Why ReLU networks yield high-confidence predictions far away from the training data and how to mitigate the problem

为什么ReLU可以在非常不同于训练的数据获得高置信度的预测 并且如何减轻这个问题

Matthias Hein
University of Tübingen

Maksym Andriushchenko
Saarland University

Julian Bitterwolf
University of Tübingen

Abstract

用于安全的问题上不仅要求泛化性好，也要知道缺陷所在

Classifiers used in the wild, in particular for safety-critical systems, should not only have good generalization properties but also should know when they don't know, in particular make low confidence predictions far away from the training data. We show that ReLU type neural networks which yield a piecewise linear classifier function fail in this regard as they produce almost always high confidence predictions far away from the training data. For bounded domains like images we propose a new robust optimization technique similar to adversarial training which enforces low confidence predictions far away from the training data. We show that this technique is surprisingly effective in reducing the confidence of predictions far away from the training data while maintaining high confidence predictions and similar test error on the original classification task compared to standard training.

本文证明ReLU可以获得分段的线性函数 但仍然导致泛化性不好
提出优化技术类似于对抗训练保证相似数据上置信度好 不相似低

1. Introduction

Neural networks have recently obtained state-of-the-art performance in several application domains like object recognition and speech recognition. They have become the de facto standard for many learning tasks. Despite this great success story and very good prediction performance there are also aspects of neural networks which are undesirable. One property which is naturally expected from any classifier is that it should know when it does not know or said more directly: far away from the training data a classifier should not make high confidence predictions. This is particularly important in safety-critical applications like autonomous driving or medical diagnosis systems where such an input should either lead to the fact that other redundant sensors are used or that a human doctor is asked to check the diagnosis. It is thus an important property of a classifier which however has not received much attention despite the fact that it seems to be a minimal requirement for any classifier.

There have been many cases reported where high confidence predictions are made far away from the training data by neural networks, e.g. fooling images [28], for out-of-distribution images [15] or in a medical diagnosis task [21]. Moreover, it has been observed that, even on the original task, neural networks often produce overconfident predictions [12]. A related but different problem are adversarial samples where very small modifications of the input can change the classifier decision [31, 11, 30]. Apart from methods which provide robustness guarantees for neural networks [14, 34, 29, 24] which give still only reasonable guarantees for small networks, up to our knowledge the only approach which has not been broken again [6, 5, 2] is adversarial training [23] using robust optimization techniques.

While several methods have been proposed to adjust overconfident predictions on the true input distribution using softmax calibration [12], ensemble techniques [18] or uncertainty estimation using dropout [10], only recently the detection of out-of-distribution inputs [15] has been tackled. The existing approaches basically either use adjustment techniques of the softmax outputs [9, 22] by temperature rescaling [12] or they use a generative model like a VAE or GAN to model boundary inputs of the true distribution [20, 33] in order to discriminate in-distribution from out-of-distribution inputs directly in the training process. While all these approaches are significant steps towards obtaining more reliable classifiers, the approaches using a generative model have been recently challenged by [26] which report that generative approaches can produce highly confident density estimates for inputs outside of the class they are supposed to model. Moreover, note that the quite useful models for confidence calibration on the input distribution like [10, 12, 18] cannot be used for out-of-distribution detection as it has been observed in [21]. Another approach is the introduction of a rejection option into the classifier [32, 4], in order to avoid decisions the classifier is not certain about.

In this paper we will show that for the class of ReLU

networks, that are networks with fully connected, convolutional and residual layers, where just ReLU or leaky ReLU are used as activation functions and max or average pooling for convolution layers, basically any neural network which results in a piecewise affine classifier function, produce arbitrarily high confidence predictions far away from the training data. This implies that techniques which operate on the output of the classifier cannot identify these inputs as out-of-distribution inputs. On the contrary we formalize the well known fact that RBF networks produce almost uniform confidence over the classes far away from the training data, which shows that there exist classifiers which satisfy the minimal requirement of not being confident in areas where one has never seen data. Moreover, we propose a robust optimization scheme motivated by adversarial training [23] which simply enforces uniform confidence predictions on noise images which are by construction far away from the true images. We show that our technique not only significantly reduces confidence on such noise images, but also on other unrelated image classification tasks and in some cases even for adversarial samples generated for the original classification task. The training procedure is simple, needs no adaptation for different out-of-distribution tasks, has similar complexity as standard adversarial training and achieves similar or marginally worse generalization performance on the original classification task.

2. ReLU networks produce piecewise affine functions

We quickly review in this section the fact that ReLU networks lead to continuous piecewise affine classifiers, see [1, 8], which we briefly summarize in order to set the ground for our main theoretical result in Section 3.

Definition 2.1. *A function $f : \mathbb{R}^d \to \mathbb{R}$ is called* piecewise affine *if there exists a finite set of polytopes $\{Q_r\}_{r=1}^M$ (referred to as* linear regions *of f) such that $\cup_{r=1}^M Q_r = \mathbb{R}^d$ and f is an affine function when restricted to every Q_r.*

Feedforward neural networks which use piecewise affine activation functions (e.g. ReLU, leaky ReLU) and are linear in the output layer can be rewritten as continuous piecewise affine functions [1]. This includes fully connected, convolutional, residual layers and even skip connections as all these layers are just linear mappings. Moreover, it includes further average pooling and max pooling. More precisely, the classifier is a function $f : \mathbb{R}^d \to \mathbb{R}^K$, where K are the number of classes, such that each component $f_i : \mathbb{R}^d \to \mathbb{R}$, is a continuous piecewise affine function and the K components $(f_i)_{i=1}^K$ have the same set of linear regions. Note that explicit upper bounds on the number of linear regions have been given [25].

In the following we follow [8]. For simplicity we just present fully connected layers. Denote by $\sigma : \mathbb{R} \to \mathbb{R}$, $\sigma(t) = \max\{0, t\}$, the ReLU activation function, by $L+1$ the number of layers and $W^{(l)} \in \mathbb{R}^{n_l \times n_{l-1}}$ and $b^{(l)} \in \mathbb{R}^{n_l}$ respectively are the weights and offset vectors of layer l, for $l = 1, \dots, L+1$ and $n_0 = d$. For $x \in \mathbb{R}^d$ one defines $g^{(0)}(x) = x$. Then one can recursively define the pre- and post-activation output of every layer as 预激活和后激活

$$f^{(k)}(x) = W^{(k)} g^{(k-1)}(x) + b^{(k)}, \quad \text{and}$$
$$g^{(k)}(x) = \sigma(f^{(k)}(x)), \quad k = 1, \dots, L,$$

so that the resulting classifier is obtained as $f^{(L+1)}(x) = W^{(L+1)} g^{(L)}(x) + b^{(L+1)}$.

Let $\Delta^{(l)}, \Sigma^{(l)} \in \mathbb{R}^{n_l \times n_l}$ for $l = 1, \dots, L$ be diagonal matrices defined elementwise as 反向求导过程的导数

$$\Delta^{(l)}(x)_{ij} = \begin{cases} \operatorname{sign}(f_i^{(l)}(x)) & \text{if } i = j, \\ 0 & \text{else.} \end{cases},$$
$$\Sigma^{(l)}(x)_{ij} = \begin{cases} 1 & \text{if } i = j \text{ and } f_i^{(l)}(x) > 0, \\ 0 & \text{else.} \end{cases}.$$

Note that for leaky ReLU the entries would be 1 and α instead. This allows to write $f^{(k)}(x)$ as composition of affine functions, that is

$$\begin{aligned} f^{(k)}(x) =& W^{(k)} \Sigma^{(k-1)}(x) \Big(W^{(k-1)} \Sigma^{(k-2)}(x) \\ & \times \Big(\dots \Big(W^{(1)} x + b^{(1)} \Big) \dots \Big) + b^{(k-1)} \Big) + b^{(k)}, \end{aligned}$$

We can further simplify the previous expression as $f^{(k)}(x) = V^{(k)} x + a^{(k)}$, with $V^{(k)} \in \mathbb{R}^{n_k \times d}$ and $a^{(k)} \in \mathbb{R}^{n_k}$ given by

$$V^{(k)} = W^{(k)} \Big(\prod_{l=1}^{k-1} \Sigma^{(k-l)}(x) W^{(k-l)} \Big) \quad \text{and}$$
$$a^{(k)} = b^{(k)} + \sum_{l=1}^{k-1} \Big(\prod_{m=1}^{k-l} W^{(k+1-m)} \Sigma^{(k-m)}(x) \Big) b^{(l)}.$$

The polytope $Q(x)$, the linear region containing x, can be characterized as an intersection of $N = \sum_{l=1}^L n_l$ half spaces given by

$$\Gamma_{l,i} = \big\{ z \in \mathbb{R}^d \,\big|\, \Delta^{(l)}(x) \big(V_i^{(l)} z + a_i^{(l)} \big) \geq 0 \big\},$$

for $l = 1, \dots, L$, $i = 1, \dots, n_l$, namely

$$Q(x) = \bigcap_{l=1,\dots,L} \bigcap_{i=1,\dots,n_l} \Gamma_{l,i}.$$

Note that N is also the number of hidden units of the network. Finally, we can write

$$f^{(L+1)}(z)\Big|_{Q(x)} = V^{(L+1)}z + a^{(L+1)},$$

which is the affine restriction of f to $Q(x)$.

3. Why ReLU networks produce high confidence predictions far away from the training data

With the explicit description of the piecewise linear classifier resulting from a ReLU type network from Section 2, we can now formulate our main theorem. It shows that, as long a very mild condition on the network holds, for any $\epsilon > 0$ one can always find for (almost) **all** directions an input z far away from the training data which realizes a confidence of $1 - \epsilon$ on z for a certain class. However, before we come to this result, we first need a technical lemma needed in the proof, which uses that all linear regions are polytopes and thus convex sets.

Lemma 3.1. *Let $\{Q_i\}_{l=1}^R$ be the set of linear regions associated to the ReLU-classifier $f : \mathbb{R}^d \to \mathbb{R}^K$. For any $x \in \mathbb{R}^d$ there exists $\alpha \in \mathbb{R}$ with $\alpha > 0$ and $t \in \{1, \dots, R\}$ such that $\beta x \in Q_t$ for all $\beta \geq \alpha$.*

Proof. Suppose the statement would be false. Then there exist $\{\beta_i\}_{i=1}^\infty$ with $\beta_i \geq 0$, $\beta_i \geq \beta_j$ if $i \leq j$ and $\beta_i \to \infty$ as $i \to \infty$ such that for $\gamma \in [\beta_i, \beta_{i+1})$ we have $\gamma x \in Q_{r_i}$ with $r_i \in \{1, \dots, R\}$ and $r_{i-1} \neq r_i \neq r_{i+1}$. As there are only finitely many regions there exist $i, j \in \mathbb{N}$ with $i < j$ such that $r_i = r_j$, in particular $\beta_i x \in Q_{r_i}$ and $\beta_j x \in Q_{r_i}$. However, as the linear regions are convex sets also the line segment $[\beta_i x, \beta_j x] \in Q_{r_i}$. However, that implies $\beta_i = \beta_j$ as neighboring segments are in different regions which contradicts the assumption. Thus there can only be finitely many $\{\beta_i\}_{i=1}^M$ and the $\{r_i\}_{i=1}^M$ have to be all different, which finishes the proof. □

Using Lemma 3.1 we can now state our first main result.

Theorem 3.1. *Let $\mathbb{R}^d = \cup_{l=1}^R Q_l$ and $f(x) = V^l x + a^l$ be the piecewise affine representation of the output of a ReLU network on Q_l. Suppose that V^l does not contain identical rows for all $l = 1, \dots, R$, then for almost any $x \in \mathbb{R}^d$ and $\epsilon > 0$ there exists an $\alpha > 0$ and a class $k \in \{1, \dots, K\}$ such that for $z = \alpha x$ it holds*

$$\frac{e^{f_k(z)}}{\sum_{r=1}^K e^{f_r(z)}} \geq 1 - \epsilon.$$

Moreover, $\lim_{\alpha \to \infty} \frac{e^{f_k(\alpha x)}}{\sum_{r=1}^K e^{f_r(\alpha x)}} = 1.$

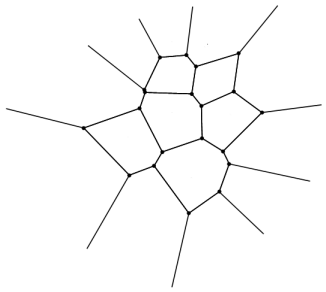

Figure 1: A decomposition of $\mathbb{R}^2$ into a finite set of polytopes. The outer polytopes extend to infinity. This is where ReLU networks realize arbitrarily high confidence predictions.

Proof. By Lemma 3.1 there exists a region Q_t with $t \in \{1, \dots, R\}$ and $\beta > 0$ such that for all $\alpha \geq \beta$ we have $\alpha x \in Q_t$. Let $f(z) = V^t z + a^t$ be the affine form of the ReLU classifier f on Q_t. Let $k^* = \arg\max_k \langle v_k^t, x \rangle$, where v_k^t is the k-th row of V^t. As V^t does not contain identical rows, that is $v_l^t \neq v_m^t$ for $l \neq m$, the maximum is uniquely attained up to a set of measure zero. If the maximum is unique, it holds for sufficiently large $\alpha \geq \beta$

$$\left\langle v_l^t - v_{k^*}^t, \alpha x \right\rangle + a_l^t - a_{k^*}^t < 0, \ \forall l \in \{1, \dots, K\} \backslash \{k^*\}. \tag{1}$$

Thus $\alpha x \in Q_t$ is classified as k^*. Moreover,

$$\frac{e^{f_{k^*}(\alpha x)}}{\sum_{l=1}^K e^{f_l(\alpha x)}} = \frac{e^{\left\langle v_{k^*}^t, \alpha x \right\rangle + a_k^t}}{\sum_{l=1}^K e^{\left\langle v_l^t, \alpha x \right\rangle + a_l^t}} \tag{2}$$

$$= \frac{1}{1 + \sum_{l \neq k^*}^K e^{\left\langle v_l^t - v_{k^*}^t, \alpha x \right\rangle + a_l^t - a_k^t}}. \tag{3}$$

By inequality (1) all the terms in the exponential are negative and thus by upscaling α, using $\langle v_{k^*}^t, x \rangle > \langle v_l^t, x \rangle$ for all $l \neq k^*$, we can get the exponential term arbitrarily close to 0. In particular,

$$\lim_{\alpha \to \infty} \frac{1}{1 + \sum_{l \neq k}^K e^{\left\langle v_l^t - v_{k^*}^t, \alpha x \right\rangle + a_l^t - a_k^t}} = 1.$$

□

Please note that the condition that for a region the linear part V^l need not contain two identical rows is very weak. It is hardly imaginable that this is ever true for a normally trained network unless the output of the network is constant anyway. Even if it is true, it just invalidates the assertion of the theorem for the points lying in this region. Without explicitly enforcing this condition it is basically impossible that this is true

for all possible asymptotic regions extending to infinity (see Figure 1). However, it is also completely open how this condition could be enforced during training of the network.

The result implies that for ReLU networks there exist infinitely many inputs which realize arbitrarily high confidence predictions of the networks. It is easy to see that the temperature rescaling of the softmax, $\frac{e^{f_k(x)/T}}{\sum_{l=1}^K e^{f_l(x)/T}}$, for temperature $T > 0$, as used in [22], will not be able to detect these cases, in particular since the first step of the method in [22] consists of going in the direction of increasing confidence. Also it is obvious that using a reject option in the classifier, see e.g. [3], will not help to detect these instances either. The result is negative in the sense that it looks like that without modifying the architecture of a ReLU network it is impossible to prevent this phenomenon. Please note that from the point of view of Bayesian decision theory the softmax function is the correct transfer function [19] for the cross-entropy loss turning the classifier output $f_k(x)$ into an estimate $\mathrm{P}(Y = k\,|x, f) = \frac{e^{f_k(x)}}{\sum_{l=1}^K e^{f_l(x)}}$ for the conditional probability at x.

While the previous result seems not to be known, the following result is at least qualitatively known [11] but we could not find a reference for it. In contrast to the ReLU networks it turns out that Radial Basis Function (RBF) networks have the property to produce approximately uniform confidence predictions far away from the training data. Thus there exist classifiers which satisfy the minimal requirement which we formulated in Section 1. In the following theorem we explicitly quantify what "far away" means in terms of parameters of the classifiers.

Theorem 3.2. *Let $f_k(x) = \sum_{l=1}^N \alpha_{kl} e^{-\gamma\|x-x_l\|_2^2}$, $k = 1,\dots,K$ be a RBF-network trained with cross-entropy loss on the training data $(x_i, y_i)_{i=1}^N$. We define $r_{\min} = \min_{l=1,\dots,N} \|x - x_l\|_2$ and $\alpha = \max_{r,k} \sum_{l=1}^N |\alpha_{rl} - \alpha_{kl}|$. If $\epsilon > 0$ and*

$$r_{\min}^2 \geq \frac{1}{\gamma} \log\Big(\frac{\alpha}{\log(1 + K\epsilon)}\Big),$$

then for all $k = 1,\dots,K$,

$$\frac{1}{K} - \epsilon \;\leq\; \frac{e^{f_k(x)}}{\sum_{r=1}^K e^{f_r(x)}} \;\leq\; \frac{1}{K} + \epsilon.$$

Proof. It holds $\frac{e^{f_k(x)}}{\sum_{r=1}^K e^{f_r(x)}} = \frac{1}{\sum_{r=1}^K e^{f_r(x)-f_k(x)}}$. With

$$|f_r(x) - f_k(x)| = \Big|\sum_l (\alpha_{rl} - \alpha_{kl}) e^{-\gamma\|x-x_l\|_2^2}\Big| \tag{4}$$

$$\leq e^{-\gamma r_{\min}^2} \sum_l |\alpha_{rl} - \alpha_{kl}| \tag{5}$$

$$\leq e^{-\gamma r_{\min}^2} \alpha, \tag{6}$$

we get

$$\frac{1}{\sum_{r=1}^K e^{f_r(x)-f_k(x)}} \geq \frac{1}{\sum_{r=1}^K e^{|f_r(x)-f_k(x)|}} \tag{7}$$

$$\geq \frac{1}{K e^{\alpha e^{-\gamma r_{\min}^2}}} \tag{8}$$

$$\geq \frac{1}{K}\frac{1}{1 + K\epsilon} \geq \frac{1}{K} - \epsilon, \tag{9}$$

where we have used in the third inequality the condition on $r_{\min}^2$ and in the last step we use $1 \geq (1-K\epsilon)(1+K\epsilon) = 1 - K^2\epsilon^2$. Similarly, we get

$$\frac{1}{\sum_{r=1}^K e^{f_r(x)-f_k(x)}} \leq \frac{1}{\sum_{r=1}^K e^{-|f_r(x)-f_k(x)|}}$$

$$\leq \frac{1}{K e^{-\alpha e^{-\gamma r_{\min}^2}}}$$

$$\leq \frac{1}{K}(1 + K\epsilon) \leq \frac{1}{K} + \epsilon.$$

This finishes the proof. □

We think that it is a very important open problem to realize a similar result as in Theorem 3.2 for a class of neural networks. Note that arbitrarily high confidence predictions for ReLU networks can be obtained only if the domain is unbounded e.g. $\mathbb{R}^d$. However, images are contained in $[0,1]^d$ and thus Theorem 3.1 does not directly apply, even though the technique can still in principle be used to produce high-confidence predictions. In the next section we propose a novel training scheme enforcing low confidence predictions on inputs far away from the training data.

4. Adversarial Confidence Enhanced Training

In this section we suggest a simple way to adjust the confidence estimation of a neural network far away from the training data, not necessarily restricted to ReLU networks studied in Theorem 3.1. Theorem 3.1 tells us that for ReLU networks a post-processing of the softmax scores is not sufficient to avoid high-confidence predictions far away from the training data - instead there seem to be two potential ways to tackle the problem: a) one uses an extra generative model either for

the in-distribution or for the out-distribution or b) one modifies directly the network via an adaptation of the training process so that uniform confidence predictions are enforced far away from the training data. As recently problems with generative models have been pointed out which assign high confidence to samples from the out distribution [26] and thus a) seems less promising we explore approach b).

We assume that it is possible to characterize a distribution of data points p_{out} on the input space for which we are sure that they do not belong to the true distribution p_{in} resp. the set of the intersection of their supports has zero or close to zero probability mass. An example of such a out-distribution p_{out} would be the uniform distribution on $[0,1]^{w\times h}$ ($w \times h$ gray scale images) or similar noise distributions. Suppose that the in-distribution consists of certain image classes like handwritten digits, then the probability mass of all images of handwritten digits under the p_{out} is zero (if it is really a low-dimensional manifold) or close to zero.

In such a setting the training objective can be written as a sum of two losses:

$$\frac{1}{N}\sum_{i=1}^{N} L_{CE}(y_i, f(x_i)) + \lambda\, \mathbb{E}\big[L_{p_{\text{out}}}(f, Z)\big], \tag{10}$$

where $(x_i, y_i)_{i=1}^N$ is the i.i.d. training data, Z has distribution p_{out} and

$$L_{CE}(y_i, f(x_i)) = -\log\Big(\frac{e^{f_{y_i}(x_i)}}{\sum_{k=1}^K e^{f_k(x_i)}}\Big) \tag{11}$$

$$L_{p_{\text{out}}}(f, z) = \frac{1}{K}\sum_{l=1}^{K} -\log\Big(\frac{e^{f_l(z)}}{\sum_{k=1}^K e^{f_k(z)}}\Big). \tag{12}$$

L_{CE} is the usual cross entropy loss on the original classification task and $L_{p_{\text{out}}}(f, z)$ is the cross-entropy loss evaluated at z between the uniform distribution over the classes and the predicted probability distribution, $\frac{e^{f_l(z)}}{\sum_{k=1}^K e^{f_k(z)}}$, on the classes using the softmax function as the link function. The full loss can be easily minimized by using SGD with batchsize B for the original data and adding $\lceil \lambda B \rceil$ samples from p_{out} on which one enforces a uniform distribution over the labels. We call this process in the following *confidence enhancing data augmentation (CEDA)*.

The problem with CEDA is that it might take too many samples to enforce low confidence anywhere. Moreover, it has been shown in the area of adversarial manipulation, that data augmentation is not sufficient for robust models. Thus we propose to use ideas from robust optimization similar to adversarial training to obtain robust networks [31, 11, 23] against adversarial manipulation, which are very small perturbations of the input changing the classifier decision. Thus we are enforcing low confidence not only at the point itself but actively minimize the worst case in a neighborhood of the point. This leads to the following formulation of *adversarial confidence enhancing training (ACET)*

$$\frac{1}{N}\sum_{i=1}^{N} L_{CE}(y_i, f(x_i)) + \lambda\, \mathbb{E}\big[\max_{\|u-Z\|_p\leq\epsilon} L_{p_{\text{out}}}(f, u)\big], \tag{13}$$

where in each SGD step one solves (approximately) for a given $z \sim p_{\text{out}}$ the optimization problem:

$$\max_{\|u-z\|_p\leq\epsilon} L_{p_{\text{out}}}(f, u). \tag{14}$$

In this paper we use always $p = \infty$. Note that if the distribution p_{out} and p_{in} have joint support, the maximum in (14) could be obtained at a point in the support of the true distribution. However, if p_{out} is a generic noise distribution like uniform noise or a smoothed version of it, then the number of cases where this happens has probability mass close to zero under p_{out} and thus does not negatively influence in (13) the loss L_{CE} on the true distribution. The optimization of ACET in (13) can be done using an adapted version of the PGD method of [23] for adversarial training where one does a few steps of projected gradient descent (potentially for a few restarts) and uses the u realizing the worst loss for computing the gradient. The resulting samples are more informative and thus lead to a faster and more significant reduction of high confidence predictions far away from the training data. We use $\epsilon = 0.3$ for gray-scale and $\epsilon = 0.1$ for color images. We present in Figure 2 and 3 for MNIST and CIFAR10 a few noise images together with their adversarial modification u generated by applying PGD to solve (14) (20 iterations for all MNIST, CIFAR10 and SVHN). One can observe that the generated images have no structure resembling images from the in-distribution.

5. Experiments

We follow in the evaluation [15, 22, 20] by training on one dataset and evaluating the confidence on other out of distribution datasets and noise images. In contrast to [22, 20] we neither use a different parameter set for each test dataset [22] nor do we use one of the test datasets during training [20]. More precisely, we train on MNIST, CIFAR10 and SVHN, where we use the LeNet architecture on MNIST taken from [23] and a ResNet architecture [13] for CIFAR10 and SVHN. For the generation of out-of-distribution images from p_{out} we proceed as follows: half of the images are generated by randomly permuting pixels of images from

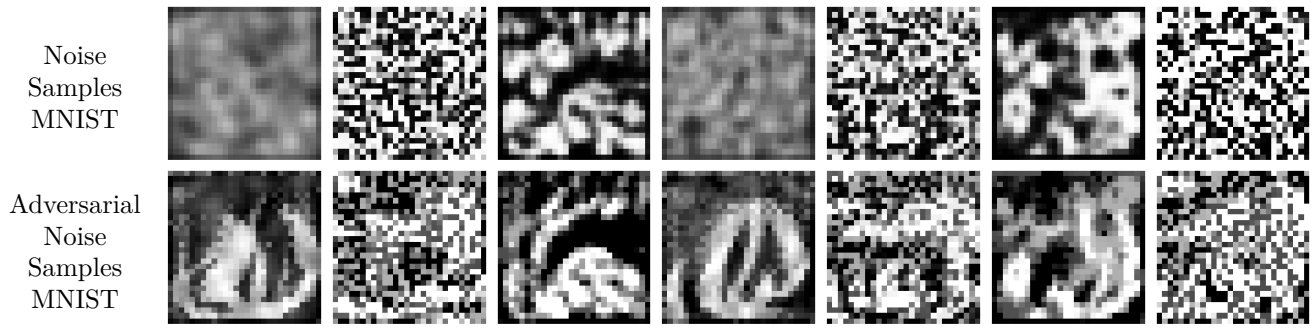

Figure 2: Top Row: Our generated noise images based on uniform noise resp. permuted MNIST together with a Gaussian filter and contrast rescaling. Bottom row: for each noise image from above we generate the corresponding adversarial noise image using PGD with 20 iterations maximizing the second part of the loss in ACET. Note that neither in the noise images nor in the adversarially modified ones there is structure similar to a MNIST image.

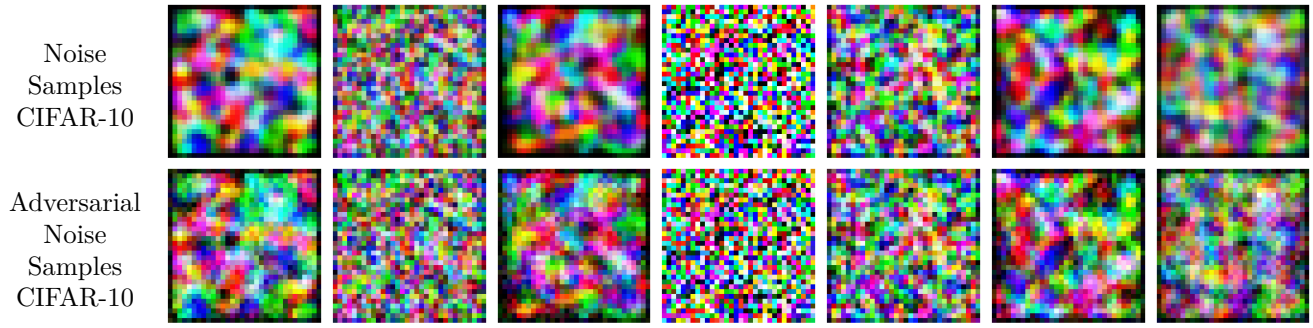

Figure 3: Top row: our generated color noise images having low-and highfrequency noise (similar to Figure 2). Bottom row: the corresponding adversarial images. Neither the noise nor the adversarial noise images show similarity to CIFAR10 images.

the training set and half of the images are generated uniformly at random. Then we apply to these images a Gaussian filter with parameter $\sigma \in [0.25, 1.25]$ to have both high frequency as well as low frequency noise. As the Gaussian filter leads to a contrast reduction we apply afterwards a contrast rescaling by applying to all channels x the sigmoid function

$$f(x) = \frac{1}{1 + e^{-\gamma(x-0.5)}},$$

where $\gamma \in [5, 30]$. The parameters σ and γ are uniformly drawn at random per batch. **Training:** We train each model normally (plain), with confidence enhancing data augmentation (CEDA) and with adversarial confidence enhancing training (ACET). It is well known that weight decay alone reduces overconfident predictions. Thus we use weight decay with regularization parameter $5 * 10^{-4}$ for all models leading to a strong baseline (plain). For CEDA (10) and ACET (13) we both use $\lambda = 1$, that means 50% of the samples in each batch are from the original training set and 50% are noise samples as described before. For ACET we use $p = \infty$ and PGD [23] with 20 iterations and $\epsilon = 0.1$ for CIFAR10 and SVHN and $\epsilon = 0.3$ for MNIST. All models are trained for 120 epochs with ADAM [16] on MNIST and SGD+momentum for CIFAR10/SVHN. The initial learning rate is 10^{-3} for MNIST and 0.1 for CIFAR10/SVHN and it is reduced by a factor of 10 at 65% and 95% of the total number of epochs. More results and details can be found in the appendix.

Evaluation: We report for each model (plain, CEDA, ACET) the test error and the mean maximal confidence (for each point this is $\max_{k=1,\dots,K} \frac{e^{f_k(x)}}{\sum_{l=1}^{K} e^{f_l(x)}}$), denoted as MMC, on the test set. In order to evaluate how well we reduce the confidence on the out-distribution, we use two datasets on CIFAR10 [17] and SVHN [27] (namely among CIFAR10, SVHN and the classroom subset of LSUN [36] we use the one on which we have *not* trained) and for MNIST we evaluate on EMNIST [7], a grayscale version of CIFAR10 and Fashion MNIST [35]. Additionally, we use the evaluation on noise. The noise is generated in the same way as the noise we use for training as well as adversarial noise, where we maximize the loss (note that we use a uniform distribution over the classes for the noise image) with 40 iterations PGD (note that for training we

Trained on MNIST	Plain (TE: **0.53%**)			CEDA (TE: 0.82%)			ACET (TE: 0.69%)		
	MMC	AUROC	FPR@95	MMC	AUROC	FPR@95	MMC	AUROC	FPR@95
MNIST	**0.990**	–	–	0.987	–	–	0.984	–	–
FMNIST	0.655	0.978	0.116	0.517	0.975	0.104	**0.211**	**0.996**	**0.0257**
EMNIST	0.821	0.881	0.369	0.790	0.886	0.361	**0.751**	**0.908**	**0.332**
grayCIFAR10	0.496	0.995	0.005	0.138	**1.000**	**0.000**	**0.100**	**1.000**	**0.000**
Noise	0.609	0.910	0.301	0.116	**1.000**	**0.000**	**0.100**	**1.000**	**0.000**
Adv. Noise	0.999	0.220	1.000	0.811	0.854	0.604	**0.104**	**1.000**	**0.000**
Adv. Samples	0.997	0.421	0.984	0.992	0.511	0.974	**0.256**	**0.961**	**0.134**
Trained on CIFAR-10	Plain (TE: 8.78%)			CEDA (TE: **8.24%**)			ACET (TE: 8.51%)		
	MMC	AUROC	FPR@95	MMC	AUROC	FPR@95	MMC	AUROC	FPR@95
CIFAR-10	**0.949**	–	–	0.946	–	–	**0.949**	–	–
SVHN	0.755	0.879	0.698	**0.685**	**0.910**	**0.588**	0.737	0.888	0.670
LSUN	**0.709**	**0.889**	**0.611**	0.729	0.865	0.652	0.725	0.880	0.634
Noise	0.791	0.763	0.716	0.277	0.976	0.151	**0.015**	**0.995**	**0.034**
Adv. Noise	1.000	0.036	1.000	1.000	0.050	1.000	**0.321**	**0.866**	**0.217**
Adv. Samples	1.000	0.036	1.000	1.000	0.016	1.000	1.000	0.021	1.000
Trained on SVHN	Plain (TE: 3.47%)			CEDA (TE: **3.45%**)			ACET (TE: 3.63%)		
	MMC	AUROC	FPR@95	MMC	AUROC	FPR@95	MMC	AUROC	FPR@95
SVHN	**0.980**	–	–	0.978	–	–	**0.980**	–	–
CIFAR-10	0.698	0.954	0.276	**0.484**	**0.978**	**0.117**	0.570	0.970	0.171
LSUN	0.686	0.959	0.251	**0.293**	**0.993**	**0.031**	0.411	0.988	0.055
Noise	0.648	0.961	0.240	0.145	0.998	0.011	**0.124**	**0.999**	**0.005**
Adv. Noise	0.994	0.004	0.971	0.992	0.001	0.986	**0.229**	**0.962**	**0.086**
Adv. Samples	1.000	0.052	1.000	1.000	0.057	1.000	**0.904**	**0.252**	**0.864**

Table 1: We train on three datasets (MNIST, CIFAR10 and SVHN) three models, plain, CEDA and ACET, and then evaluate them on out-of-distribution samples (other datasets, noise, adversarial noise and adversarial samples built from the test set on which was trained). We report training error of all models and show the mean maximum confidence (MMC) on the in- and out-distribution samples (smaller is better for out-distribution samples), the AUC of ROC curve (AUROC) for the separation of in- and out-distribution based on confidence value (higher is better) and the FPR at 95% true positive rate for the same problem (smaller is better).

	Plain			CEDA			ACET		
	MNIST	CIFAR-10	SVHN	MNIST	CIFAR-10	SVHN	MNIST	CIFAR-10	SVHN
Median α	3.58	5.16	8.92	850.56	15.41	284.85	7583.70	55.21	18.49

Table 2: We evaluate all trained models on inputs scaled by a constant $\alpha \geq 1$ (note that the resulting inputs will not constitute valid images anymore, since in most cases they exceed the $[0,1]^d$ box). We find the minimum α such that the models output 99.9% confidence on them, and report the median over 10 000 examples. As predicted by Theorem 3.1 we observe that it is always possible to obtain overconfident predictions just by scaling inputs by some constant α, and for plain models this constant is smaller than for our CEDA and ACET.

use 20 iterations) in the ϵ ball wrt the $\|\cdot\|_\infty$-norm with ϵ equal to the one used during training for 40 iterations. Finally, we check also the confidence on adversarial samples computed for the test set of the in distribution dataset using 40 PGD iterations with the same ϵ as used for ACET. The latter two evaluation modalities are novel compared to [15, 22, 20]. The adversarial noise is interesting as it shows how bad one can still get in a neighborhood of a noise image and it potentially detects an over-adaptation to the noise model used during training in particular in CEDA. The evaluation on adversarial samples is interesting as one can hope that the reduction of the confidence for out-of-distribution images also reduces the confidence of adversarial samples as typically adversarial samples are off the data manifold and thus are also out-of-distribution samples (even though their distance to the true distribution is small). As evaluation criteria we use the average confidence, the area under the ROC curve (AUC) where we use the confidence as a threshold for the detection problem (in-distribution vs. out-distribution). Moreover, we report in the same

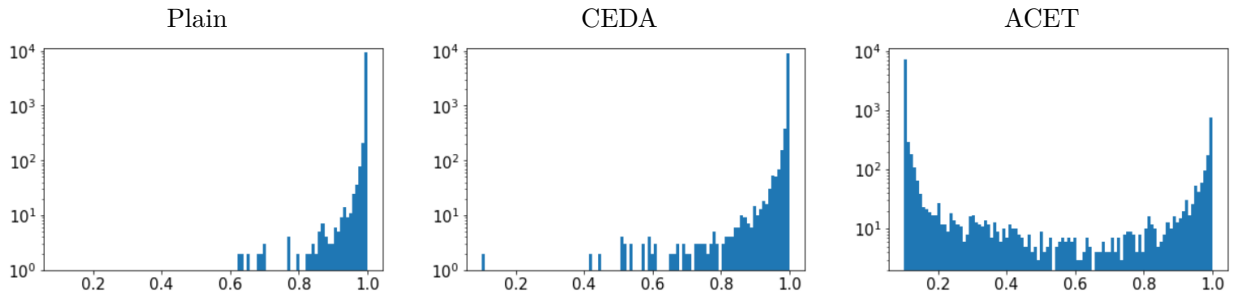

Figure 4: Histogram of confidence values (logarithmic scale) of adversarial samples for MNIST test points. ACET is the only model where most adversarial samples have very low confidence. Note, however that the ACET model has not been trained on adversarial samples of MNIST, but only on adversarial noise.

setting the false positive rate (FPR) when the true positive rate (TPR) is fixed to 95%. All results can be found in Table 1.

Main Results: We show in Table 1 the results of plain (normal training), CEDA and ACET. First of all, we observe that there is almost no difference between the test errors of all three methods. Thus improving the confidence far away from the training data does not impair the generalization performance. We also see that the plain models always produce high confidence predictions on noise images and completely fail on adversarial noise. CEDA produces low confidence on noise images but fails on adversarial noise which was to be expected as similar findings have been made for the creation of adversarial samples. Only ACET produces always low confidence predictions on adversarial noise and has high AUROC. For the out-of-distribution datasets CEDA and ACET improve most of the time the maximum confidence and the AUROC, sometimes with very strong improvements like on MNIST evaluated on F-MNIST or SVHN evaluated on LSUN. However, one observes that it is more difficult to reduce the confidence for related tasks e.g. MNIST evaluated on EMNIST or CIFAR10 evaluated on LSUN, where the image structure is more similar. Finally, an interesting outcome is that ACET reduces the confidence on adversarial examples, see Figure 4 for an illustration, and achieves on MNIST very high AUROC values so that one can detect adversarial examples via thresholding the confidence. Obviously plain and CEDA fail on this task. The good performance of ACET is to some extent unexpected as we just bias the model towards uniform confidence far away from the training data, but adversarial examples are still close to the original image. In summary, ACET does improve confidence estimates significantly compared to the plain but also compared to CEDA, in particular on adversarial noise and adversarial examples. In particular, its very good effect also on adversarial examples is an interesting side effect and shows in our opinion that the models have become more reliable.

Far away high confidence predictions: Theorem 3.1 states that ReLU networks always attain high confidence predictions far away from the training data. The two network architectures used in this paper are ReLU networks. It is thus interesting to investigate if the confidence-enhanced training CEDA or ACET makes it harder to reach high confidence than for the plain model. We do the following experiment: we take random noise images x and then search for the smallest α such that the classifier attains 99.9% confidence on αx. This is exactly the construction from Theorem 3.1 and the result can be found in Table 2.

We observe in Table 2 that indeed the required upscaling factor α is significantly higher for CEDA and ACET than for the plain model which implies that our models also influence the network far away from the training data. Moreover, this also shows that even training methods explicitly aiming at counteracting the phenomenon of high confidence predictions far away from the training data, cannot prevent this.

6. Conclusion

We have shown in this paper that the problem of arbitrarily high confidence predictions of ReLU networks far away from the training data cannot be avoided even with modifications like temperature rescaling [12]. On the other hand we have shown that CEDA and in particular ACET are a good way to reach much better confidence estimates for image data. CEDA and ACET can be directly used for any model with little implementation overhead. For the future it would be desirable to have network structures which have provably the property that far away from the training data the confidence is uniform over the classes: the network knows when it does not know.

References

[1] R. Arora, A. Basuy, P. Mianjyz, and A. Mukherjee. Understanding deep neural networks with rectified linear unit. In *ICLR*, 2018. 2

[2] A. Athalye, N. Carlini, and D. Wagner. Obfuscated gradients give a false sense of security: Circumventing defenses to adversarial examples. In *ICML*, 2018. 1

[3] P. Bartlett and M. H. Wegkamp. Classification with a reject option using a hinge loss. *JMLR*, 9:1823–1840, 2008. 4

[4] A. Bendale and T. Boult. Towards open set deep networks. In *CVPR*, 2016. 1

[5] N. Carlini and D. Wagner. Adversarial examples are not easily detected: Bypassing ten detection methods. In *ACM Workshop on Artificial Intelligence and Security*, 2017. 1

[6] N. Carlini and D. Wagner. Towards evaluating the robustness of neural networks. In *IEEE Symposium on Security and Privacy*, 2017. 1

[7] G. Cohen, S. Afshar, J. Tapson, and A. van Schaik. Emnist: an extension of mnist to handwritten letters. preprint, arXiv:1702.05373v2, 2017. 6

[8] F. Croce and M. Hein. A randomized gradient-free attack on relu networks. In *GCPR*, 2018. 2

[9] T. DeVries and G. W. Taylor. Learning confidence for out-of-distribution detection in neural networks. preprint, arXiv:1802.04865v1, 2018. 1

[10] Y. Gal and Z. Ghahramani. Dropout as a bayesian approximation: Representing model uncertainty in deep learning. In *ICML*, 2016. 1

[11] I. J. Goodfellow, J. Shlens, and C. Szegedy. Explaining and harnessing adversarial examples. In *ICLR*, 2015. 1, 4, 5

[12] C. Guo, G. Pleiss, Y. Sun, and K. Weinberger. On calibration of modern neural networks. In *ICML*, 2017. 1, 8

[13] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016. 5

[14] M. Hein and M. Andriushchenko. Formal guarantees on the robustness of a classifier against adversarial manipulation. In *NIPS*, 2017. 1

[15] D. Hendrycks and K. Gimpel. A baseline for detecting misclassified and out-of-distribution examples in neural networks. In *ICLR*, 2017. 1, 5, 7

[16] D. P. Kingma and J. Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014. 6

[17] A. Krizhevsky. Learning multiple layers of features from tiny images. technical report, 2009. 6

[18] B. Lakshminarayanan, A. Pritzel, and C. Blundell. Simple and scalable predictive uncertainty estimation using deep ensembles. In *NIPS*, 2017. 1

[19] M. Lapin, M. Hein, and B. Schiele. Loss functions for top-k error: Analysis and insights. In *CVPR*, 2016. 4

[20] K. Lee, H. Lee, K. Lee, and J. Shin. Training confidence-calibrated classifiers for detecting out-of-distribution samples. In *ICLR*, 2018. 1, 5, 7

[21] C. Leibig, V. Allken, M. S. Ayhan, P. Berens, and S. Wahl. Leveraging uncertainty information from deep neural networks for disease detection. *Scientific Reports*, 7, 2017. 1

[22] S. Liang, Y. Li, and R. Srikant. Enhancing the reliability of out-of-distribution image detection in neural networks. In *ICLR*, 2018. 1, 4, 5, 7

[23] A. Madry, A. Makelov, L. Schmidt, D. Tsipras, and A. Valdu. Towards deep learning models resistant to adversarial attacks. In *ICLR*, 2018. 1, 2, 5, 6

[24] M. Mirman, T. Gehr, and M. Vechev. Differentiable abstract interpretation for provably robust neural networks. In *ICML*, 2018. 1

[25] G. Montufar, R. Pascanu, K. Cho, and Y. Bengio. On the number of linear regions of deep neural networks. In *NIPS*, 2014. 2

[26] E. Nalisnick, A. Matsukawa, Y. Whye Teh, D. Gorur, and B. Lakshminarayanan. Do deep generative models know what they don't know? preprint, arXiv:1810.09136v1, 2018. 1, 5

[27] Y. Netzer, T. Wang, A. Coates, A. Bissacco, B. Wu, and A. Y. Ng. Reading digits in natural images with unsupervised feature learning. In *NIPS Workshop on Deep Learning and Unsupervised Feature Learning*, 2011. 6

[28] A. Nguyen, J. Yosinski, and J. Clune. Deep neural networks are easily fooled: High confidence predictions for unrecognizable images. In *CVPR*, 2015. 1

[29] A. Raghunathan, J. Steinhardt, and P. Liang. Certified defenses against adversarial examples. In *ICLR*, 2018. 1

[30] P. F. S.-M. Moosavi-Dezfooli, A. Fawzi. Deepfool: a simple and accurate method to fool deep neural networks. In *CVPR*, pages 2574–2582, 2016. 1

[31] C. Szegedy, W. Zaremba, I. Sutskever, J. Bruna, D. Erhan, I. Goodfellow, and R. Fergus. Intriguing properties of neural networks. In *ICLR*, pages 2503–2511, 2014. 1, 5

[32] A. Tewari and P. Bartlett. On the consistency of multiclass classification methods. *Journal of Machine Learning Research*, 8:1007–1025, 2007. 1

[33] W. Wang, A. Wang, A. Tamar, X. Chen, and P. Abbeel. Safer classification by synthesis. preprint, arXiv:1711.08534v2, 2018. 1

[34] E. Wong and J. Z. Kolter. Provable defenses against adversarial examples via the convex outer adversarial polytope. In *ICML*, 2018. 1

[35] H. Xiao, K. Rasul, and R. Vollgraf. Fashion-MNIST: a novel image dataset for benchmarking machine learning algorithms. preprint, arXiv:1708.07747, 2017. 6

[36] F. Yu, A. Seff, Y. Zhang, S. Song, T. Funkhouser, and J. Xiao. Lsun: Construction of a large-scale image dataset using deep learning with humans in the loop. preprint, arXiv:1506.03365v3, 2015. 6

Appendix

7. Noise Images

In the main paper in Figure 2 and Figure 3 we show samples of the generated noise together with adversarial noise images for the uniform noise. Here we show also examples of the permutation noise, generated by taking permutations of random original images and then applying the Gaussian filter and the contrast-enhancing. Below we show this type of noise images for MNIST and CIFAR10 analogously to Figure 2 resp. Figure 3 in the paper.

7.1. Permutation noise from MNIST images (complements Figure 2 of main paper)

The examples of permutation noise and the adversarial permutation noise for MNIST can be found in Figure 5.

7.2. Permutation noise from CIFAR-10 images (complements Figure 3 of main paper)

The examples of permutation noise and the adversarial permutation noise for CIFAR-10 can be found in Figure 6.

8. The effect of Adversarial Confidence Enhanced Training

In this section we compare predictions of the plain model trained on MNIST (Figure 7) and the model trained with ACET (Figure 8). We analyze the images that receive the lowest maximum confidence on the original dataset (MNIST), and the highest maximum confidence on the two datasets that were used for evaluation (EMNIST, grayCIFAR-10).

Evaluated on MNIST: We observe that for both models the lowest maximum confidence corresponds to hard input images that are either discontinous, rotated or simply ambiguous.

Evaluated on EMNIST: Note that some handwritten letters from EMNIST, e.g. 'o' and 'i' may look exactly the same as digits '0' and '1'. Therefore, one should not expect that an ideal model assigns uniform confidences to all EMNIST images. For Figure 7 and Figure 8 we consider predictions on letters that in general do not look exactly like digits ('a', 'b', 'c', 'd'). We observe that the images with the highest maximum confidence correspond to the handwritten letters that *resemble* digits, so the predictions of both models are justified.

Evaluated on Grayscale CIFAR-10: This dataset consists of the images that are clearly distinct from digits. Thus, one can expect uniform confidences on such images, which is achieved by the ACET model (Table 1), but not with the plain model. The mean maximum confidence of the ACET model is close to 10%, with several individual images that are scored with up to 29.60% confidence. Note, that this is much better than for the plain model, which assigns up to 99.56% confidence for the images that have nothing to do with digits. This result is particularly interesting, since the ACET model has not been trained on grayCIFAR-10 examples, and yet it shows much better confidence calibration for out-of-distribution samples.

9. ROC curves

We show the ROC curves for the binary classification task of separating *True* (in-distribution) images from *False* (out-distribution) images. These correspond to the AUROC values (area under the ROC curve) reported in Table 1 in the main paper. As stated in the paper the separation of in-distribution from out-distribution is done by thresholding the maximal confidence value over all classes taken from the original multi-class problem. Note that the ROC curve shows on the vertical axis the True Positive Rate (TPR), and the horizontal axis is the False Positive Rate (FPR). Thus the FPR@95%TPR value can be directly read off from the ROC curve as the FPR value achieved for 0.95 TPR. Note that a value of 1 of AUROC corresponds to a perfect classifier. A value below 0.5 means that the ordering is reversed: out-distribution images achieve on average higher confidence than the in-distribution images. The worst case is an AUROC of zero, in which case all out-distribution images achieve a higher confidence value than the in-distribution images.

ROC curves for the models trained on MNIST

In the ROC curves for the plain, CEDA and ACET models for MNIST that are presented in Figure 9, the different grades of improvements for the six evaluation datasets can be observed. For noise, in the plain model the curve is quite far away from the upper left corner, while for the models trained with CEDA and ACET, it reaches that corner, which is the ideal case. For adversarial noise, the plain model is worse than a random classifier, which manifests itself in the fact that the ROC curve runs below the diagonal. While CEDA is better, ACET achieves the ideal result here as well.

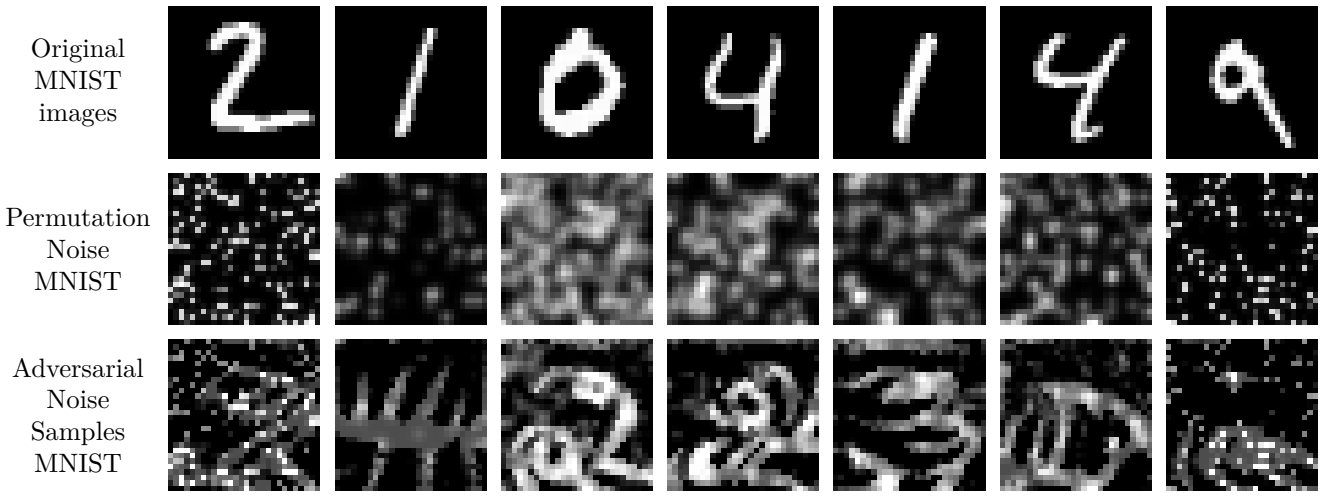

Figure 5: Top Row: Original MNIST images. Middle Row: The pixels of the original image are permuted and blurred. Its pixel value distribution is closer to that of the original dataset rather than the uniform noise. Bottom Row: Adversarial samples generated from the shown permutation noise for the CEDA model, with a difference of up to 0.3 in the ∞-norm to the middle row.

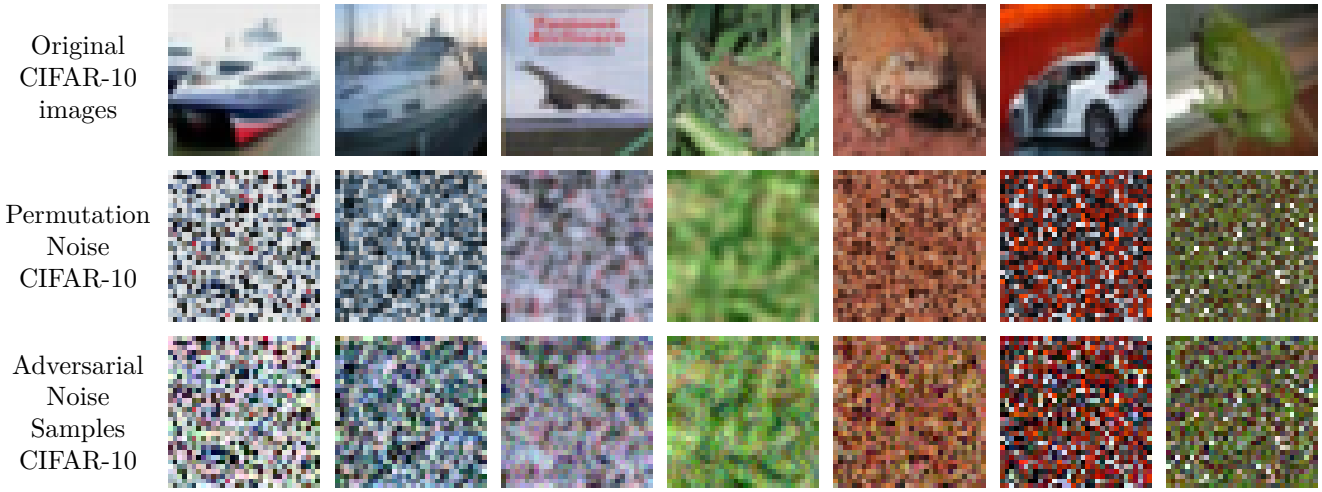

Figure 6: Top Row: Original CIFAR-10 images. Middle Row: Permutation noise generated from those images. Bottom Row: Corresponding adversarial noise samples for the CEDA model, with a difference of up to 0.1 in the ∞-norm to the middle row.

ROC curves for the models trained on CIFAR-10

The ROC curves for CIFAR10 show that this dataset is harder than MNIST or SVHN. While CEDA and ACET improve on SVHN, the difference is small. For LSUN even plain training is slightly better (only time for all three datasets). However, on noise and adversarial noise ACET outperforms all other methods.

ROC curves for the models trained on SVHN

CEDA and ACET outperform significantly plain training. While CEDA and ACET perform similar on CIFAR-10, LSUN and noise, ACET outperforms CEDA clearly on adversarial noise and adversarial samples.

10. Histograms of confidence values

As the AUROC or the FPR@95%TPR just tell us how well the confidence values of in-distribution and out-distribution are ordered, we also report the histograms of achieved confidence values on the original dataset (in-distribution) on which it was trained and the different evaluation datasets. The histograms show how many times the maximum confidence for an image had a certain value between 0.1 (minimal possible value) and 1.0 (maximal possible value), for the test set. They give a more detailed picture than the single numbers for mean maximum confidence, area un-

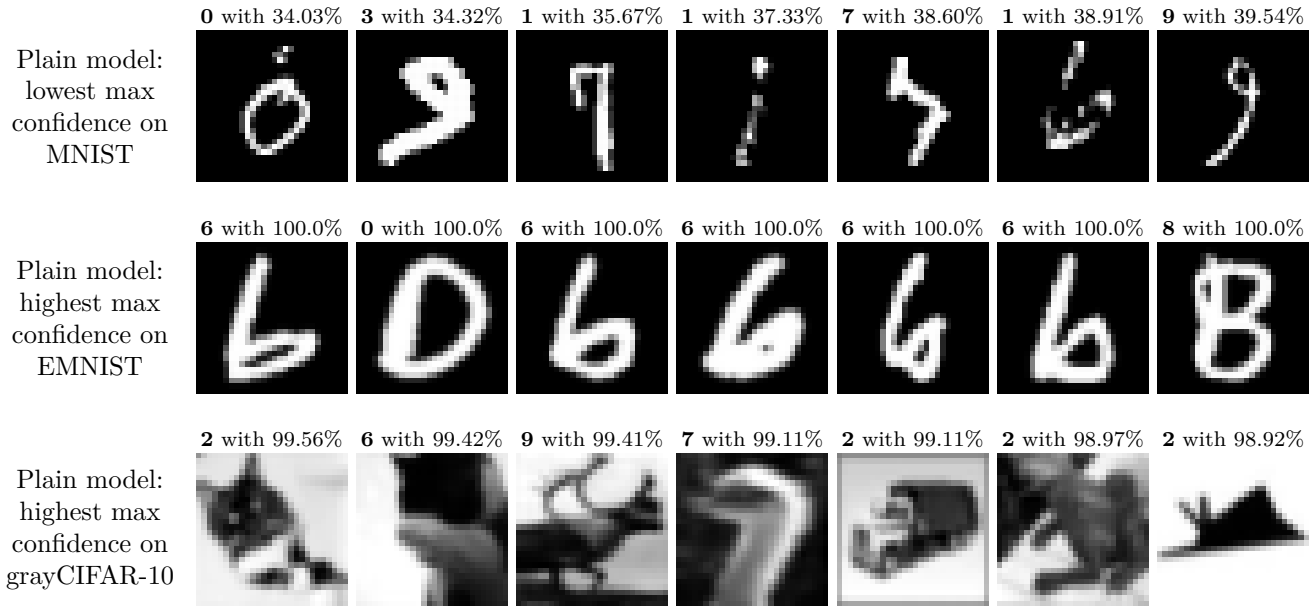

Figure 7: Top Row: predictions of the plain MNIST model with the lowest maximum confidence. Middle Row: predictions of the plain MNIST model on letters 'a', 'b', 'c', 'd' of EMNIST with the highest maximum confidence. Bottom Row: predictions of the plain MNIST model on the grayscale version of CIFAR-10 with the highest maximum confidence. Note that although the predictions on EMNIST are mostly justified, the predictions on CIFAR-10 are overconfident on the images that have no resemblance to digits.

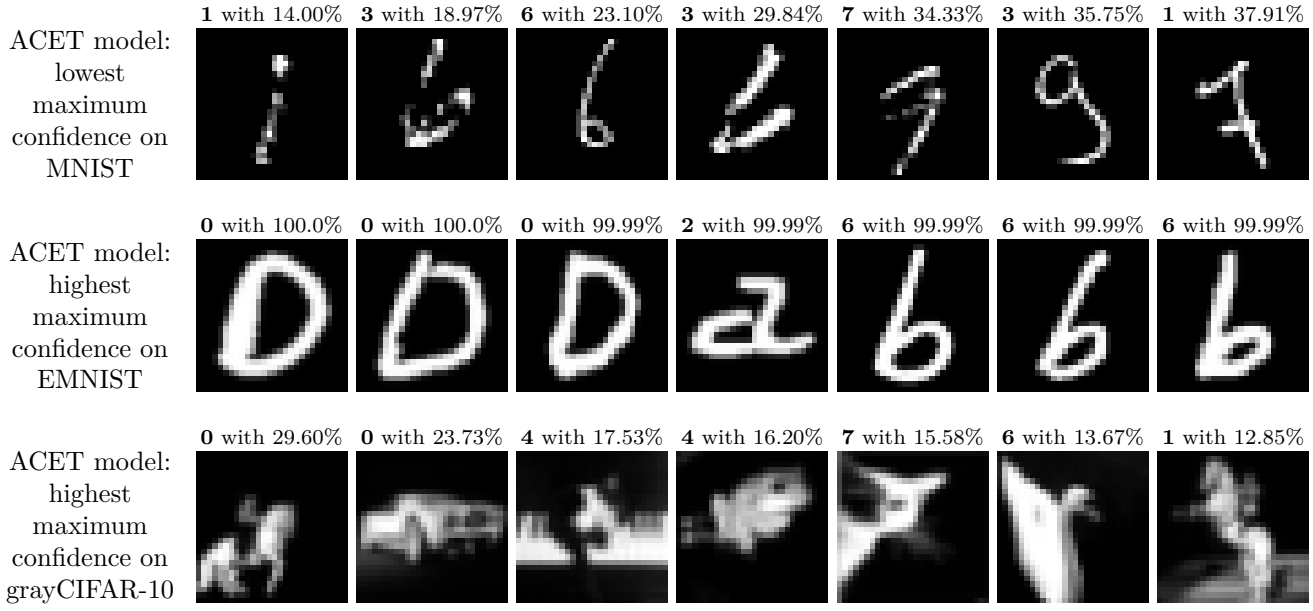

Figure 8: Top Row: predictions of the ACET MNIST model with the lowest maximum confidence. Middle Row: predictions of the ACET MNIST model on letters 'a', 'b', 'c', 'd' of EMNIST with the highest maximum confidence. Bottom Row: predictions of the ACET MNIST model on the grayscale version of CIFAR-10 with the highest maximum confidence. Note that for the ACET model the predictions on both EMNIST and grayCIFAR-10 are now justified.

der ROC and FPR@95% TPR. As visible in the top row, the confidence values for clean MNIST test images don't change significantly for CEDA resp. ACET. For FMNIST, gray CIFAR-10 and Noise inputs, the maximum confidences of CEDA are generally shifted to lower values, and those of ACET even more so. For EMNIST, the same effect is observable, though much weaker. For adversatial noise, CEDA is partially successful in lowering the confidences, with most predictions still above 80% confidence, while the vast majority of ACET confidences is close to the optimal value of 10%. As discussed in the main paper, CEDA is

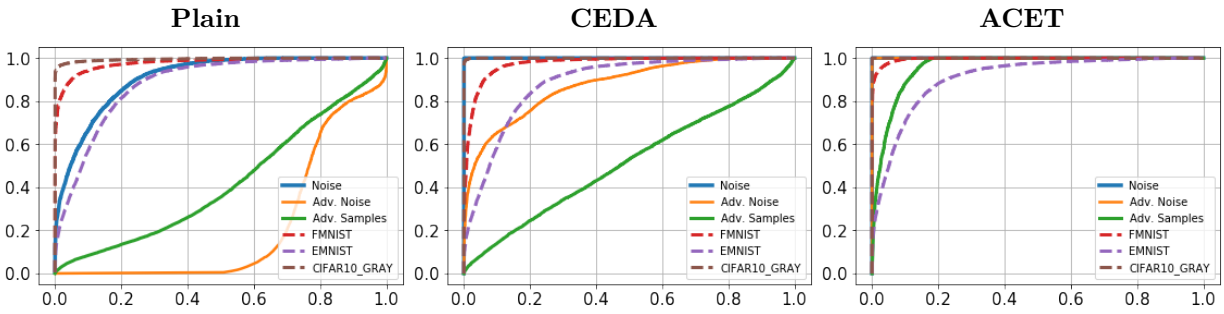

Figure 9: ROC curves of the MNIST models on the evaluation datasets.

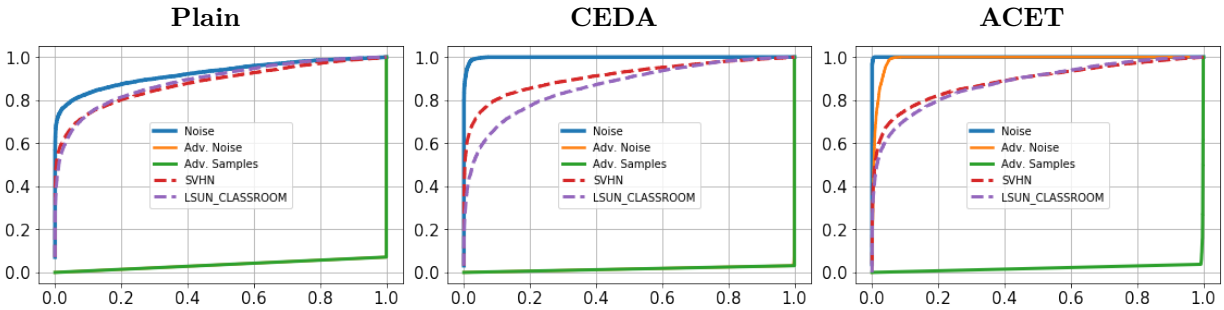

Figure 10: ROC curves of the CIFAR-10 models on the evaluation datasets.

not very beneficial for adversarial images, while ACET lowers its confidence to an average value of 26% here.

10.1. Histograms of confidence values for models trained on MNIST

As visible in the top row of Figure 12, the confidence values for clean MNIST test images don't change significantly for CEDA resp. ACET. For FMNIST, gray CIFAR-10 and Noise inputs, the maximum confidences of CEDA are generally shifted to lower values, and those of ACET even more so. For EMNIST, the same effect is observable, though much weaker due to the similarity of characters and digits. For adversarial noise, CEDA is partially successful in lowering the confidences, with most predictions still above 80% confidence, while the vast majority of ACET confidences is close to the optimal value of 10%. As discussed in the main paper, CEDA is not very beneficial for adversarial images, while ACET lowers its confidence to an average value of 26% here.

10.2. Histograms of confidence values for models trained on CIFAR-10

In Figure 13, CEDA and ACET lower significantly the confidence on noise, and ACET shows improvement for adversarial noise, which fools the plain and CEDA models completely. For CIFAR-10 all models yield very high confidence values on adversarial images. Compared to MNIST ACET leads only to a very small change.

10.3. Histograms of confidence values for models trained on SVHN

Figure 14 shows that both CEDA and ACET assign lower confidences to the out-of-distribution samples from SVHN house numbers and LSUN classroom examples. CEDA and ACET also improve on noise samples. While a large fraction of adversarial samples/noise still achieve high confidence values, our ACET trained model is the only one that lowers the confidences for adversarial noise and adversarial SVHN samples significantly.

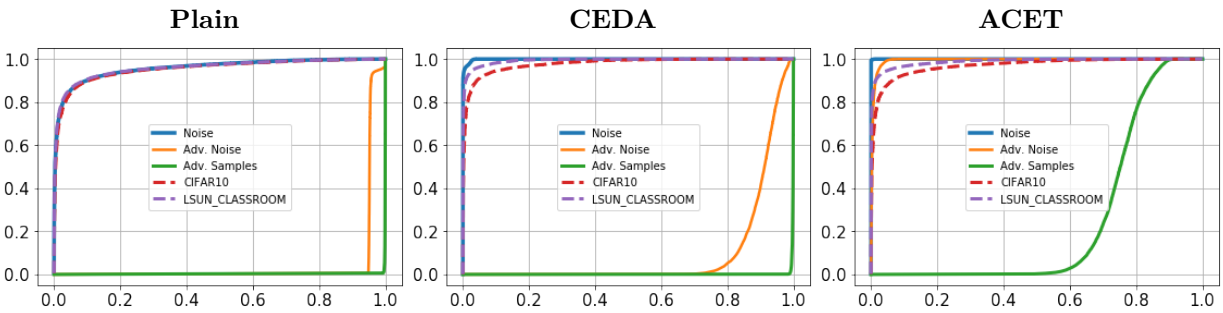

Figure 11: ROC curves of the SVHN models on the evaluation datasets.

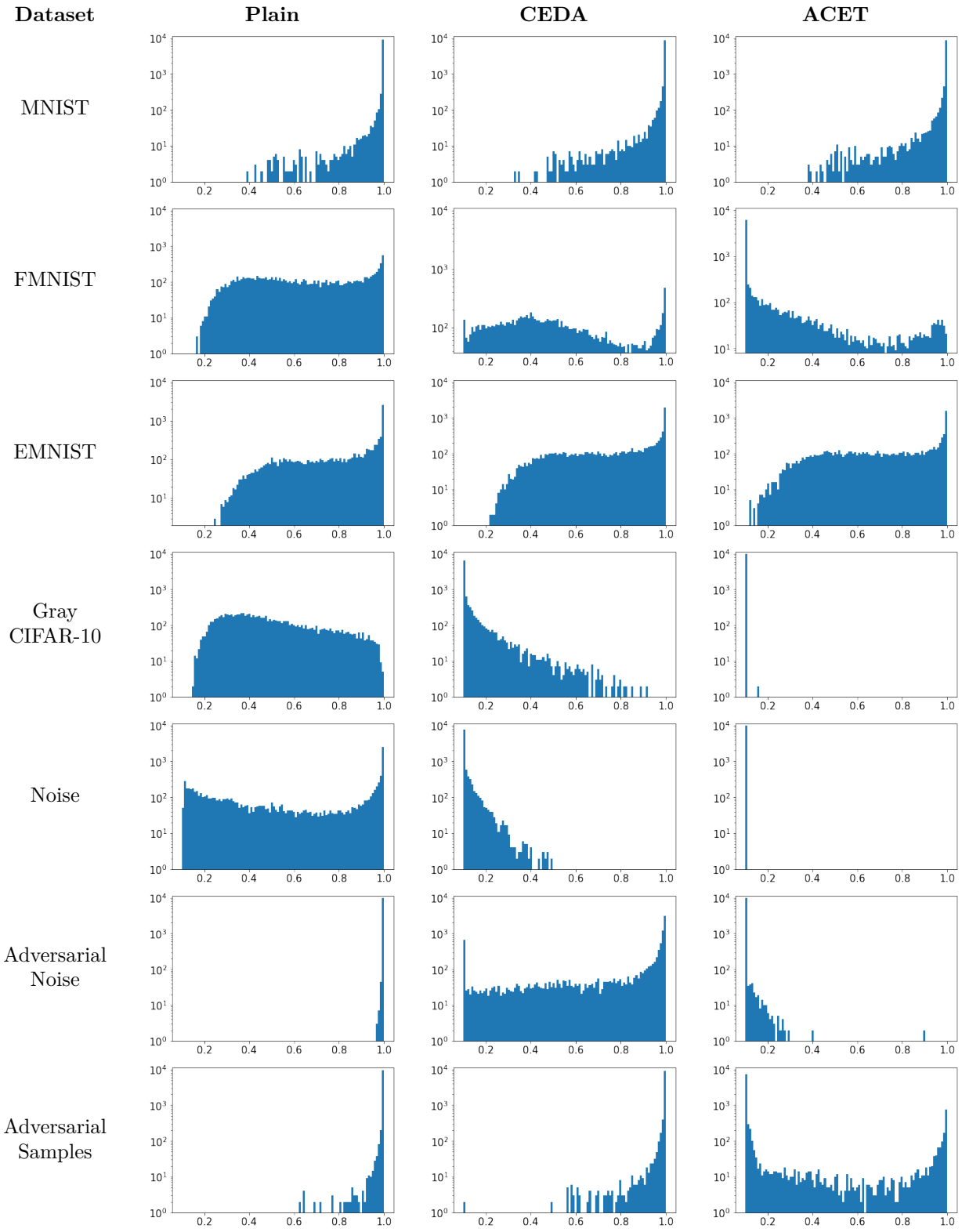

Figure 12: Histograms (logarithmic scale) of maximum confidence values of the three compared models for MNIST on various evaluation datasets.

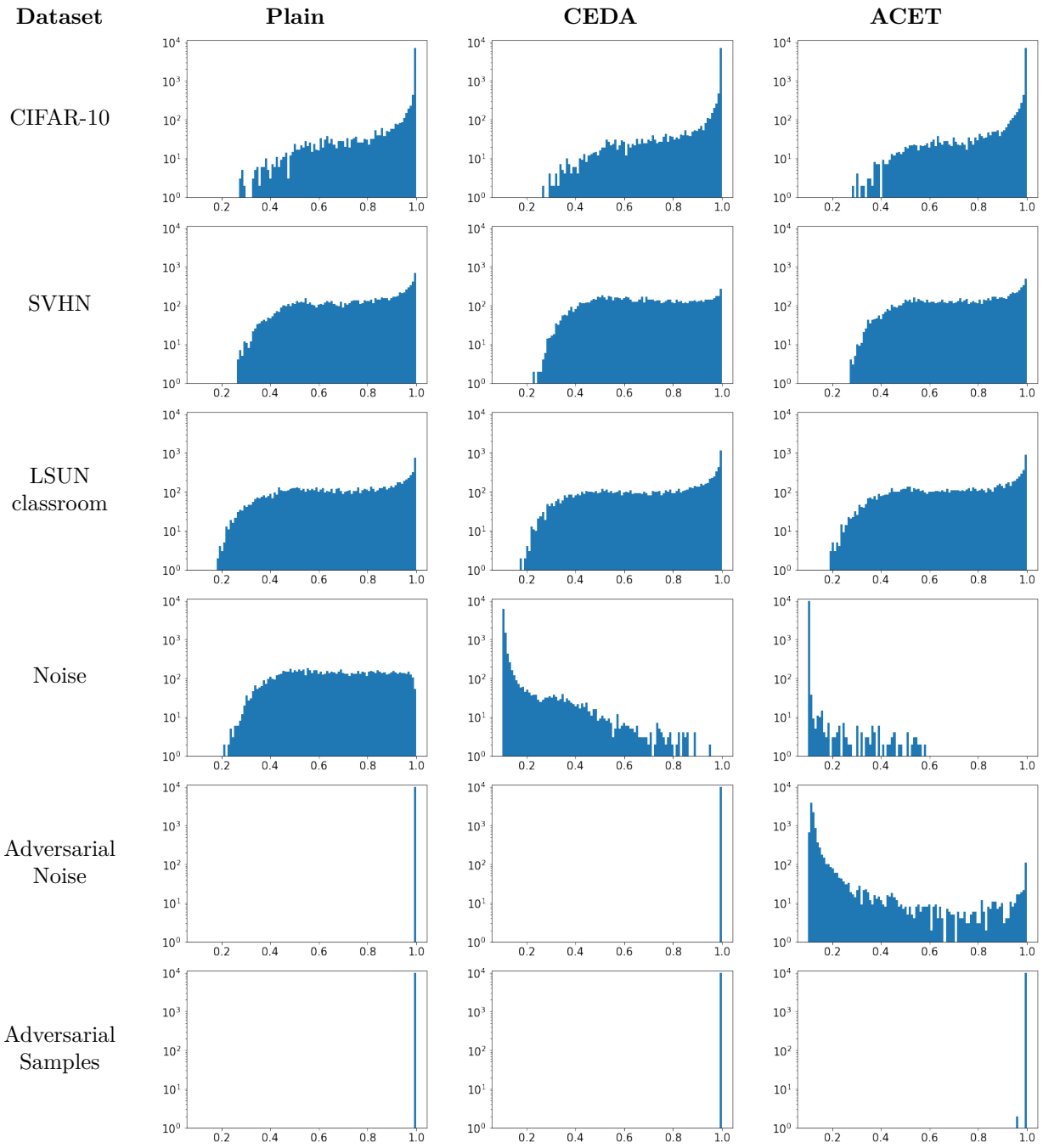

Figure 13: Histograms (logarithmic scale) of maximum confidence values of the three compared models for CIFAR-10 on various evaluated datasets.

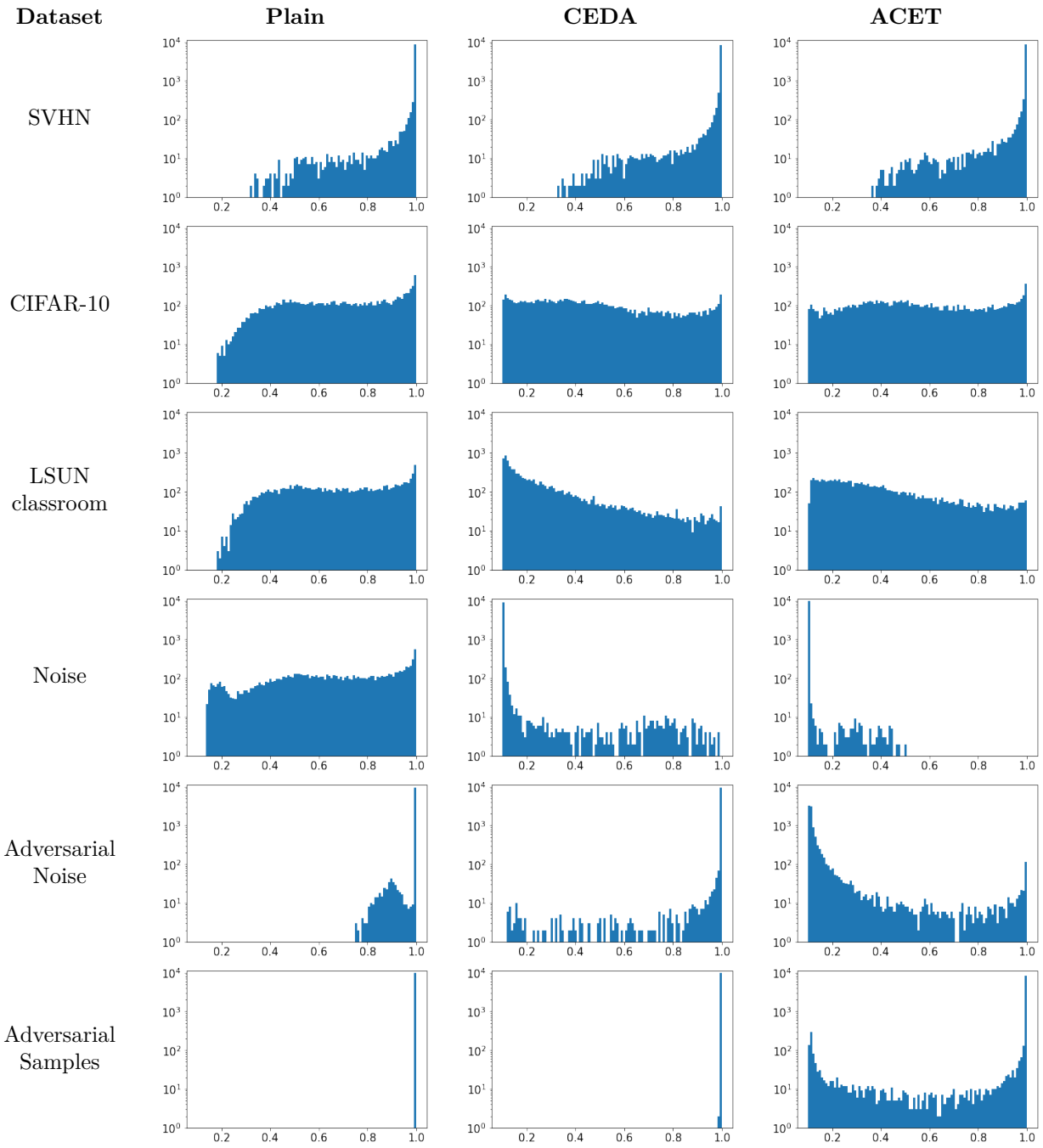

Figure 14: Histograms (logarithmic scale) of maximum confidence values of the three compared models for SVHN on various evaluated datasets.