, e.g., larger perturbations or other threat models. In this paper, we introduce **confidence-calibrated adversarial training (CCAT)** where the key idea is to enforce that the confidence on adversarial examples decays with their distance to the attacked examples. We show that CCAT preserves better the accuracy of normal training while robustness against adversarial examples is achieved via confidence thresholding. Most importantly, in strong contrast to adversarial training, the robustness of CCAT generalizes to larger perturbations and other threat models, not encountered during training. We also discuss our extensive work to design strong adaptive attacks against CCAT and standard adversarial training which is of independent interest. We present experimental results on MNIST, SVHN and Cifar10.

1 INTRODUCTION

Deep neural networks have shown tremendous improvements in various learning tasks including applications in computer vision, natural language processing or text processing. However, the discovery of adversarial examples, i.e., nearly imperceptibly perturbed inputs that cause mis-classification, has revealed severe security threats, as demonstrated by attacking popular computer vision services such as Google Cloud Vision (Ilyas et al., 2018a) or Clarifai (Liu et al., 2016; Bhagoji et al., 2017). As the number of safety- and privacy-critical applications is increasing, e.g., autonomous driving or medical imaging, this problem becomes even more important.

While defenses promising certified robustness against adversarial examples have received considerable attention (Hein & Andriushchenko, 2017; Wong & Kolter, 2018; Weng et al., 2018; Mirman et al., 2018; Gowal et al., 2018), such approaches are limited to small networks and often lead to significantly increased test error. In practice, adversarial training, i.e., training on adversarial examples as proposed by Madry et al. (2018), can be regarded as the state-of-the-art and, to the best of our knowledge, has not been broken so far. However, adversarial training is known to increase test error significantly. Only on simple datasets such as MNIST (LeCun et al., 1998), adversarial training is able to preserve accuracy. This observation is typically described as a trade-off between robustness and accuracy (Schmidt et al., 2018; Stutz et al., 2019). Furthermore, the success of adversarial training strongly depends on the attack used during training. The achieved robustness does not translate to novel attacks, e.g., allowing larger adversarial perturbations at test time or different threat models (Song et al., 2018; Sharma & Chen, 2017).

Contributions: We aim to address both problems: the robustness-accuracy trade-off and the poor generalization to other or stronger attacks. To this end, we introduce **confidence-calibrated adversarial training (CCAT)** based on the idea that the confidence in an adversarial example should decrease as a function of the distance to the attacked data point. Specifically, we bias the network to predict for the adversarial example a convex combination of the uniform distribution over the labels and the label of the attacked point, which tends to become uniform as the distance to the attacked point increases. We show that this novel procedure of adversarial training leads to better "generalization" of the robustness to both stronger attacks and other threat models. This is in contrast to standard adversarial training, which does not in general generalize to other attacks. The main reason is that adversarial training does not tell the network how to extrapolate beyond the specific perturbations seen during training, which we overcome with CCAT. We show that our approach allows to detect adversarial examples based on their confidence while better preserving the accuracy of normal training, thereby improving also upon the robustness-accuracy trade-off of regular adversarial training. We will make our code and results publicly available.

2 Related Work

Adversarial examples are roughly divided into white-box attacks, i.e., with access to the models, its weights and gradients, e.g. (Goodfellow et al., 2014; Madry et al., 2017; Carlini & Wagner, 2017), and black-box attacks, i.e., only with access to the output of the model, e.g. (Chen et al., 2017; Brendel & Bethge, 2017; Su et al., 2017; Ilyas et al., 2018b; Sarkar et al., 2017; Narodytska & Kasiviswanathan, 2017). White-box attacks utilizing projected gradient ascent to maximize the training loss or surrogate objectives, e.g., (Madry et al., 2017; Carlini & Wagner, 2017), have become state-of-the-art. In contrast, we directly maximize the confidence in any class different from the true class, similar to (Hein et al., 2019), to attack our proposed training procedure. Additionally, momentum (Dong et al., 2018), backtracking and alternative initializations than used in the literature are required for successful attacks against our models.

Many defenses against adversarial attacks have been proposed, e.g. (Yuan et al., 2017; Akhtar & Mian, 2018; Biggio & Roli, 2018), of which some have been shown to be ineffective, e.g., in (Athalye et al., 2018; Athalye & Carlini, 2018). Other methods aiming at certified robustness (Hein & Andriushchenko, 2017; Wong & Kolter, 2018; Weng et al., 2018; Mirman et al., 2018), adversarial training is the standard to achieve robust models. While adversarial training was proposed in different variants (Zantedeschi et al., 2017; Miyato et al., 2016; Huang et al., 2015; Shaham et al., 2018; Sinha et al., 2018; Lee et al., 2017; Madry et al., 2017), the formulation by Madry et al. (2017) received considerable attention and has been extended in various ways, e.g., to universal adversarial examples (Shafahi et al., 2018; Pérolat et al., 2018), using a curriculum learning scheme (Cai et al., 2018) or ensembles of networks (Tramèr et al., 2017; Grefenstette et al., 2018). Our CCAT differs from regular adversarial training in the imposed distribution over the labels enforced during training on adversarial examples and by the attack objective. These seemingly simple modifications lead to a classifier which can extrapolate its robustness to other attack models.

3 Confidence Calibration of Adversarial Examples

Adversarial training, specifically the robust optimization formulation proposed by Madry et al. (2018), has become standard for obtaining neural networks robust against adversarial examples. In fact, adversarial training is among the few approaches that have not been shown to be ineffective. However, it is known to reduce the accuracy significantly, especially on challenging tasks. Similarly, the robustness obtained through adversarial training is argued to generalize poorly to stronger attacks and other threat models. Our goal is to overcome these problems with our proposed confidence-calibrated adversarial training (CCAT).

We consider a classifier $f : \mathbb{R}^d \rightarrow \mathbb{R}^K$ where K is the number of classes and f_k denotes the confidence for class k. We assume that the cross-entropy loss is used during training, even though our approach can be used with other losses as well. Given $x \in \mathbb{R}^d$ classified correctly as $y = \text{argmax}_k f_k(x)$, an adversarial perturbation $x + \delta$ is defined as a "small" change δ such that $\text{argmax}_k f_k(x + \delta) \neq y$, i.e., the classifier changes its decision. The strength of the change δ is measured by some l_p-norm with $p \in \{1, 2, \infty\}$. $p = \infty$ is a popular choice in the literature as this leads to the smallest perturbation per feature/pixel.

3.1 Robust Loss Formulation of Adversarial Training

The successful and theoretically elegant robust optimization formulation by Madry et al. (2018) is given as the following min-max problem:

$$\min_w \mathbb{E}\left[\max_{\|\delta\|_\infty \leq \epsilon} \mathcal{L}(f(x + \delta; w), y)\right] \tag{1}$$

with w being the classifier's parameters and $\mathcal{L}$ being the cross-entropy loss. During mini-batch training the inner maximization problem, i.e.,

$$\max_{\|\delta\|_\infty \le \epsilon} \mathcal{L}(f(x+\delta; w), y), \tag{2}$$

is approximately solved. In addition to the l_∞-constraint, a box constraint, i.e., $\tilde{x}_i = (x+\delta)_i \in [0, 1]$, is enforced for images. Note that for the cross-entropy loss, Eq. (2) is equivalent to finding the point $x + \delta$ with *minimal* confidence in the true class y. Finally, for neural networks, we note that Eq. (2) is a non-convex optimization problem. In (Madry et al., 2018) the problem is tackled using projected gradient descent (PGD), which is typically initialized using a random δ with $\|\delta\|_\infty \le \epsilon$. At test time one uses the best out of several random restarts of Eq. (2) to assess robustness.

In contrast to adversarial training as proposed in (Madry et al., 2018), which computes adversarial examples for the *full* mini-batch, others compute adversarial examples only for 50% of the mini-batch, e.g. (Szegedy et al., 2013). Compared to Eq. (1), this approach effectively minimizes

$$\min_w \Big(\mathbb{E}\Big[\max_{\|\delta\|_\infty \le \epsilon} \mathcal{L}(f(x+\delta; w), y)\Big] + \mathbb{E}\big[\mathcal{L}(f(x; w), Y)\big]\Big). \tag{3}$$

This improves test accuracy on clean examples compared to Eq. (1) but typically leads to worse robustness. Intuitively, this variant already optimizes the mentioned robustness-accuracy trade-off.

There are two problems of adversarial training in Eq. (1). First, the ϵ-ball around training examples might include examples from other classes. Then, Eq. (2) will focus on these regions such that adversarial training for these examples gets "stuck". This case is illustrated in our theoretical toy dataset in Sec. 3.3. Here, both 100% and 50% adversarial training, cf. Eq. (1) and (3), are not able to find the Bayes optimal classifier in a fully deterministic problem, i.e., zero Bayes error. This might contribute to the observed drop in accuracy for adv. training on datasets such as Cifar10 (Krizhevsky, 2009). Second and most importantly, adversarial training as in Eq. (1) does not give any guidance to the classifier how to extrapolate the classifier beyond the used ϵ-ball during training. Even worse, it enforces high confidence predictions everywhere inside the ϵ-ball but clearly one cannot extrapolate high-confidence predictions to increasingly larger neighborhoods. Thus, it is not surprising that adversarial examples can often be found right beyond the ϵ-ball, i.e., there is no generalization of robustness to stronger attacks of the same type or other threat models, e.g., other p-norm balls.

3.2 Confidence-Calibrated Adversarial Training

We address both problems of adversarial training: the tension between accuracy and robustness and the poor generalization to larger ϵ-balls and other threat models. The required modifications as outlined in Alg. 1 are small but effective. During training, instead of searching for an adversarial

Algorithm 1 Pseudo-code of confidence-calibrated adversarial training (CCAT). The main changes compared to regular adversarial training as, e.g., described in (Madry et al., 2018) or (Szegedy et al., 2013), are in the attack (line 4) and the probability distribution over the classes (line 6,7), which becomes more uniform as distance $\|\delta\|_\infty$ increases.

```
1:  while true do
2:      choose random batch $(x_1, y_1), \ldots, (x_B, y_B)$.
3:      for $b = 1, \ldots, B/2$ do
4:          {maximize confidence in other classes than true one of adversarial example $\tilde{x}_b$, Eq. (2):}
5:          $\delta_b := \text{argmax}_{\|\delta\|_\infty \le \epsilon} \max_{k \ne y_b} f_k(x_b + \delta)$
6:          $\tilde{x}_b := x_b + \delta_b$
7:          {probability over classes of $\tilde{x}_b$ becomes more uniform as $\|\delta_b\|_\infty$ increases:}
8:          $\lambda := e^{-\rho\|\delta_b\|_\infty}$ or $\lambda := (1 - \min(1, \|\delta\|_\infty/\epsilon))^\rho$
9:          {$\tilde{y}_b$ is convex combination of one hot and uniform distribution over the classes:}
10:         $\tilde{y_b} := \lambda \, \text{one\_hot}(y_b) + \frac{(1-\lambda)}{K}\mathbb{1}$
11:     end for
12:     update parameters using $\sum_{b=1}^{B/2} \mathcal{L}(f(\tilde{x}_b), \tilde{y}_b) + \sum_{b=B/2}^{B} \mathcal{L}(f(x_b), y_b)$
13: end while
```

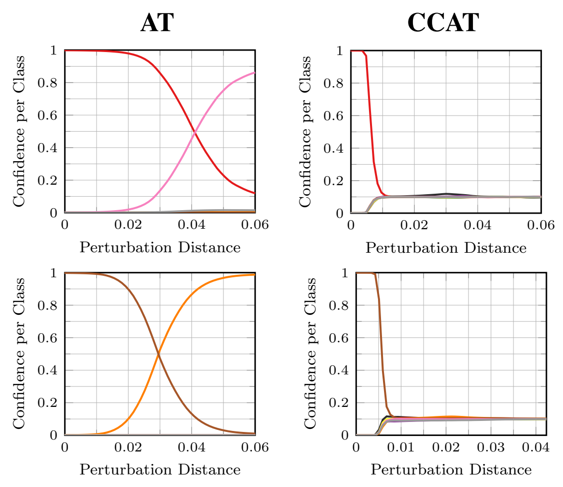

Figure 1: **Illustration of Confidence Calibration.** For adversarial training (AT) and our confidence-calibrated adversarial training (CCAT) with $\rho_{\text{pow}} = 10$ using the power transition in Eq. (6), both trained with $\epsilon = 0.03$ on SVHN, we show the probabilities for all ten classes along adversarial directions. Adversarial examples were computed using our L_∞-PGD-Conf attack. The robustness of AT does not generalize as directly after or in the ϵ-ball the classifier attains high confidence in a different class, whereas CCAT predicts close to uniform confidence after some transition phase and thus adversarial samples can be easily distinguished from test examples due to their low confidence.

example $x + \delta$ that minimizes the confidence in the true label y, as in Eq. (2), we search for an adversarial example that maximizes the confidence in an arbitrary other label $k \neq y$:

$$\max_{\|\delta\|_\infty \leq} \max_{k \neq y} f_k(x + \delta; w) \tag{4}$$

This is motivated by our defense strategy to detect adversarial examples based on their (low) confidence. Thus, a natural adaptive attack against this strategy is maximizing the target confidence, similar to (Goodfellow et al., 2019). During training, CCAT biases the classifier by feeding back adversarial examples into the training process with label distribution shifted to the uniform distribution on adversarial examples, given as:

$$\hat{p}(k) = \lambda p_y(k) + (1 - \lambda)u(k), \quad k = 1, \dots, K. \tag{5}$$

Here, $p_y(k)$ is the original "one-hot" distribution, i.e., $p_y(k) = 1$ iff $k = y$ and $p_y(k) = 0$ otherwise, and $u(k) = \frac{1}{K}$ is the uniform distribution. Thus, we enforce a convex combination of the original label distribution and the uniform distribution which is controlled by the parameter λ. We choose λ to decrease with the distance $\|\delta\|_\infty$ of the adversarial example to the attacked example x. We consider two similar variants of transitions:

$$\begin{aligned} \lambda =& e^{-\rho\|\delta\|_\infty} && \text{("exponential transition" (exp)} \\ \lambda =& (1 - \min(1, \|\delta\|_\infty/\epsilon))^\rho && \text{("power transition" (pow))} \end{aligned} \tag{6}$$

This ensures that for $\delta = 0$ we impose the original (one-hot) label. For growing δ, however, the influence of the original label decays proportional to $\|\delta\|_\infty$. The speed of decay is controlled by the parameter ρ. For the exponential transition, we always have a bias towards the true label as even for large ρ, λ will be non-zero. In case of the power transition, $\lambda = 0$ for $\|\delta\|_\infty \geq \epsilon$, meaning a pure uniform distribution is enforced. We call this procedure **confidence-calibrated adversarial training (CCAT)**. Both transitions used in CCAT guide the classifier to decrease its confidence to uniform when leaving the "data manifold" in an adversarial way – we note that adversarial examples leave the data-manifold (Stutz et al., 2019). In this way the classifier can generalize its robustness to stronger attacks and other threat models as it predicts simply uniform confidence there, see Fig. 1. It is important to note that in CCAT in Alg. 1 we train on 50% clean and 50% adversarial examples in each mini-batch. Training only on adversarial examples will not work as we loose signal where the true data manifold lies.

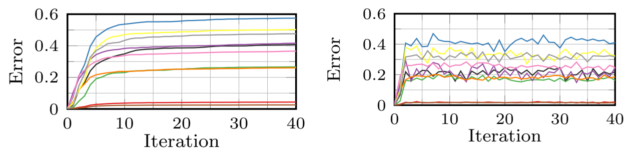

Figure 2: **Momentum and backtracking.** Our PGD-Conf with 40 iterations with momentum and backtracking (left) and without both (right). We plot the objective of Eq. (4) over iterations for 10 samples (different colors).

RErr @99%TPR in % on **SVHN** (L_∞ attack with $\epsilon = 0.03$)							
Optimization	momentum+backtrack					mom	
Initialization	zero				rand	zero	zero
Iterations T	40	200	2000	4000	4000	300	300
AT	38.4	46.2	49.9	50.1	51.8	38.1	30.8
AT Conf	27.4	40.5	46.9	47.3	48.1	28.5	23.8
CCAT, $\rho_{\text{pow}} = 10$	4.0	5.0	22.8	23.3	5.2	2.6	2.6

Table 1: **Attack ablation study on SVHN.** Comparison of our adapted L_∞ PGD-Conf attack with $\epsilon = 0.03$ on the test set for different number of iterations T and configurations of momentum, backtracking and initialization. As backtracking needs an additional forward pass per iteration, we compare $T = 200$ with backtracking to $T = 300$ without. Attacks on AT succeed within a few iterations, but are more difficult against CCAT and require initialization at zero.

3.3 Confidence-Calibrated Adversarial Training Yields Accurate Models

The following Proposition analyzes 100% adversarial training, cf., Eq. (1) as proposed by Madry et al. (2018) and its 50% variant, cf. Eq. (3), and our confidence-calibrated variant:

Proposition 1. *We consider a classification problem with two points $x = 0$ and $x = \epsilon$ in $\mathbb{R}$ with deterministic labels, that is $p(y = 2|x = 0) = 1$ and $p(y = 1|x = \epsilon) = 1$ and the problem is fully determined by the probability $p_0 = p(x = 0)$ as $p(x = \epsilon) = 1 - p_0$. The Bayes error of this classification problem is zero. The Bayes optimal classifier of*

- 100% *adversarial training yields an error of* $\min\{p_0, 1 - p_0\}$.
- *adversarial training with* 50% *adversarial and* 50% *clean examples yields an error of* $\min\{p_0, 1 - p_0\}$.
- *Our confidence-calibrated adversarial training with* 50% *clean and* 50% *adversarial examples yields* zero error *if* $\lambda < \min\{\frac{p_0}{1-p_0}, \frac{1-p_0}{p_0}\}$.

This proposition shows a clear advantage of our confidence-calibrated adversarial training over regular adversarial training. It reconfirms that there is indeed a tension between accuracy and robustness when using adversarial training both in the 100% and the 50% variants as it has recently been discussed (Tsipras et al., 2018; Stutz et al., 2019). However, our confidence-calibrated adversarial training can resolve this if λ and thus ρ in Eq. (6) is chosen appropriately.

4 Experiments

We evaluate our CCAT based on the ability to reject adversarial examples by their confidence and generalize robustness to larger ϵ-balls and other threat models. Thus, we use a two-stage approach: first, we decide whether a given (potentially adversarial) example is rejected or classified; second, we evaluate the robustness and accuracy on the non-rejected examples. We present experiments on MNIST, (LeCun et al., 1998) SVHN (Netzer et al., 2011) and Cifar10 (Krizhevsky, 2009).[1]

Attacks: We follow (Madry et al., 2018) and use projected gradient descent (PGD) to minimize the negatives of Eq. (2) and (4); we denote them as PGD-CE and PGD-Conf. The perturbation δ is initialized uniformly over direction and distance; for PGD-Conf, we additionally use $\delta = 0$ as initialization. Different from (Madry et al., 2018), we run exactly T iterations (no early stopping) and take the perturbation corresponding to the best objective of the T iterations. In addition to momentum, as in (Dong et al., 2018), we propose to use an adaptive learning rate in combination with a backtracking scheme to improve the attacks: after each iteration, the computed update is only applied if it improves the objective; otherwise the learning rate is reduced. For evaluation, we use $T = 2000$ iterations, 10 random retries and learning rate of 0.001 for PGD-Conf and $T = 200$ with 50 random retries and learning rate 0.05 for PGD-CE. As black-box attacks, we additionally use random sampling, the attack by Ilyas et al. (2018a), adapted with momentum and backtracking optimizing Eq. (4) for $T = 2000$ iterations with 10 attempts, a variant of (Narodytska & Kasiviswanathan,

[1]Additional details, precise descriptions of the white- and black-box attacks used, and more experimental results can be found in Appendix B.

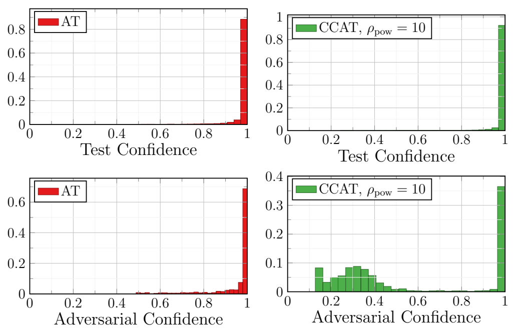

Figure 3: **Confidence histograms on SVHN.** For AT (left) and CCAT (right) with $\rho_{\text{pow}} = 10$ (right), we show confidence histograms corresponding to *correctly classified* test examples (top) and *successful* adversarial examples (bottom). We consider the worst-case adversarial examples across all tested L_∞ attacks for $\epsilon = 0.03$.

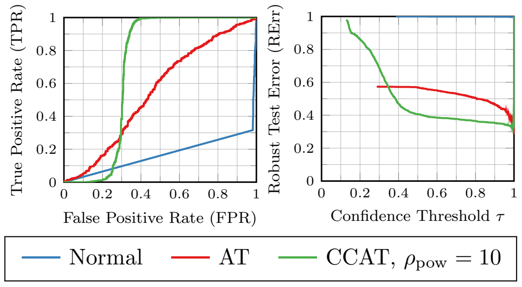

Figure 4: **ROC and RErr curves on SVHN.** Left: ROC curves, i.e., FPR against TPR when distinguishing *correctly classified* test examples from *successful* adversarial examples by confidence. Right: RErr against confidence threshold τ. For evaluation, we choose τ in order to obtain 99%TPR. As described in the text, RErr subsumes both Err and FPR. Curves based on worst-case examples across all tested L_∞ attacks.

2017) with $T = 2000$ iterations and the "cube" attack Andriushchenko (2019) with $T = 5000$ iterations. The black-box attacks ensure that our defense avoids, e.g., gradient masking as described in (Athalye et al., 2018). In addition to L_∞ attacks, we also consider PGD-CE, PGD-Conf and the cube attack for L_2-attacks. For each model, we attack the first 1000 test examples and evaluate on the per-example *worst-case* adversarial examples, i.e., the adversarial examples with highest confidence per example but across all attacks and attempts.

Training: We use ResNet-20 (He et al., 2016) on all datasets, implemented in PyTorch (Paszke et al., 2017). The networks are initialized using (He et al., 2015) and trained using stochastic gradient descent with batch size of 100 for 100 or 200 epochs (MNIST and SVHN/Cifar10, respectively). We use $T = 40$ iterations, $\epsilon = 0.3$ on MNIST and $\epsilon = 0.03$ on SVHN and Cifar10 for the attacks during training – images are normalized to $[0, 1]$. For CCAT we initialize perturbations uniformly or at zero.

Evaluation Metrics: We evaluate our proposed approach in terms of detection of adversarial examples: *successful* adversarial examples are considered negatives and *correctly classified* test examples are considered positives. We report the area under the ROC curve, i.e., ROC AUC. For fair comparison with regular adversarial training (AT), we report both test error (Err) and robust test error (RErr) and also extend these metrics to our detection setting; specifically, we report both Err and RErr for a confidence threshold τ resulting in a true positive rate (TPR) of 99%, i.e., the network is allowed to reject only up to 1% of correctly classified test examples. Then, RErr(τ) is calculated as:

$$\text{RErr}(\tau) = \frac{\sum_{n=1}^{N} \mathbb{1}_{f(x_n)\neq y_n} \mathbb{1}_{c(x_n)\geq\tau} + \sum_{n=1}^{N} \mathbb{1}_{f(x_n)=y_n} \mathbb{1}_{f(\tilde{x}_n)\neq y_n} \mathbb{1}_{c(\tilde{x}_n)\geq\tau}}{\sum_{n=1}^{N} \mathbb{1}_{c(x_n)\geq\tau} + \sum_{n=1}^{N} \mathbb{1}_{c(x_n)<\tau} \mathbb{1}_{c(\tilde{x}_n)\geq\tau} \mathbb{1}_{f(x_n)=y_n} \mathbb{1}_{f(\tilde{x}_n)\neq y_n}}, \tag{7}$$

Here, τ is the confidence-threshold fixed on the held-out last 1000 test examples, $\{(x_n, y_n)\}_{n=1}^N$ are test examples, $c(x_n)$ denotes the classifier's confidence on x_n, and $\tilde{x}_n$ are adversarial examples. The enumerator counts the number of incorrectly classified test examples x_n with $c(x_n) \geq \tau$ (first term) and the number of *successful* adversarial examples $\tilde{x}_n$ on *correctly classified* test examples with $c(\tilde{x}_n) \geq \tau$ (second term). The denominator counts test examples x_n with $c(x_n) \geq \tau$ (first term) and the number of *successful* adversarial examples $\tilde{x}_n$ with $c(\tilde{x}_n) \geq \tau$ but where the corresponding test example x_n has $c(x_n) < \tau$ (second term). The latter takes care of the special case where adversarial examples have higher confidence than their corresponding test examples, which is encouraged by the objective of our PGD-Conf attack, see Eq. (4). In total this yields a correct fraction within $[0, 1]$. For $\tau = 0$, Eq. (7) reduces to the "regular" RErr. The confidence-thresholded Err(τ) corresponds to taking only the first terms in both enumerator and denominator. We also note that RErr(τ) naturally subsumes the false positive rate (FPR). RErr is always computed on the 1000 attacked test examples; Err is computed on all test examples (without the held-out part for determining τ@99%TPR).

MNIST (L_∞ attack with $\epsilon = 0.3$ during training)							
		τ=0			τ@99%TPR		
Attack	Training	Err in %	RErr in %	ROC AUC	Err in %	RErr in %	τ
$L_\infty,\ \epsilon = 0.3$	AdvTrain	**0.50**	**7.20**	0.97	**0.00**	**1.00**	1.00
	CCAT	**0.50**	100.00	**0.99**	0.10	7.70	0.99
$L_\infty,\ \epsilon = 0.4$	AT		100.00	0.20		100.00	
	CCAT		100.00	**0.94**		**40.00**	
$L_2,\ \epsilon = 3$	AT		98.80	0.73		81.30	
	CCAT		**82.60**	**1.00**		**1.40**	
SVHN (L_∞ attack with $\epsilon = 0.03$ during training)							
$L_\infty,\ \epsilon = 0.03$	AT	3.40	57.30	0.55	2.50	55.60	0.56
	CCAT	**2.90**	**97.80**	**0.70**	**2.10**	**38.50**	0.60
$L_\infty,\ \epsilon = 0.06$	AT		**89.00**	0.32		88.30	
	CCAT		99.80	**0.70**		**46.00**	
$L_2,\ \epsilon = 1$	AT		92.40	0.26		92.00	
	CCAT		**81.80**	**0.91**		**18.50**	
CIFAR10 (L_∞ attack with $\epsilon = 0.03$ during training)							
$L_\infty,\ \epsilon = 0.03$	AT	16.60	**62.70**	**0.64**	15.10	**62.30**	0.35
	CCAT	**10.10**	96.70	0.60	**8.70**	67.90	0.40
$L_\infty,\ \epsilon = 0.06$	AT		**93.70**	0.35		93.60	
	CCAT		99.20	**0.43**		**91.50**	
$L_2,\ \epsilon = 1$	AT		**74.40**	0.59		73.90	
	CCAT		81.80	**0.77**		**46.20**	

Table 2: **Main results on MNIST, SVHN and Cifar10.** Comparison of AT and CCAT on MNIST (top), SVHN (middle) and Cifar10 (bottom). We report worst-case results across all tested attacks, for L_∞ and L_2 attacks; the used ϵ values are reported in the left most column. During training, L_∞ attacks with $\epsilon = 0.3$ on MNIST and $\epsilon = 0.03$ on SVHN/Cifar10 were used. In all cases we report "regular" Err and RErr, their confidence-thresholded variants for τ@99%TPR as well as ROC AUC.

4.1 Ablation Study

Momentum and Backtracking: Fig. 2 illustrates the advantage of momentum and the proposed backtracking scheme for PGD-Conf with $T = 40$ iterations on 10 test examples of SVHN. As shown in Fig. 2 and Tab. 1, better objective values can be achieved within fewer iterations and avoiding oscillation which is important at training time. However, also at test time, Tab. 1 shows that attacking our CCAT model effectively requires up to $T = 2000$ iterations and zero initialization so that RErr for τ@99%TPR stagnates. In contrast, PGD-Conf performs better against AT even for smaller T and without momentum or backtracking. Thus, finding high-confidence adversarial examples against CCAT is more difficult than for AT. Overall, this illustrates our immense effort put into attacking our proposed defense with an adapted attack, novel optimization techniques together with large number of iterations and black box attacks for avoiding gradient obfuscation. We are sure that our defense cannot be easily broken and thus our reported performance is reliable.

Evaluation Metrics: Fig. 4 shows ROC and RErr curves on SVHN, considering AT and CCAT with $\rho_{\text{pow}} = 10$, i.e., using the power transition from Eq. (6). The ROC curves, and the corresponding AUC value, quantify how well (successful) adversarial examples can be distinguished from (correctly classified) test examples. Note our conservative choice to use the confidence threshold τ at 99%TPR (τ depends only on correctly classified test examples, not adversarial examples), loosing at most 1% correctly classified examples. We note that RErr implicitly includes FPR as well as Err; additionally, allowing confidence thresholding will naturally also improve robustness for AT. On SVHN and Cifar10, we found the power transition with $\rho_{\text{pow}} = 10$ from Eq. (6) to work best. Up to $\rho = 10$, performance regarding RErr for τ@99% TPR continuously improves and after $\rho_{\text{pow}} = 10$ performance stagnates, as shown in detail in the appendix. On MNIST, interestingly, exponential transition with $\rho_{\text{exp}} = 7$ performs best; we assume that the slight bias towards the true label preserved in the exponential transition helps.

		MNIST ($\epsilon = 0.3$)		**SVHN** ($\epsilon = 0.03$)		**CIFAR10** ($\epsilon = 0.03$)	
		τ@**99**%TPR		τ@**99**%TPR		τ@**99**%TPR	
		ROC AUC	FPR in %	ROC AUC	FPR in %	ROC AUC	FPR in %
Rand	Normal	0.34	100.0	0.95	72.8	0.83	87.1
	AT	0.99	44.5	**1.00**	0.9	0.60	100.0
	CCAT	**1.00**	**0.0**	**1.00**	**0.0**	**1.00**	**0.0**
Dist	Normal	0.34	100.0	0.43	100.0	0.49	100.0
	AT	0.63	100.0	0.89	98.8	0.31	100.0
	CCAT	**1.00**	**0.0**	**1.00**	**0.0**	**1.00**	**0.0**

Table 3: **Noise and distal adversarial example results on MNIST, SVHN and Cifar10.** Robustness against uniform noise (Rand) and distal adversarial examples (Dist), i.e., high-confidence adversarial computed on uniform noise using our L_∞ PGD-Conf attack by considering Eq. (4) without any true label; we use $\epsilon = 0.3$ on MNIST, $\epsilon = 0.03$ on SVHN/Cifar10. We report ROC AUC and FPR for a confidence threshold of τ@99%TPR.

4.2 Results

In Tab. 2, we report the main results of our paper, namely robustness across all evaluated L_∞ attacks for the same ϵ used during training and an increased ϵ. As RErr for τ@99%TPR is always lower than its unthresholded variant for $\tau = 0$, showing the general advantage of allowing rejection of adversarial examples, we concentrate on the newly introduced thresholded metrics. While on MNIST, CCAT incurs a drop of roughly 6% in RErr, and on Cifar10 a drop of roughly 5% against L_∞ with the same ϵ as during training, it significantly outperforms AT on SVHN, by more than 16%. On SVHN and Cifar10, Err is additionally improved – on Cifar10, the improvement is particularly significant with roughly 6%. For larger ϵ, robustness of AT degrades significantly, while CCAT is able to preserve robustness to some extend, especially on SVHN. Only on Cifar10, RErr degrades similarly to AT. In terms of generalization to another threat model, here L_2 attacks, CCAT outperforms AT significantly. We note that on all datasets, the considered ϵ values of 3 on MNIST and 1 on SVHN/Cifar10 correspond to L_2-balls which are *not* contained in the L_∞-ball used during training. Robustness of AT degrades significantly, while CCAT generalizes to this new attack model. Here one can clearly see the effect of better extrapolation properties of CCAT.

As second experiment we report in Tab. 3 results for detecting uniform noise and adversarial uniform noise (i.e., distal adversarial examples) based on their confidence. For the latter, we sample uniform noise and subsequently use PGD-Conf to maximize the confidence (without considering any true label in Eq. (4)) in the L_∞-ball around the noise point. We use the same hyper-parameters and ϵ values as used for PGD-Conf in Tab. 2. Although ROC AUC values are high on uniform noise, an FPR of 40% or higher shows that AT assigns high confidence to uniform noise. When maximizing confidence on uniform noise, FPR approaches 100% on all datasets. In contrast CCAT allows to separate these attacks perfectly from test examples. This further supports that CCAT induces a bias beyond the ϵ-ball used for training.

5 Conclusion

We proposed **confidence-calibrated adversarial training (CCAT)** which addresses two limitations of regular adversarial training (Madry et al., 2018; Szegedy et al., 2013): an apparent accuracy-robustness problem, i.e., adversarial training tends to worsen accuracy; and, more importantly, the lack of “generalizable” robustness, i.e., obtaining robust models against a larger class of adversarial attacks than used during training (e.g., by allowing larger adversarial perturbations or other threat models). CCAT achieves comparable or better robustness against the threat model at training time with better test accuracy on SVHN and Cifar10. However, in strong contrast to adversarial training, CCAT is able to generalize to stronger attacks in L_∞, L_2-attacks and reduces confidence on (adversarial) uniform noise as it naturally extrapolates beyond the L_∞-ball used during training. This opens up new directions of research aiming at models which are robust in a broad sense.

REFERENCES

Naveed Akhtar and Ajmal Mian. Threat of adversarial attacks on deep learning in computer vision: A survey. *arXiv.org*, abs/1801.00553, 2018.

Maksym Andriushchenko. Provable adversarial defenses for boosting. Master's thesis, Saarland University, August 2019.

Anish Athalye and Nicholas Carlini. On the robustness of the CVPR 2018 white-box adversarial example defenses. *arXiv.org*, abs/1804.03286, 2018.

Anish Athalye, Nicholas Carlini, and David A. Wagner. Obfuscated gradients give a false sense of security: Circumventing defenses to adversarial examples. *arXiv.org*, abs/1802.00420, 2018.

Arjun Nitin Bhagoji, Warren He, Bo Li, and Dawn Song. Exploring the space of black-box attacks on deep neural networks. *arXiv.org*, abs/1712.09491, 2017.

Battista Biggio and Fabio Roli. Wild patterns: Ten years after the rise of adversarial machine learning. *Pattern Recognition*, 84:317–331, 2018.

Wieland Brendel and Matthias Bethge. Comment on "biologically inspired protection of deep networks from adversarial attacks". *arXiv.org*, abs/1704.01547, 2017.

Qi-Zhi Cai, Chang Liu, and Dawn Song. Curriculum adversarial training. In *IJCAI*, pp. 3740–3747, 2018.

Nicholas Carlini and David Wagner. Towards evaluating the robustness of neural networks. In *SP*, 2017.

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 *AISec*, 2017.

Yinpeng Dong, Fangzhou Liao, Tianyu Pang, Hang Su, Jun Zhu, Xiaolin Hu, and Jianguo Li. Boosting adversarial attacks with momentum. In *CVPR*, 2018.

Ian Goodfellow, Yao Qin, and David Berthelot. Evaluation methodology for attacks against confidence thresholding models, 2019. URL `https://openreview.net/forum?id=H1g0piA9tQ`.

Ian J Goodfellow, Jonathon Shlens, and Christian Szegedy. Explaining and harnessing adversarial examples. *arXiv.org*, abs/1412.6572, 2014.

Sven Gowal, Krishnamurthy Dvijotham, Robert Stanforth, Rudy Bunel, Chongli Qin, Jonathan Uesato, Relja Arandjelovic, Timothy A. Mann, and Pushmeet Kohli. On the effectiveness of interval bound propagation for training verifiably robust models. *arXiv.org*, abs/1810.12715, 2018.

Edward Grefenstette, Robert Stanforth, Brendan O'Donoghue, Jonathan Uesato, Grzegorz Swirszcz, and Pushmeet Kohli. Strength in numbers: Trading-off robustness and computation via adversarially-trained ensembles. *arXiv.org*, abs/1811.09300, 2018.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. In *ICCV*, 2015.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *CVPR*, 2016.

Matthias Hein and Maksym Andriushchenko. Formal guarantees on the robustness of a classifier against adversarial manipulation. In *NeurIPS*, 2017.

Matthias Hein, Maksym Andriushchenko, and Julian Bitterwolf. Why ReLU networks yield high-confidence predictions far away from the training data and how to mitigate the problem. *CVPR*, 2019.

Ruitong Huang, Bing Xu, Dale Schuurmans, and Csaba Szepesvári. Learning with a strong adversary. *arXiv.org*, abs/1511.03034, 2015.

Andrew Ilyas, Logan Engstrom, Anish Athalye, and Jessy Lin. Black-box adversarial attacks with limited queries and information. In *ICML*, 2018a.

Andrew Ilyas, Logan Engstrom, and Aleksander Madry. Prior convictions: Black-box adversarial attacks with bandits and priors. *arXiv.org*, abs/1807.07978, 2018b.

Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *ICML*, 2015.

Alex Krizhevsky. Learning multiple layers of features from tiny images. Technical report, 2009.

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

Hyeungill Lee, Sungyeob Han, and Jungwoo Lee. Generative adversarial trainer: Defense to adversarial perturbations with GAN. *arXiv.org*, abs/1705.03387, 2017.

Yanpei Liu, Xinyun Chen, Chang Liu, and Dawn Song. Delving into transferable adversarial examples and black-box attacks. *arXiv.org*, abs/1611.02770, 2016.

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

Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu. Towards deep learning models resistant to adversarial attacks. *ICLR*, 2018.

Matthew Mirman, Timon Gehr, and Martin T. Vechev. Differentiable abstract interpretation for provably robust neural networks. In *ICML*, pp. 3575–3583, 2018.

Takeru Miyato, Shin-ichi Maeda, Masanori Koyama, Ken Nakae, and Shin Ishii. Distributional smoothing with virtual adversarial training. *ICLR*, 2016.

Nina Narodytska and Shiva Prasad Kasiviswanathan. Simple black-box adversarial attacks on deep neural networks. In *CVPR Workshops*, 2017.

Yuval Netzer, Tao Wang, Adam Coates, Alessandro Bissacco, Bo Wu, and Andrew Y. Ng. Reading digits in natural images with unsupervised feature learning. In *NeurIPS*, 2011.

Adam Paszke, Sam Gross, Soumith Chintala, Gregory Chanan, Edward Yang, Zachary DeVito, Zeming Lin, Alban Desmaison, Luca Antiga, and Adam Lerer. Automatic differentiation in pytorch. In *NeurIPS Workshops*, 2017.

F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine learning in Python. *JMLR*, 12:2825–2830, 2011.

Julien Pérolat, Mateusz Malinowski, Bilal Piot, and Olivier Pietquin. Playing the game of universal adversarial perturbations. *CoRR*, abs/1809.07802, 2018.

Sayantan Sarkar, Ankan Bansal, Upal Mahbub, and Rama Chellappa. UPSET and ANGRI : Breaking high performance image classifiers. *arXiv.org*, abs/1707.01159, 2017.

Ludwig Schmidt, Shibani Santurkar, Dimitris Tsipras, Kunal Talwar, and Aleksander Madry. Adversarially robust generalization requires more data. *CoRR*, arXiv.org, 2018.

Ali Shafahi, Mahyar Najibi, Zheng Xu, John P. Dickerson, Larry S. Davis, and Tom Goldstein. Universal adversarial training. *arXiv.org*, abs/1811.11304, 2018.

Uri Shaham, Yutaro Yamada, and Sahand Negahban. Understanding adversarial training: Increasing local stability of supervised models through robust optimization. *Neurocomputing*, 307:195–204, 2018.

Yash Sharma and Pin-Yu Chen. Attacking the madry defense model with l1-based adversarial examples. *arXiv.org*, abs/1710.10733, 2017.

Aman Sinha, Hongseok Namkoong, and John C. Duchi. Certifiable distributional robustness with principled adversarial training. *ICLR*, 2018.

Yang Song, Rui Shu, Nate Kushman, and Stefano Ermon. Generative adversarial examples. *arXiv.org*, abs/1805.07894, 2018.

David Stutz, Matthias Hein, and Bernt Schiele. Disentangling adversarial robustness and generalization. *CVPR*, 2019.

Jiawei Su, Danilo Vasconcellos Vargas, and Kouichi Sakurai. One pixel attack for fooling deep neural networks. *arXiv.org*, abs/1710.08864, 2017.

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

Florian Tramèr, Alexey Kurakin, Nicolas Papernot, Dan Boneh, and Patrick D. McDaniel. Ensemble adversarial training: Attacks and defenses. *arXiv.org*, abs/1705.07204, 2017.

Dimitris Tsipras, Shibani Santurkar, Logan Engstrom, Alexander Turner, and Aleksander Madry. Robustness may be at odds with accuracy. *arXiv.org*, abs/1805.12152, 2018.

Tsui-Wei Weng, Huan Zhang, Hongge Chen, Zhao Song, Cho-Jui Hsieh, Luca Daniel, Duane S. Boning, and Inderjit S. Dhillon. Towards fast computation of certified robustness for relu networks. In *ICML*, 2018.

Eric Wong and J. Zico Kolter. Provable defenses against adversarial examples via the convex outer adversarial polytope. In *ICML*, 2018.

Xiaoyong Yuan, Pan He, Qile Zhu, Rajendra Rana Bhat, and Xiaolin Li. Adversarial examples: Attacks and defenses for deep learning. *arXiv.org*, abs/1712.07107, 2017.

Valentina Zantedeschi, Maria-Irina Nicolae, and Ambrish Rawat. Efficient defenses against adversarial attacks. In *AISec*, 2017.

A Proof of Proposition 1

Proposition 2. *We consider a classification problem with two points $x = 0$ and $x = \epsilon$ in $\mathbb{R}$ with deterministic labels, that is $p(y = 2|x = 0) = 1$ and $p(y = 1|x = \epsilon) = 1$ and the problem is fully determined by the probability $p_0 = p(x = 0)$ as $p(x = \epsilon) = 1 - p_0$. The Bayes error of this classification problem is zero. The Bayes optimal classifier of*

- 100% *adversarial training yields an error of* $\min\{p_0, 1 - p_0\}$.
- *adversarial training with* 50% *adversarial and* 50% *clean examples yields an error of* $\min\{p_0, 1 - p_0\}$.
- *Our confidence-calibrated adversarial training with* 50% *clean and* 50% *adversarial examples yields* zero error *if* $\lambda < \min\{{p_0}/{1-p_0}, {1-p_0}/{p_0}\}$.

Proof. We introduce

$$a = f_1(0) - f_2(0), \quad b = f_1(\epsilon) - f_2(\epsilon). \tag{8}$$

and express the confidences of class 1 and 2 in terms of these quantities.

$$\hat{p}(y = 2|x = x) = \log\left(\frac{e^{f_2(x)}}{e^{f_1(x)} + e^{f_2(x)}}\right) = \log\left(1 + e^{f_1(x) - f_2(x)}\right) = \log\left(1 + e^{a}\right).$$

$$\hat{p}(y = 1|x = x) = \log\left(\frac{e^{f_1(x)}}{e^{f_1(x)} + e^{f_2(x)}}\right) = \log\left(1 + e^{f_2(x) - f_1(x)}\right) = \log\left(1 + e^{-b}\right).$$

For our confidence-calibrated adversarial training one first has to solve:

$$\underset{\|\delta\|_\infty \leq \epsilon}{\operatorname{argmax}}\ \hat{p}(y = 2|x + \delta) = \begin{cases} 0 & \text{if } a > b \\ \epsilon & \text{else.} \end{cases}.$$

$$\underset{\|\delta\|_\infty \leq \epsilon}{\operatorname{argmax}}\ \hat{p}(y = 1|x + \delta) = \begin{cases} \epsilon & \text{if } a > b \\ 0 & \text{else.} \end{cases}.$$

With this we can write the total loss (remember that we have half normal cross-entropy loss and half the loss for the adversarial part with the modified "labels") as

$$\begin{aligned}
L(a, b) = p_0 \Big[&\log(1 + e^{a}) \mathbb{1}_{a \geq b} \\
&+ \mathbb{1}_{a<b}\Big(\frac{(1+\lambda)}{2} \log(1 + e^{b}) + \frac{(1-\lambda)}{2} \log(1 + e^{-b})\Big)\Big] \\
&+ (1 - p_0)\Big[\log(1 + e^{-b}) \mathbb{1}_{a \geq b} \\
&+ \mathbb{1}_{a<b}\Big(\frac{(1+\lambda)}{2} \log(1 + e^{-a}) + \frac{(1-\lambda)}{2} \log(1 + e^{a})\Big)\Big] \\
&+ \log(1 + e^{a}) p_0 + \log(1 + e^{-b})(1 - p_0),
\end{aligned}$$

where we have omitted a global factor $\frac{1}{2}$ for better readability. We distinguish two sets in the optimization. First we consider the case $a \geq b$. Then it is easy to see that in order to minimize the loss we have $a = b$.

$$\partial_a L = \frac{e^{a}}{1 + e^{a}} p_0 (1 + \gamma) - \frac{e^{-a}}{1 + e^{-a}} (1 - p_0)(1 + \gamma)$$

This yields $e^{a} = \frac{1-p_0}{p_0}$ or $a = \log\left(\frac{1-p_0}{p_0}\right)$. The other case is $a < b$. We get

$$\begin{aligned}
\partial_a L =& \Big[\frac{(1+\lambda)}{2} \frac{-e^{-a}}{1 + e^{-a}} + \frac{(1-\lambda)}{2} \frac{e^{a}}{1 + e^{a}}\Big](1 - p_0) \\
&+ p_0 \frac{e^{a}}{1 + e^{a}} \\
\partial_b L =& \Big[\frac{(1+\lambda)}{2} \frac{e^{b}}{1 + e^{b}} + \frac{(1-\lambda)}{2} \frac{-e^{b}}{1 + e^{-b}}\Big] p_0 \\
&+ (1 - p_0) \frac{-e^{-b}}{1 + e^{-b}}
\end{aligned}$$

This yields the solution

$$a^* = \log\Big(\frac{\frac{1+\lambda}{2}(1-p_0)}{p_0 + \frac{1-\lambda}{2}(1-p_0)}\Big)$$
$$b^* = \log\Big(\frac{\frac{1-\lambda}{2}p_0 + (1-p_0)}{\frac{1+\lambda}{2}p_0}\Big)$$

It is straightforward to check that $a^* < b^*$ for all $0 < p_0 < 1$. We have $a^* < 0$ if

$$1 > \frac{1-p_0}{p_0}\lambda,$$

and $b^* > 0$ if

$$1 > \frac{p_0}{1-p_0}\lambda.$$

Thus we recover the Bayes classifier if

$$\lambda < \min\Big\{\frac{1-p_0}{p_0}, \frac{p_0}{1-p_0}\Big\}.$$

Now we consider the approach by Madry et al. (2018) with 100% adversarial training. The loss can be written as

$$\begin{aligned} L(a,b) =& \max\Big\{\log(1+e^a), \log(1+e^b)\Big\}p_0 \\ &+ \max\Big\{\log(1+e^{-a}), \log(1+e^{-b})\Big\}(1-p_0) \end{aligned}$$

The loss is minimized if $a = b$ as then both maxima are minimal. This results in the loss

$$L(a) = \log(1+e^a)p_0 + \log(1+e^{-a})(1-p_0).$$

The critical point is attained at $a^* = b^* = \log\Big(\frac{1-p_0}{p_0}\Big)$. This is never Bayes optimal for all $0 < p_0 < 1$.

Next we consider 50% adversarial plus 50% clean training. The loss can be written as

$$\begin{aligned} L(a,b) =& \max\Big\{\log(1+e^a), \log(1+e^b)\Big\}p_0 \\ &+ \max\Big\{\log(1+e^{-a}), \log(1+e^{-b})\Big\}(1-p_0) \\ &+ \log(1+e^a)p_0 + \log(1+e^{-b})(1-p_0) \end{aligned}$$

We make a case distinction. If $a \geq b$, then the loss reduces to

$$\begin{aligned} L(a,b) =& \log(1+e^a)p_0 + \log(1+e^{-b})(1-p_0) \\ &+ \log(1+e^a)p_0 + \log(1+e^{-b})(1-p_0) \\ &\geq L(a,a) \\ &= 2\log(1+e^a)p_0 + 2\log(1+e^{-a})(1-p_0) \end{aligned}$$

Solving for the critical point yields $a^* = \log\Big(\frac{1-p_0}{p_0}\Big) = b^*$. Next we consider the set $a \leq b$. This yields the loss

$$\begin{aligned} L(a,b) =& \log(1+e^b)p_0 + \log(1+e^{-a})(1-p_0) \\ &+ \log(1+e^a)p_0 + \log(1+e^{-b})(1-p_0) \end{aligned}$$

Solving for the critical point yields $a^* = \log\Big(\frac{1-p_0}{p_0}\Big) = b^*$ which fulfills $a \leq b$. Actually, it coincides with the solution found already. One does not recover the Bayes classifier for any $0 < p_0 < 1$.

Moreover, we note that

$$a^* = b^* = \begin{cases} > 0 & \text{if } p_0 < \frac{1}{2}, \\ < 0 & \text{if } p_0 > \frac{1}{2}. \end{cases}$$

Thus, we classify $x = 0$ correctly, if $p_0 > \frac{1}{2}$ and $x = \epsilon$ correctly if $p_0 < \frac{1}{2}$. Thus the error is given by $\min\{p_0, 1-p_0\}$. $\square$

Algorithm 2 Pseudo-code for the used projected gradient descent (PGD) procedure to maximize Eq. (9) or Eq. (10) subject to the constraints $\tilde{x}_i = x_i + \delta_i \in [0,1]$ and $\|\delta\|_\infty \leq \epsilon$; in practice, the procedure is applied on batches of inputs. The algorithm is also easily adapted to work with a L_2-norm; only the projections on line 6 and 24 needs to be adapted.

```
    input: example $x$ with label $y$
    input: number of iterations $T$
    input: learning rate $\gamma$, momentum $\beta$, learning rate factor $\alpha$
    input: initial $\delta^{(0)}$, e.g., Eq. (11) or $\delta^{(0)} = 0$
 1: $v := 0$ {best objective achieved}
 2: $\tilde{x} := x + \delta^{(0)}$ {best adversarial example}
 3: $g^{(-1)} := 0$ {accumulated gradients}
 4: for $t = 0, \ldots, T$ do
 5:   {projection onto $L_\infty$ $\epsilon$-ball and on $[0,1]$:}
 6:   clip $\delta_i^{(t)}$ to $[-\epsilon, \epsilon]$
 7:   clip $x_i + \delta_i^{(t)}$ to $[0,1]$
 8:   {forward and backward pass to get objective and gradient:}
 9:   $v^{(t)} := \mathcal{F}(x + \delta^{(t)}, y)$
10:   $g^{(t)} := \text{sign}\left(\nabla_{\delta^{(t)}} \mathcal{F}(x + \delta^{(t)}, y)\right)$
11:   {keep track of adversarial example resulting in best objective:}
12:   if $v^{(t)} > v$ then
13:     $v := v^{(t)}$
14:     $\tilde{x} := x + \delta^{(t)}$
15:   end if
16:   {iteration $T$ is only meant to check whether last update improved objective:}
17:   if $t = T$ then
18:     break
19:   end if
20:   {integrate momentum term:}
21:   $g^{(t)} := \beta g^{(t-1)} + (1 - \beta) g^{(t)}$
22:   {“try” the update step and see if objective increases:}
23:   $\hat{\delta}^{(t)} := \delta^{(t)} + \gamma g^{(t)}$
24:   clip $\hat{\delta}_i^{(t)}$ to $[-\epsilon, \epsilon]$
25:   clip $x_i + \hat{\delta}_i^{(t)}$ to $[0,1]$
26:   $\hat{v}^{(t)} := \mathcal{F}(x + \hat{\delta}^{(t)}, y)$
27:   {only keep the update if the objective increased; otherwise decrease learning rate:}
28:   if $\hat{v}^{(t)} \geq v^{(t)}$ then
29:     $\delta^{(t+1)} := \hat{\delta}^{(t)}$
30:   else
31:     $\gamma := \gamma/\alpha$
32:   end if
33: end for
34: return $\tilde{x}$, $\tilde{v}$
```

B Experiments

We give additional details on our experimental setup, specifically regarding attacks, training and the used evaluation metrics. Afterwards, we include additional experimental results, including ablation studies, results for 98% true positive rate (TPR), and results per employed attack.

B.1 Attacks

Complementary to the description of the projected gradient descent (PGD) attack by Madry et al. (2018) and our adapted attack, we provide a detailed algorithm in Alg. 2. We note that the objective

RErr @99%TPR in % on **MNIST** (L_∞ attack with $\epsilon = 0.3$)

Optimization	momentum+backtrack					momentum			
Initialization	zero				rand	zero		zero	
Iterations T	40	200	2000	4000	4000	60	300	60	300
AT	0.4	0.4	0.4	0.3	0.6	0.4	0.6	0.4	0.4
AT Conf	0.8	0.8	0.8	0.8	1.1	1.0	1.1	1.0	1.0
CCAT, $\rho_{\text{exp}} = 7$	0.6	3.3	4.9	6.8	5.5	0.9	3.8	0.0	0.1

RErr @99%TPR in % on **SVHN** (L_∞ attack with $\epsilon = 0.03$)

Optimization	momentum+backtrack					momentum			
Initialization	zero				rand	zero		zero	
Iterations T	40	200	2000	4000	4000	60	300	60	300
AT	38.4	46.2	49.9	50.1	51.8	37.7	38.1	29.9	30.8
AT Conf	27.4	40.5	46.9	47.3	48.1	27.1	28.5	21.1	23.8
CCAT, $\rho_{\text{pow}} = 10$	4.0	5.0	22.8	23.3	5.2	2.6	2.6	2.6	2.6

RErr @99%TPR in % on **CIFAR10** (L_∞ attack with $\epsilon = 0.03$)

Optimization	momentum+backtrack					momentum			
Initialization	zero				rand	zero		zero	
Iterations T	40	200	2000	4000	4000	60	300	60	300
AT	60.9	60.8	60.8	60.8	60.9	60.9	60.9	57.4	57.6
AT Conf	60.4	60.6	60.5	60.5	60.9	60.4	60.6	56.2	56.6
CCAT, $\rho_{\text{pow}} = 10$	14.8	16.2	40.2	41.3	34.9	7.2	7.2	7.2	7.2

Table 4: **Detailed attack ablation studies on MNIST, SVHN and Cifar10.** Complementary to Tab. 1, we compare our L_∞ PGD-Conf attack with T iterations and different combinations of momentum, backtracking and initialization on all three datasets. We consider AT, AT trained with PGD-Conf, and CCAT; we report RErr for confidence threshold τ@99%TPR. As backtracking requires an additional forward pass per iteration, we use $T = 60$ and $T = 300$ for attacks without backtracking to be comparable to attacks with $T = 40$ and $T = 200$ with backtracking.

maximized in (Madry et al., 2018) is

$$\mathcal{F}(x + \delta, y) = \mathcal{L}(f(x + \delta; w), y) \tag{9}$$

where $\mathcal{L}$ denotes the cross-entropy loss, $f(\cdot; w)$ denotes the model and (x, y) is an input-label pair from the test set. Our adapted attack, in contrast, maximizes

$$\mathcal{F}(x + \delta, y) = \max_{k \neq y} f_k(x + \delta; w). \tag{10}$$

We denote these two variants as PGD-CE and PGD-Conf, respectively. Deviating from (Madry et al., 2018), we initialize δ uniformly over directions and norm (instead of uniform initialization over the volume of the ϵ-ball):

$$\delta = u\epsilon \frac{\delta'}{\|\delta'\|_\infty}, \quad \delta' \sim \mathcal{N}(0, I), u \sim U(0, 1) \tag{11}$$

where δ' is sampled from a standard Gaussian and $u \in [0, 1]$ from a uniform distribution. We also consider zero initialization, i.e., $\delta = 0$; while for random initialization we always consider multiple attempts, 10 for PGD-Conf and 50 for PGD-CE, with zero initialization, we use only 1 attempt.

Alg. 2 also gives more details on the employed momentum and backtracking scheme. These two "tricks" add two additional hyper-parameters to the number of iterations T and the learning rate γ, namely the momentum parameter β and the learning rate factor α. After each iteration, the computed update, already including the momentum term, is only applied if this improves the objective. This is checked through an additional forward pass. If not, the learning rate is divided by α, and the update is rejected. Alg. 2 includes this scheme as an algorithm for an individual test example x with label y for brevity; however, extending it to work on batches, which is used in our paper, is straight-forward. In practice, for PGD-CE, with $T = 200$ iterations, we use $\beta = 0.9$ and $\alpha = 1.25$; for PGD-Conf, with $T = 2000$ iterations, we use $\beta = 0.9$ and $\alpha = 1.1$.

We also give more details on the used black-box attacks. For random sampling, we apply Eq. (11) $T = 5000$ times in order to maximize Eq. (10). We also implemented the black-box attack of Ilyas

MNIST (L_∞ attack with $\epsilon = 0.3$ during training)

	τ=0			τ@99%TPR	
	Err in %	RErr in %	ROC AUC	Err in %	RErr in %
Normal	0.40	100.00	0.34	0.10	100.00
AT	0.50	5.60	0.97	0.00	0.40
AT Conf	0.50	6.00	0.98	0.10	1.10
CCAT, $\rho_{\text{exp}} = 3$	0.40	88.10	0.99	0.10	11.90
CCAT, $\rho_{\text{exp}} = 5$	0.40	88.70	0.99	0.10	10.80
CCAT, $\rho_{\text{exp}} = 7$	0.50	74.70	0.99	0.10	5.70
CCAT, $\rho_{\text{exp}} = 9$	0.30	86.80	0.99	0.10	11.00
CCAT, $\rho_{\text{pow}} = 10$	0.30	64.80	0.95	0.10	17.30

SVHN (L_∞ attack with $\epsilon = 0.03$ during training)

	τ=0			τ@99%TPR	
	Err in %	RErr in %	ROC AUC	Err in %	RErr in %
Normal	3.60	99.90	0.17	2.60	99.90
AT	3.40	56.90	0.55	2.50	54.90
AT Conf	3.70	58.70	0.61	2.80	52.50
CCAT, $\rho_{\text{pow}} = 1$	2.70	82.40	0.74	2.20	43.00
CCAT, $\rho_{\text{pow}} = 2$	2.90	79.60	0.68	2.10	44.20
CCAT, $\rho_{\text{pow}} = 6$	2.90	72.10	0.64	1.80	32.80
CCAT, $\rho_{\text{pow}} = 10$	2.90	91.00	0.67	2.10	38.50
CCAT, $\rho_{\text{exp}} = 7$	2.90	73.10	0.66	2.00	54.70

CIFAR10 (L_∞ attack with $\epsilon = 0.03$ during training)

	τ=0			τ@99%TPR	
	Err in %	RErr in %	ROC AUC	Err in %	RErr in %
Normal	8.30	100.00	0.20	7.40	100.00
AT	16.60	61.30	0.65	15.10	60.90
AT Conf	16.10	61.70	0.63	15.10	61.50
CCAT, $\rho_{\text{pow}} = 1$	9.70	95.30	0.63	8.70	72.40
CCAT, $\rho_{\text{pow}} = 2$	9.70	95.10	0.60	8.40	70.60
CCAT, $\rho_{\text{pow}} = 6$	9.20	94.10	0.54	8.00	69.80
CCAT, $\rho_{\text{pow}} = 10$	10.10	95.00	0.60	8.70	63.00
CCAT, $\rho_{\text{exp}} = 7$	13.20	77.40	0.66	11.70	68.60

Table 5: **Training ablation studies on MNIST, SVHN and Cifar10.** We report results for different choices of ρ and transitions, cf. Eq. (6). We report RErr and Err with confidence threshold $\tau = 0$ and τ@99%TPR as well as ROC AUC. The models are tested against our L_∞ PGD-Conf attack with $T = 2000$ iterations and zero as well as random initialization. On MNIST, the exponential transition, especially $\rho_{\text{exp}} = 7$ performs best; on Cifar10, the power transition with $\rho_{\text{pow}} = 10$ works best – performance stagnates for $\rho_{\text{pow}} > 10$. On SVHN, we also use $\rho_{\text{pow}} = 10$, although $\rho_{\text{pow}} = 6$ shows better results. However, against larger ϵ-balls, we found that $\rho_{\text{pow}} = 10$ works significantly better.

et al. (2018a) using a population of 50 and variance of 0.1 for estimating the gradient in Line 10 of Alg. 2; a detailed algorithm is provided in (Ilyas et al., 2018a). We use a learning rate of 0.001 (note that the gradient is signed, as in (Madry et al., 2018)) and also integrated a momentum with $\beta = 0.9$ and backtracking as described in Alg. 2 with $\alpha = 1.1$ and $T = 2000$ iterations. We use zero and random initialization; in the latter case we allow 10 random retries. For the simple black-box attack we follow the algorithmic description in (Narodytska & Kasiviswanathan, 2017) considering only axis-aligned perturbations of size ϵ per pixel. We run the attack for $T = 2000$ iterations and allow 10 random retries. Finally, we use a variant of the cube attack proposed in (Andriushchenko, 2019). We run the attack for $T = 5000$ iterations with a probability of change of 0.05. We emphasize that, except for (Ilyas et al., 2018a), these attacks are not gradient-based and do not approximate the gradient.

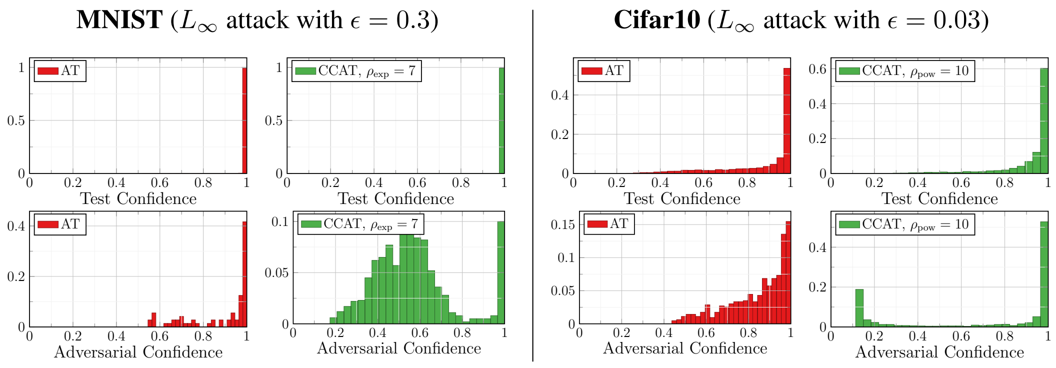

Figure 5: **Confidence histograms on MNIST and Cifar10.** As in Fig. 3, we show histograms of confidences on correctly classified test examples (top) and on successful adversarial examples (bottom) for both AT and CCAT. Note that on AT, the number of successful adversarial examples is usually lower than on CCAT, i.e., reflects the RErr for $\tau = 0$ in Tab. 2; for CCAT in contrast, nearly all adversarial examples are successful, while only a part has high confidence. Histograms obtained for the worst-case adversarial examples across all tested L_∞ attacks.

B.2 TRAINING

As described in Sec. 4, we follow the ResNet-20 architecture by He et al. (2016) implemented in PyTorch (Paszke et al., 2017). For training we use a batch size of 100 and train for 100 and 200 epochs on MNIST and SVHN/Cifar10, respectively: this holds for normal training, adversarial training (AT) and confidence-calibrated adversarial training (CCAT). For the latter two, we use PGD-CE and PGD Conf, respectively, for $T = 40$ iterations, momentum and backtracking ($\beta = 0.9$, $\alpha = 1.5$). For PGD-CE we use a learning rate of 0.05, 0.01 and 0.005 on MNIST, SVHN and Cifar10. For PGD-Conf we use a learning rate of 0.05. For training, we use standard stochastic gradient descent, starting with a learning rate of 0.1 on MNIST/SVHN and 0.075 on Cifar10. The learning rate is multiplied by 0.95 after each epoch. We do not use weight decay; but the network includes batch normalization (Ioffe & Szegedy, 2015). On SVHN and Cifar10, we use random cropping, random flipping (only Cifar10) and contrast augmentation during training. We always train on 50% clean and 50% adversarial examples per batch, i.e., each batch contains both clean and adversarial examples which is important when using batch normalization.

B.3 EVALUATION METRICS

For reproducibility and complementing the discussion in the main paper, we describe the used evaluation metrics and evaluation procedure in more detail. Adversarial examples are computed on the first 1000 examples of the test set; the used confidence threshold is computed on the last 1000 examples of the test set; test errors are computed on all test examples minus the last 1000. As we consider multiple attacks, and some attacks allow multiple random attempts, we always consider the worst case adversarial example per test example and across all attacks/attempts; the worst-case is selected based on confidence.

ROC AUC: To compute ROC curves, and the area under the curve, i.e., ROC AUC, we define negatives as *successful* adversarial examples (on correctly classified test examples) and positives as the corresponding *correctly classified* test examples. The ROC AUC as well as the curve itself can easily be calculated using (Pedregosa et al., 2011). Practically, the generated curve could be used to directly estimate a threshold corresponding to a pre-determined true positive rate (TPR). However, this requires interpolation; after trying several constant interpolation schemes, we concluded that the results are distorted significantly, especially for TPRs close to 100%. Thus, we followed a simpler scheme as mentioned above: on a held out validation set of size 1000 (the last 1000 samples of the test set), we sorted the corresponding confidences, and picked the confidence threshold in order to obtain the desired TPR, e.g., 99%, on this set exactly.

Robust Test Error: For clarity, we repeat our confidence-integrated definition of the robust test error:

$$\text{RErr}(\tau) = \frac{\sum\limits_{n=1}^{N} \mathbb{1}_{f(x_n)\neq y_n} \mathbb{1}_{c(x_n)\geq\tau} + \sum\limits_{n=1}^{N} \mathbb{1}_{f(x_n)=y_n} \mathbb{1}_{f(\tilde{x}_n)\neq y_n} \mathbb{1}_{c(\tilde{x}_n)\geq\tau}}{\sum\limits_{n=1}^{N} \mathbb{1}_{c(x_n)\geq\tau} + \sum\limits_{n=1}^{N} \mathbb{1}_{c(x_n)<\tau} \mathbb{1}_{c(\tilde{x}_n)\geq\tau} \mathbb{1}_{f(x_n)=y_n} \mathbb{1}_{f(\tilde{x}_n)\neq y_n}}, \tag{12}$$

As described in the main paper, Eq. (12) quantifies the performance of a classifier with reject-option at a specific confidence threshold τ on both clean and adversarial examples. The regular robust test error, corresponding to Eq. (12) at $\tau = 0$, can also be written as

$$\text{RErr}(0) = \frac{\sum\limits_{n=1}^{N} \mathbb{1}_{f(x_n)\neq y_n} + \sum\limits_{n=1}^{N} \mathbb{1}_{f(x_n)=y_n} \mathbb{1}_{f(\tilde{x}_n)\neq y_n}}{\sum\limits_{n=1}^{N} 1}. \tag{13}$$

The robust test error is easy to handle as it quantifies overall performance, including generalization on clean examples (first term in Eq. (13)) and robustness on adversarial examples corresponding to correctly classified clean examples (second term in Eq. (13)). Additionally, the robust test error lies in $[0, 1]$. Generalizing Eq. (13) to $\tau > 0$ is non-trivial due to the following considerations: First, when integrating a confidence threshold, the reference set (i.e., the denominator) needs to be adapted. Otherwise, the metric will not reflect the actual performance after thresholding, i.e., rejecting examples – specifically, the values are not comparable across different confidence thresholds τ. Then, Eq. (13) might be adapted as follows:

$$\text{RErr}(\tau) = \frac{\sum\limits_{n=1}^{N} \mathbb{1}_{f(x_n)\neq y_n} \mathbb{1}_{c(x_n)\geq\tau} + \sum\limits_{n=1}^{N} \mathbb{1}_{f(x_n)=y_n} \mathbb{1}_{f(\tilde{x}_n)\neq y_n} \mathbb{1}_{c(\tilde{x}_n)\geq\tau}}{\sum\limits_{n=1}^{N} \mathbb{1}_{c(x_n)\geq\tau}}. \tag{14}$$

Note that in the nominator, we only consider clean and adversarial examples with confidence above the threshold τ, i.e., $c(x_n) \geq \tau$ and $c(\tilde{x}_n) \geq \tau$. Similarly, the denominator has been adapted accordingly, as after rejection, the reference set changes, too. However, this formulation has a problem when adversarial examples obtain higher confidence than the original clean examples. Thus, second, we need to account for the case where $c(\tilde{x}_n) > \tau \geq c(x_n)$, i.e., an adversarial example obtains a higher confidence than the corresponding clean example. Then, the nominator may exceed the denominator, resulting in a value larger than one. To mitigate this problem, we need to include exactly this case in the denominator. Formalizing this case, we see that it corresponds exactly to the second term in the denominator or Eq. (12):

$$\sum_{n=1}^{N} \mathbb{1}_{c(x_n)<\tau} \mathbb{1}_{c(\tilde{x}_n)\geq\tau} \mathbb{1}_{f(x_n)=y_n} \mathbb{1}_{f(\tilde{x}_n)\neq y_n} \tag{15}$$

Overall, with Eq. (12) we obtain a metric that lies within $[0, 1]$, is comparable across thresholds τ and, for $\tau = 0$, reduces to the regular robust test error as used in related work (Madry et al., 2018).

Worst-Case Evaluation: As described in Sec. 4, we report worst-case results across all evaluated attacks; individual results are partly reported in Tab. 7 8, and 9. In practice, we accumulate all adversarial examples from multiple attempts (e.g., multiple random restarts for PGD or random sampling) and across all attacks for each test example individually. Subsequently, we only keep the (successful) adversarial example with highest confidence (per test example). We use these "worst-case" adversarial examples for evaluation. Compared to related work, this procedure mimics a strong attacker that uses different attack strategies in parallel and always picks the strongest adversarial example per test example.

B.4 Ablation Study

Complementing Sec. 4, we include ablation studies for MNIST, SVHN and Cifar10; for our attack in Tab. 4 and training in Tab. 5.

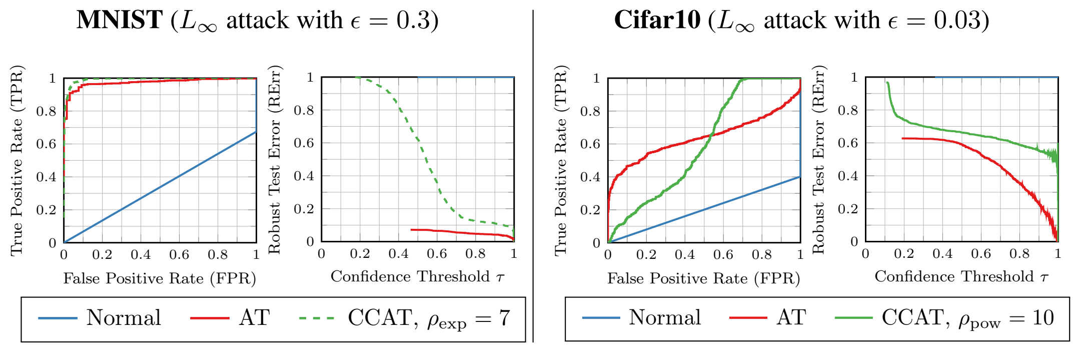

Figure 6: **ROC and RErr curves on MNIST and Cifar10.** ROC curves, i.e. FPR plotted against TPR for all possible confidence thresholds τ, and RErr curves, i.e., RErr over confidence threshold τ for AT and CCAT, including different ρ parameters. Worst-case adversarial examples across all L_∞ attacks were tested.

Regarding the proposed attack, i.e., PGD-Conf, using momentum and backtracking, Tab. 4 shows that the main observations for SVHN can be transferred to MNIST and Cifar10. Only the improvement of backtracking *and* momentum over just using momentum cannot be confirmed on MNIST. Note that for fair comparison, T iterations with backtracking are equivalent to $3/2T$ iterations without backtracking; which is why we include results for $T = 60$ and $T = 300$. However, the importance of using enough iterations, i.e., $T = 2000$, and zero initialization to attack CCAT is still clearly visible. Interestingly, against AT on SVHN, more iterations are also beneficial, while this is not required on MNIST and Cifar10.

Tab. 5 also reports results for CCAT with different transitions, cf. Eq. (6), and values for ρ. As mentioned before, on SVHN and Cifar10, power transition with $\rho_{\text{pow}} = 10$ works best; for larger ρ performance stagnates. It is also important to note that the power transition does not preserve a bias towards to true label, i.e., for the maximum possible perturbation ($\|\delta\|_\infty = \epsilon$), Eq. (6) forces the network to predict a purely uniform distribution. This is in contrast to the exponential transition, where the true one-hot distribution always receives a non-zero weight. On MNIST, we found this to work considerably better.

B.5 Analysis

For further analysis, Fig. 5 shows confidence histograms for AT and CCAT on MNIST and Cifar10. The confidence histograms for CCAT reflect the expected behavior: adversarial examples are mostly successful in changing the label, which is supported by high RErr values for confidence threshold $\tau = 0$, but their confidence is pushed towards a uniform distributions. For AT, in contrast, successful adversarial examples – fewer in total – generally obtain high confidence; this results in confidence thresholding being not effective for AT. This behavior, however, is less pronounced on MNIST. Here, the exponential transition results in adversarial examples with confidence slightly higher than uniform confidence, i.e., 0.1. This might be the result of preserving a bias towards the true label through Eq. (6). In fact, for lower ρ, we found that this behavior is pronounced, until, for very small ρ, the behavior of AT is obtained.

In Fig. 7, we plot the probabilities for all ten classes along an adversarial direction. We note that these directions do not necessarily correspond to successful adversarial examples. Instead, we chose the first 10 test examples on SVHN. The adversarial examples were obtained using our L_∞ PGD-Conf attack with $T = 2000$ iterations and zero initialization for $\epsilon = 0.03$. For AT, we usually observe a change in predictions along these directions; some occur within $\|\delta\|_\infty \leq \epsilon$, corresponding to successful adversarial examples, some occur for $\|\delta\|_\infty > \epsilon$, corresponding to unsuccessful adversarial examples (within ϵ). However, AT always assigns high confidence. Thus, when allowing larger adversarial perturbations at test time, robustness of AT reduces significantly. For CCAT, in contrast, there are only few such cases; more often, the model achieves a near uniform prediction for small $\|\delta\|_\infty$ and extrapolates this behavior beyond the ϵ-ball used for training. On SVHN, this behavior successfully allows to generalize the robustness to larger adversarial perturbations. Furthermore, these plots illustrate why using more iterations at test time, and using techniques such as

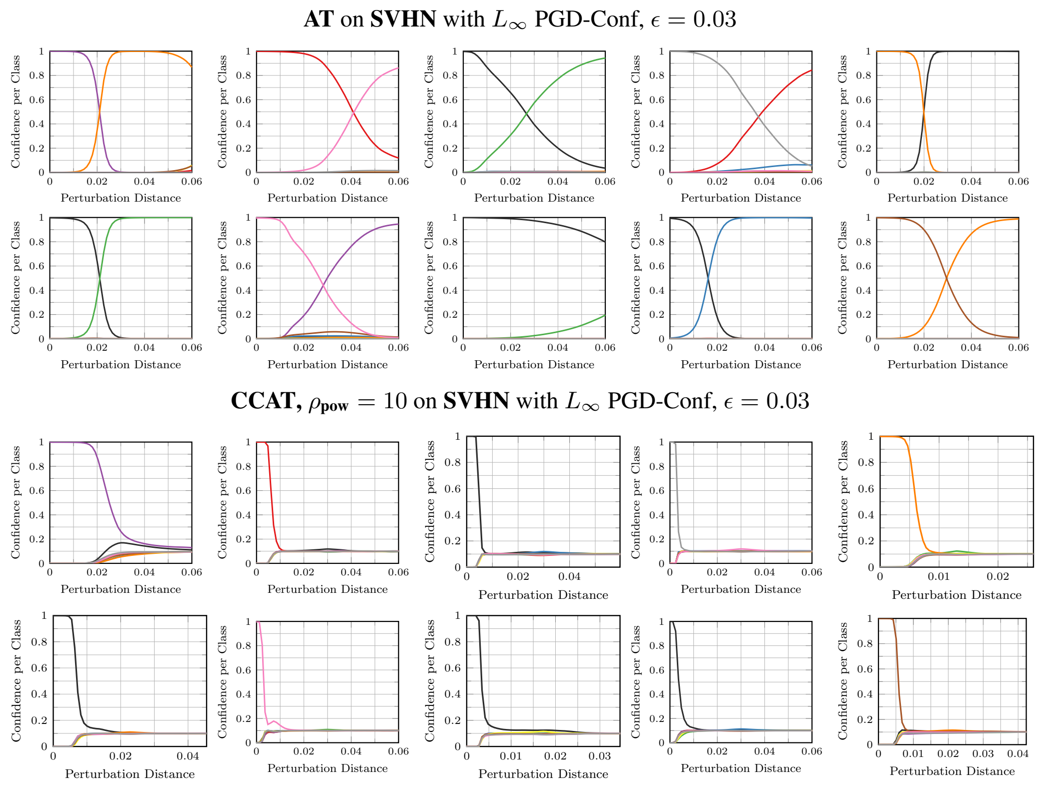

Figure 7: **Effect of confidence calibration on SVHN.** Confidences for classes along adversarial directions for AT and CCAT, $\rho_{\text{pow}} = 10$. Adversarial examples were computed using PGD-Conf with $T = 2000$ iterations and $\gamma = 0.001$ and zero initialization. For both AT and CCAT, the first ten examples of the SVHN test set are shown.

momentum and backtracking, are necessary to find adversarial examples as the objective becomes more complex compared to AT.

B.6 Results

In the following, we present and discuss complementary results corresponding to the main results of our paper as presented in Tab. 2 and 3. To this end, we consider requiring only 98% TPR instead of 99% TPR, and most importantly, break down our analysis by the different white-box and black-box attacks used.

Main results for 98% **TPR:** Tab. 6 reports results (corresponding to Tab. 2) requiring only 98%TPR. This implies, that compared to 99%TPR, up to 1% more correctly classified test examples can be rejected. For relatively simple tasks such as MNIST and SVHN, where Err values are low, this is a significant "sacrifice". However, as can be seen, robustness in terms of RErr only improves slightly. We found that the same holds for 95%TPR.

Per-Attack Results.: Finally Tab. 7, 8 and 9 we break down the results of Tab. 2 regarding the used attacks. For simplicity we focus on PGD-CE and PGD-Conf while reporting the used black-box attacks together, i.e., taking the worst-case adversarial examples across all black-box attacks. On MNIST, where AT performs very well in practice, it is striking that for $^4/_3\epsilon = 0.4$ even black-box attacks are able to reduce robustness completely, resulting in high RErr. This observation also transfers to SVHN and Cifar10. For CCAT, black-box attacks are only effective on Cifar10, where they result in roughly 87% RErr with τ@99%TPR. It can also be seen that PGD-CE performs significantly worse against our CCAT compared to AT, which shows that it is essential to optimize the right objective, i.e., maximize confidence against CCAT.

MNIST (L_∞ attack with $\epsilon = 0.3$ during training)							
		τ=0			τ@98%TPR		
Attack	Training	Err in %	RErr in %	ROC AUC	Err in %	RErr in %	τ
$L_\infty, \epsilon = 0.3$	AT	0.50	7.20	0.97	0.00	0.50	1.00
	CCAT	0.50	100.00	0.99	0.00	5.60	1.00
$L_\infty, \epsilon = 0.4$	AT	0.50	100.00	0.20	0.00	100.00	1.00
	CCAT	0.50	100.00	0.94	0.00	29.20	1.00
$L_2, \epsilon = 3$	AT	0.50	98.80	0.73	0.00	67.60	1.00
	CCAT	0.50	82.60	1.00	0.00	0.60	1.00
SVHN (L_∞ attack with $\epsilon = 0.03$ during training)							
$L_\infty, \epsilon = 0.03$	AT	3.40	57.30	0.55	1.80	52.90	0.71
	CCAT	2.90	97.80	0.70	1.50	36.50	0.79
$L_\infty, \epsilon = 0.06$	AT	3.40	89.00	0.32	1.80	86.90	0.71
	CCAT	2.90	99.80	0.70	1.50	34.80	0.79
$L_2, \epsilon = 1$	AT	3.40	92.40	0.26	1.80	91.30	0.71
	CCAT	2.90	81.80	0.91	1.50	15.60	0.79
CIFAR10 (L_∞ attack with $\epsilon = 0.03$ during training)							
$L_\infty, \epsilon = 0.03$	AT	16.60	62.70	0.64	14.00	61.90	0.39
	CCAT	10.10	96.70	0.60	8.30	67.20	0.44
$L_\infty, \epsilon = 0.06$	AT	16.60	93.70	0.35	14.00	93.60	0.39
	CCAT	10.10	99.20	0.43	8.30	90.90	0.44
$L_2, \epsilon = 1$	AT	16.60	74.40	0.59	14.00	73.60	0.39
	CCAT	10.10	81.90	0.77	8.30	45.70	0.44

Table 6: **Main results for 98%TPR on MNIST, SVHN and Cifar10.** While reporting results for 99%TPR in the main paper, cf. Tab. 2, reducing the TPR requirement to 98%TPR generally improves results for CCAT, but only slightly. Again, we report Err and RErr for $\tau = 0$ and τ@98%TPR as well as ROC AUC.

MNIST (L_∞ attack with $\epsilon = 0.3$ during training)							
		τ=0			τ@99%TPR		
Attack	Training	Err in %	RErr in %	ROC AUC	Err in %	RErr in %	τ
L_∞, $\epsilon = 0.3$, PGD Conf	Normal	0.40	100.00	0.34	0.10	100.00	0.98
	AT	0.50	5.60	0.97	0.00	0.40	1.00
	CCAT	0.50	74.70	0.99	0.10	5.70	0.99
L_∞, $\epsilon = 0.3$, PGD CE	Normal	0.40	100.00	0.34	0.10	100.00	0.98
	AT	0.50	6.70	0.97	0.00	0.80	1.00
	CCAT	0.50	100.00	1.00	0.10	4.30	0.99
L_∞, $\epsilon = 0.3$, Black-Box	Normal	0.40	100.00	0.34	0.10	100.00	0.98
	AT	0.50	7.20	0.98	0.00	1.00	1.00
	CCAT	0.50	100.00	1.00	0.10	0.40	0.99
L_∞, $\epsilon = 0.4$, PGD Conf	Normal	0.40	100.00	0.34	0.10	100.00	0.98
	AT	0.50	99.80	0.36	0.00	97.80	1.00
	CCAT	0.50	97.10	0.96	0.10	15.40	0.99
L_∞, $\epsilon = 0.4$, PGD CE	Normal	0.40	100.00	0.34	0.10	100.00	0.98
	AT	0.50	100.00	0.20	0.00	100.00	1.00
	CCAT	0.50	100.00	0.97	0.10	29.60	0.99
L_∞, $\epsilon = 0.4$, Black-Box	Normal	0.40	100.00	0.34	0.10	100.00	0.98
	AT	0.50	100.00	0.23	0.00	100.00	1.00
	CCAT	0.50	100.00	0.99	0.10	3.90	0.99

MNIST (L_∞ attack with $\epsilon = 0.3$ during training)							
		τ=0			τ@99%TPR		
Attack	Training	Err in %	RErr in %	ROC AUC	Err in %	RErr in %	τ
L_2, $\epsilon = 3$, PGD Conf	AT	0.50	3.40	0.98	0.00	0.20	1.00
	CCAT	0.50	4.40	1.00	0.10	0.00	0.99
L_2, $\epsilon = 3$, PGD CE	AT	0.50	29.20	0.93	0.00	11.50	1.00
	CCAT	0.50	82.40	1.00	0.10	0.90	0.99
L_2, $\epsilon = 3$, Black-Box	AT	0.50	98.70	0.73	0.00	80.10	1.00
	CCAT	0.50	38.40	1.00	0.10	0.60	0.99

Table 7: **Per-attack L_∞ and L_2 results on MNIST.** Per-attack results considering PGD-CE, as in Madry et al. (2018), our PGD-Conf and the remaining black-box attacks, see text. We report results for L_∞ attacks using $\epsilon = 0.3$, also used for training, and $4/3\epsilon = 0.4$ as well as L_2 attacks with $\epsilon = 3$. For the black-box attacks, we take the per-example worst-case across all black-box attacks.

SVHN (L_∞ attack with $\epsilon = 0.03$ during training)

		τ=0			τ@99%TPR		
Attack	Training	Err in %	RErr in %	ROC AUC	Err in %	RErr in %	τ
L_∞, $\epsilon = 0.03$, PGD Conf	Normal	3.60	99.90	0.17	2.60	99.90	0.78
	AT	3.40	56.90	0.55	2.50	54.90	0.56
	CCAT	2.90	91.00	0.67	2.10	38.50	0.60
L_∞, $\epsilon = 0.03$, PGD CE	Normal	3.60	100.00	0.17	2.60	99.90	0.78
	AT	3.40	50.70	0.68	2.50	43.20	0.56
	CCAT	2.90	94.90	1.00	2.10	2.60	0.60
L_∞, $\epsilon = 0.03$, Black-Box	Normal	3.60	99.70	0.23	2.60	99.70	0.78
	AT	3.40	46.20	0.95	2.50	30.80	0.56
	CCAT	2.90	79.50	1.00	2.10	6.30	0.60
L_∞, $\epsilon = 0.06$, PGD Conf	Normal	3.60	100.00	0.16	2.60	100.00	0.78
	AT	3.40	86.10	0.32	2.50	84.70	0.56
	CCAT	2.90	98.70	0.70	2.10	36.80	0.60
L_∞, $\epsilon = 0.06$, PGD CE	Normal	3.60	100.00	0.16	2.60	100.00	0.78
	AT	3.40	88.90	0.66	2.50	88.00	0.56
	CCAT	2.90	100.00	0.99	2.10	17.20	0.60
L_∞, $\epsilon = 0.06$, Black-Box	Normal	3.60	100.00	0.17	2.60	100.00	0.78
	AT	3.40	84.00	0.78	2.50	82.30	0.56
	CCAT	2.90	82.30	1.00	2.10	4.80	0.60

SVHN (L_∞ attack with $\epsilon = 0.03$ during training)

		τ=0			τ@99%TPR		
Attack	Training	Err in %	RErr in %	ROC AUC	Err in %	RErr in %	τ
L_2, $\epsilon = 1$, PGD Conf	AT	3.40	78.70	0.50	2.50	77.10	0.56
	CCAT	2.90	74.40	0.90	2.10	18.40	0.60
L_2, $\epsilon = 1$, PGD CE	AT	3.40	92.40	0.27	2.50	92.00	0.56
	CCAT	2.90	100.00	0.99	2.10	3.70	0.60
L_2, $\epsilon = 1$, Black-Box	AT	3.40	29.80	0.98	2.50	13.70	0.56
	CCAT	2.90	100.00	1.00	2.10	2.60	0.60

Table 8: **Per-attack L_∞ and L_2 results on SVHN.** Per-attack results considering PGD-CE, as in Madry et al. (2018), our PGD-Conf and the remaining black-box attacks, see text. We report results for L_∞ attacks using $\epsilon = 0.03$, also used for training, and $2\epsilon = 0.06$ as well as L_2 attacks with $\epsilon = 1$. For the black-box attacks, we take the per-example worst-case across all black-box attacks.

CIFAR10 (L_∞ attack with $\epsilon = 0.03$ during training)							
		τ=0			τ@99%TPR		
Attack	Training	Err in %	RErr in %	ROC AUC	Err in %	RErr in %	τ
L_∞, $\epsilon = 0.03$, PGD Conf	Normal	8.30	100.00	0.20	7.40	100.00	0.59
	AT	16.60	61.30	0.65	15.10	60.90	0.35
	CCAT	10.10	95.00	0.60	8.70	63.00	0.40
L_∞, $\epsilon = 0.03$, PGD CE	Normal	8.30	100.00	0.20	7.40	100.00	0.59
	AT	16.60	62.30	0.67	15.10	61.40	0.35
	CCAT	10.10	100.00	0.99	8.70	9.80	0.40
L_∞, $\epsilon = 0.03$, Black-Box	Normal	8.30	100.00	0.21	7.40	100.00	0.59
	AT	16.60	57.30	0.70	15.10	56.90	0.35
	CCAT	10.10	96.40	0.90	8.70	49.30	0.40
L_∞, $\epsilon = 0.06$, PGD Conf	Normal	8.30	100.00	0.20	7.40	100.00	0.59
	AT	16.60	92.20	0.37	15.10	92.10	0.35
	CCAT	10.10	97.30	0.49	8.70	66.80	0.40
L_∞, $\epsilon = 0.06$, PGD CE	Normal	8.30	100.00	0.20	7.40	100.00	0.59
	AT	16.60	93.70	0.40	15.10	93.60	0.35
	CCAT	10.10	100.00	0.98	8.70	10.40	0.40
L_∞, $\epsilon = 0.06$, Black-Box	Normal	8.30	100.00	0.20	7.40	100.00	0.59
	AT	16.60	87.20	0.50	15.10	87.10	0.35
	CCAT	10.10	99.50	0.78	8.70	87.00	0.40

CIFAR10 (L_∞ attack with $\epsilon = 0.03$ during training)							
		τ=0			τ@99%TPR		
Attack	Training	Err in %	RErr in %	ROC AUC	Err in %	RErr in %	τ
L_2, $\epsilon = 1$, PGD Conf	AT	16.60	65.30	0.63	15.10	64.90	0.35
	CCAT	10.10	80.90	0.78	8.70	45.20	0.40
L_2, $\epsilon = 1$, PGD CE	AT	16.60	74.60	0.61	15.10	73.90	0.35
	CCAT	10.10	100.00	0.95	8.70	18.90	0.40
L_2, $\epsilon = 1$, Black-Box	AT	16.60	36.90	0.81	15.10	35.80	0.35
	CCAT	10.10	100.00	1.00	8.70	8.80	0.40

Table 9: **Per-attack L_∞ and L_2 results on Cifar10.** Per-attack results considering PGD-CE, as in Madry et al. (2018), our PGD-Conf and the remaining black-box attacks, see text. We report results for L_∞ attacks using $\epsilon = 0.03$, also used for training, and $2\epsilon = 0.06$ as well as L_2 attacks with $\epsilon = 1$. For the black-box attacks, we take the per-example worst-case across all black-box attacks.