

---

# Provable Worst Case Guarantees for the Detection of Out-of-Distribution Data

---

**Julian Bitterwolf**  
University of Tübingen

**Alexander Meinke**  
University of Tübingen

**Matthias Hein**  
University of Tübingen

## Abstract

Deep neural networks are known to be overconfident when applied to out-of-distribution (OOD) inputs which clearly do not belong to any class. This is a problem in safety-critical applications since a reliable assessment of the uncertainty of a classifier is a key property, allowing to trigger human intervention or to transfer into a safe state. In this paper, we are aiming for certifiable worst case guarantees for OOD detection by enforcing not only low confidence at the OOD point but also in an  $l_\infty$ -ball around it. For this purpose, we use interval bound propagation (IBP) to upper bound the maximal confidence in the  $l_\infty$ -ball and minimize this upper bound during training time. We show that non-trivial bounds on the confidence for OOD data generalizing beyond the OOD dataset seen at training time are possible. Moreover, in contrast to certified adversarial robustness which typically comes with significant loss in prediction performance, certified guarantees for worst case OOD detection are possible without much loss in accuracy.

## 1 Introduction

Deep neural networks are the state-of-the-art in many application areas. Nevertheless it is still a major concern to use deep learning in safety-critical systems, e.g. medical diagnosis or self-driving cars, since it has been shown that deep learning classifiers suffer from a number of unexpected failure modes, such as low robustness to natural perturbations [12, 17], overconfident predictions [31, 14, 18, 16] as well as adversarial vulnerabilities [36]. For safety critical applications, empirical checks are not sufficient in order to trust a deep learning system in a high-stakes decision. Thus provable guarantees on the behavior of a deep learning system are needed.

One property that one expects from a robust classifier is that it should *not* make highly confident predictions on data that is very different from the training data. However, ReLU networks have been shown to be provably overconfident far away from the training data [16]. This is a big problem as (guaranteed) low confidence of a classifier when it operates out of its training domain can be used to trigger human intervention or to let the system try to achieve a safe state when it “detects” that it is applied outside of its specification. Several approaches to the out-of-distribution (OOD) detection task have been studied [18, 25, 23, 24, 16]. The current state-of-the-art performance of OOD detection in image classification is achieved by enforcing low confidence on a large training set of natural images that is considered as out-distribution [19, 28].

Deep neural networks are also notoriously susceptible to small adversarial perturbations in the input [36, 4] which change the decision of a classifier. Research so far has concentrated on adversarial robustness around the in-distribution. Several empirical defenses have been proposed but many could be broken again [8, 3, 1]. Adversarial training and variations [27, 42] perform well empirically, but typically no robustness guarantees can be given. Certified adversarial robustness has been achieved by explicit computation of robustness certificates [15, 38, 33, 29, 13] and randomized smoothing [6].

Adversarial changes to generate high confidence predictions on the out-distribution have received much less attention although it has been shown early on that they can be used to fool a classifier

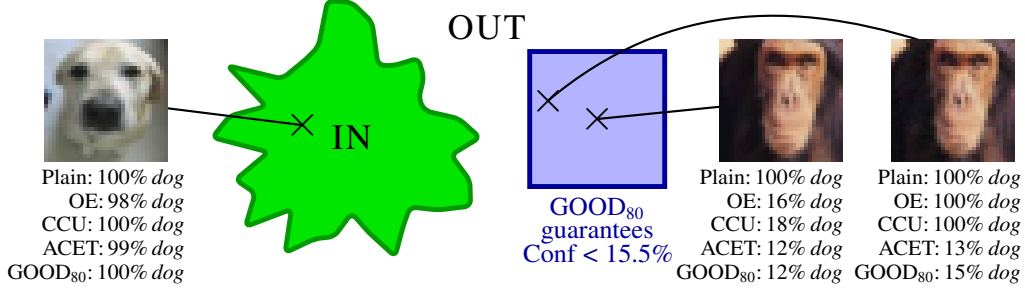


Figure 1: **Overconfident predictions on out-distribution inputs.** **Left:** On the in-distribution CIFAR-10 all methods have similar high confidence on the image of a *dog*. **Middle:** For the out-distribution image of a *chimpanzee* from CIFAR-100 the plain model is overconfident. **Right:** When maximizing the confidence inside the  $l_\infty$ -ball of radius 0.01 around this image (for the OE model), also CCU and OE become overconfident. ACET and our GOOD<sub>80</sub> perform well in having empirical low confidence, but only GOOD<sub>80</sub> guarantees that the confidence in the  $l_\infty$ -ball of radius 0.01 around the middle image is less than 15.5% for any class (note that 10% corresponds to maximal uncertainty).

[31, 34, 35]. Thus, even if a classifier consistently manages to identify samples as not belonging to the in-distribution, it might still assign very high confidence to only marginally perturbed samples from the out-distribution, see Figure ?? . A first empirical defense using a type of adversarial training for OOD detection has been proposed in [16]. However, up to our knowledge in the area of certified out-of-distribution detection the only robustness guarantees for OOD were given in [28], where they use a density estimator for in- and out-distribution and integrate that into the predictive uncertainty of the neural network, which allows them to guarantee that far away from the training data the confidence of the neural network becomes uniform over the classes. Moreover, they can provide worst case guarantees on the confidence on some balls around uniform noise. However, they are not able to provide meaningful guarantees around points which are similar or even close to the in-distribution data and, as we will show, provide only weak guarantees against  $l_\infty$ -adversaries.

In this work we aim to provide worst-case OOD guarantees not only for noise but also for images from related but different image classification tasks. For this purpose we use the techniques from interval bound propagation (IBP) [13] to derive a provable upper bound on the maximal confidence of the classifier in an  $l_\infty$ -ball of radius  $\epsilon$  around a given point. By minimizing this bound on the out-distribution using our training scheme GOOD (Guaranteed Out-Of-distribution Detection) we arrive at the first models which have guaranteed low confidence even on image classification tasks related to the original one; e.g., we get state-of-the-art results on separating letters from EMNIST from digits in MNIST even though the digit classifier has never seen any images of letters at training time. In particular, the guarantees for the training out-distribution generalize to other out-distribution datasets. In contrast to classifiers which have certified adversarial robustness on the in-distribution, GOOD has the desirable property to achieve provable guarantees for OOD detection with almost no loss in accuracy on the in-distribution task even on datasets like CIFAR-10.

## 2 Out-of-distribution detection: setup and baselines

Let  $f : \mathbb{R}^d \rightarrow \mathbb{R}^K$  be a feedforward neural network (DNN) with a last linear layer where  $d$  is the input dimension and  $K$  the number of classes. In all experiments below we use the ReLU activation function. The logits of  $f(x)$  for  $x \in \mathbb{R}^d$  are transformed via the softmax function into a probability distribution  $p(x)$  over the classes with:

$$p_k(x) := \frac{e^{f_k(x)}}{\sum_{l=1}^K e^{f_l(x)}} \text{ for } k = 1, \dots, K. \quad (1)$$

By  $\text{Conf}_f(x) = \max_{k=1, \dots, K} p_k(x)$  we define the confidence of the classifier  $f$  in the prediction  $\arg\max_{k=1, \dots, K} p_k(x)$  at  $x$ .

The general goal of OOD detection is to have low confidence predictions for all inputs  $x$  which are clearly not belonging to the in-distribution task, especially for all inputs lying in a region which has zero probability under the in-distribution. One typical criterion to measure OOD detection

performance is to use  $\text{Conf}_f(x)$  as a feature and compute the AUC of in- versus out-distribution (how well are confidences of in- and out-distribution separated). We discuss a proper conservative measurement of the AUC in case of indistinguishable confidence values, e.g. due to numerical precision, in Appendix C.

As baselines and motivation for our provable approach we use the OOD detection methods Outlier Exposure (OE) [19] and Confidence Enhancing Data Augmentation (CEDA) [16], which use as objective for training

$$\frac{1}{N} \sum_{i=1}^N \mathcal{L}_{\text{CE}}(x_i^{\text{IN}}, y_i^{\text{IN}}) + \frac{\kappa}{M} \sum_{j=1}^M \mathcal{L}_{\text{OUT}}(x_j^{\text{OUT}}), \quad (2)$$

where  $\{(x_i^{\text{IN}}, y_i^{\text{IN}}) \mid 1 \leq i \leq N\}$  is the in-distribution training set,  $\{x_j^{\text{OUT}} \mid 1 \leq j \leq M\}$  the out-distribution training set, and  $\mathcal{L}_{\text{CE}}$  the cross-entropy loss. The hyper-parameter  $\kappa$  determines the relative magnitude of the two loss terms and is most of the time chosen to be one. OE and CEDA differ in the choice of the loss  $\mathcal{L}_{\text{OUT}}$  for the out-distribution where OE uses the cross-entropy loss between  $p(x_j^{\text{OUT}})$  and the uniform distribution and CEDA uses  $\log \text{Conf}_f(x_j^{\text{OUT}})$ . Note that both the CEDA and OE loss attain their global minimum when  $p(x)$  is the uniform distribution. Their difference is typically minor in practice. An important question is the choice of the out-distribution. For general image classification, it makes sense to use an out-distribution which encompasses basically any possible image one could ever see at test time and thus the set of all natural images is a good out-distribution; following [19] we use the 80 Million Tiny Images dataset [37] as a proxy for that.

While OE and CEDA yield state-of-the-art OOD detection performance for image classification tasks when used together with the 80M Tiny Images dataset as out-distribution, they are, similarly to normal classifiers, vulnerable to adversarial manipulation of the out-distribution images where the attack is trying to maximize the confidence in this scenario [28]. Thus [16] proposed Adversarial Confidence Enhanced Training (ACET) which replaces the CEDA loss with  $\max_{\|\hat{x} - x_j^{\text{OUT}}\|_{\infty} \leq \epsilon} \log \text{Conf}_f(\hat{x})$  and can be seen as adversarial training on the out-distribution for an  $l_{\infty}$ -threat model. However, similar to adversarial training on the in-distribution [27] this does not yield any guarantees for out-distribution detection. In the next section we discuss how to use interval-bound-propagation (IBP) to get guaranteed OOD detection performance in a  $l_{\infty}$ -neighborhood of every out-distribution input.

### 3 Provable guarantees for out-of-distribution detection

Our goal is to minimize the confidence of the classifier not only on the out-distribution images themselves but in a whole neighborhood around them. For this purpose, we first derive bounds on the maximal confidence on some  $l_{\infty}$ -ball around a given point. In certified adversarial robustness IBP [13] currently leads to the best guarantees for deterministic classifiers under the  $l_{\infty}$ -threat model. While other methods for deriving guarantees yield tighter bounds [38, 29], they are not easily scalable and, when optimized, the bounds given by IBP have been shown to be very tight [13].

**IBP.** Interval bound propagation [13] provides entrywise lower and upper bounds  $\underline{z}_{k,\epsilon}$  resp.  $\overline{z}_{k,\epsilon}$  for the output  $z_k$  of the  $k$ -th layer of a neural network given that the input  $x$  is varied in the  $l_{\infty}$ -ball of radius  $\epsilon$ . With  $z_0 = x$  and  $\underline{z}_{0,\epsilon} = x - \epsilon \cdot \mathbf{1}$  and  $\overline{z}_{0,\epsilon} = x + \epsilon \cdot \mathbf{1}$  ( $\mathbf{1}$  is the vector of all ones) and  $W_k$  being the weights of the  $k$ -th layer (fully connected, convolutional, residual etc.), one gets upper and lower bounds of the next layers via forward propagation:

$$\begin{aligned} \overline{z}_k^{\epsilon} &= \max(W_k, 0) \cdot \overline{z}_{k-1}^{\epsilon} + \min(W_k, 0) \cdot \underline{z}_{k-1}^{\epsilon} + b_k \\ \underline{z}_k^{\epsilon} &= \min(W_k, 0) \cdot \overline{z}_{k-1}^{\epsilon} + \max(W_k, 0) \cdot \underline{z}_{k-1}^{\epsilon} + b_k, \end{aligned} \quad (3)$$

where the min/max expressions are taken componentwise. The activation function (e.g. ReLU) is directly applied to the bounds. The forward propagation of the bounds is of similar nature as a standard forward pass and back-propagation w.r.t. the weights is relatively straightforward.

**Upper bound on the confidence in terms of the logits.** The log confidence of the model at  $x$  can be written as

$$\log \text{Conf}(x) = \max_{k=1, \dots, K} \log \frac{e^{f_k(x)}}{\sum_{l=1}^K e^{f_l(x)}} = \max_{k=1, \dots, K} -\log \sum_{l=1}^K e^{f_l(x) - f_k(x)}. \quad (4)$$

We assume that the last layer is affine:  $f(x) = W_L \cdot z_{L-1}(x) + b_L$ , where  $L$  is the number of layers of the network. We calculate the upper bounds of all  $K^2$  logit differences as:

$$\begin{aligned}
\max_{\|\hat{x}-x\|_\infty \leq \epsilon} f_k(\hat{x}) - f_l(\hat{x}) &= \max_{\|\hat{x}-x\|_\infty \leq \epsilon} W_{L,k} \cdot z_{L-1}(\hat{x}) + b_{L,k} - W_{L,l} \cdot z_{L-1}(\hat{x}) - b_{L,l} \\
&= \max_{\|\hat{x}-x\|_\infty \leq \epsilon} (W_{L,k} - W_{L,l}) \cdot z_{L-1}(\hat{x}) + b_{L,k} - b_{L,l} \\
&\leq \max(W_{L,k} - W_{L,l}, 0) \cdot \overline{z_{L-1}(x)}^\epsilon \\
&\quad + \min(W_{L,k} - W_{L,l}, 0) \cdot \underline{z_{L-1}(x)}^\epsilon + b_{L,k} - b_{L,l} \\
&=: \overline{f_k(x) - f_l(x)}^\epsilon,
\end{aligned} \tag{5}$$

where  $W_{L,k}$  denotes the  $k$ -th row of  $W_L$  and  $b_{L,k}$  is the  $k$ -th component of  $b_L$ . Note that this upper bound of the logit difference can be negative and is zero for  $l = k$ . Using this upper bound on the logit difference in Equation (4), we obtain an upper bound on the log confidence:

$$\max_{\|\hat{x}-x\|_\infty \leq \epsilon} \log \text{Conf}(\hat{x}) \leq \max_{k=1,\dots,K} -\log \sum_{l=1}^K e^{-\overline{f_k(x) - f_l(x)}^\epsilon} \tag{6}$$

We use the bound in (6) to evaluate the guarantees on the confidences for given out-distribution datasets. However, minimizing it directly during training leads to numerical problems, especially at the beginning of training, when the upper bounds  $\overline{f_k(x) - f_l(x)}^\epsilon$  are very large for  $l \neq k$ , which makes training numerically infeasible. Instead, we rather upper bound the log confidence again by bounding the sum inside the negative log from below with  $K$  times its lowest term:

$$\begin{aligned}
\max_{k=1,\dots,K} -\log \sum_{l=1}^K e^{-\overline{f_k(x) - f_l(x)}^\epsilon} &\leq \max_{k=1,\dots,K} -\log \left( K \cdot \min_{l=1,\dots,K} e^{-\overline{f_k(x) - f_l(x)}^\epsilon} \right) \\
&= \max_{k,l=1,\dots,K} \overline{f_k(x) - f_l(x)}^\epsilon - \log K
\end{aligned} \tag{7}$$

While this bound can considerably differ from the potentially tighter bound of Equation (6), it is often quite close as one term in the sum dominates the others. Moreover, both bounds have the same global minimum when all logits are equal over the  $l_\infty$ -ball. We omit the constant  $\log K$  in the following as it does not matter for training.

The direct minimization of the upper bound in (7) is still difficult, in particular for more challenging in-distribution datasets like SVHN and CIFAR-10, as the bound  $\max_{k,l=1,\dots,K} \overline{f_k(x) - f_l(x)}^\epsilon$  can be several orders of magnitude larger than the in-distribution loss. Therefore, we use the logarithm of this quantity. However, we also want to have a more fine-grained optimization when the upper bound becomes small in the later stage of the training. Thus we define the Confidence Upper Bound loss  $\mathcal{L}_{\text{CUB}}$  for an OOD input as

$$\mathcal{L}_{\text{CUB}}(x; \epsilon) := \log \left( \frac{\left( \max_{k,l=1,\dots,K} \overline{f_k(x) - f_l(x)}^\epsilon \right)^2}{2} + 1 \right). \tag{8}$$

Note that  $\log(\frac{a^2}{2} + 1) \approx \frac{a^2}{2}$  for small  $a$  and thus we achieve the more fine-grained optimization with an  $l_2$ -type of loss in the later stages of training. The overall objective of **fully applied Guaranteed OOD Detection training (GOOD<sub>100</sub>)** is the minimization of

$$\frac{1}{N} \sum_{i=1}^N \mathcal{L}_{\text{CE}}(x_i^{\text{IN}}, y_i^{\text{IN}}) + \frac{\kappa}{M} \sum_{j=1}^M \mathcal{L}_{\text{CUB}}(x_j^{\text{OUT}}; \epsilon), \tag{9}$$

where  $\{(x_i^{\text{IN}}, y_i^{\text{IN}}) \mid 1 \leq i \leq N\}$  is the in-distribution training set and  $\{x_j^{\text{OUT}} \mid 1 \leq j \leq M\}$  the out-distribution. The hyper-parameter  $\kappa$  determines the relative magnitude of the two loss terms. During training we slowly increase this value and  $\epsilon$  in order to further stabilize the training with GOOD.

**Quantile-GOOD: trade-off between clean and guaranteed AUC.** Training models by minimizing (9) means that the classifier gets severely punished if *any* training OOD input receives a high confidence upper bound. If OOD inputs exist to which the classifier already assigns high confidence without even considering the worst case, e.g. as these inputs share features with the in-distribution, it makes little sense to enforce low confidence guarantees. Later in the experiments we show that for difficult tasks like CIFAR-10 this can happen. In such cases the normal AUC for OOD detection gets worse as the high loss of the out-distribution part effectively leads to low confidence on a significant part of the in-distribution which is clearly undesirable.

Hence, for OOD inputs  $x$  which are not clearly distinguishable from the in-distribution, it is preferable to just have the “normal” loss  $\mathcal{L}_{\text{CUB}}(x_j^{\text{OUT}}; 0)$  without considering the worst case. We realize this by enforcing the loss with the guaranteed upper bounds on the confidence just on some quantile of the easier OOD inputs, namely the ones with the lowest guaranteed out-distribution loss  $\mathcal{L}_{\text{CUB}}(x; \epsilon)$ . We first order the OOD training set by the potential loss  $\mathcal{L}_{\text{CUB}}(x; \epsilon)$  of each sample in ascending order  $\pi$ , that is  $\mathcal{L}_{\text{CUB}}(x_{\pi_1}^{\text{OUT}}) \leq \mathcal{L}_{\text{CUB}}(x_{\pi_2}^{\text{OUT}}) \leq \dots \leq \mathcal{L}_{\text{CUB}}(x_{\pi_M}^{\text{OUT}})$ . We then apply the loss  $\mathcal{L}_{\text{CUB}}(x; \epsilon)$  to the lower quantile  $q$  of the points (the ones with the smallest loss  $\mathcal{L}_{\text{CUB}}(x; \epsilon)$ ) and take  $\mathcal{L}_{\text{CUB}}(x; 0)$  for the remaining samples, which means no worst-case guarantees on the confidence are enforced:

$$\frac{1}{N} \sum_{i=1}^N \mathcal{L}_{\text{CE}}(x_i^{\text{IN}}, y_i^{\text{IN}}) + \frac{\kappa}{M} \sum_{j=1}^{\lfloor q \cdot M \rfloor} \mathcal{L}_{\text{CUB}}(x_{\pi_j}^{\text{OUT}}; \epsilon) + \frac{\kappa}{M} \sum_{j=\lfloor q \cdot M \rfloor + 1}^M \mathcal{L}_{\text{CUB}}(x_{\pi_j}^{\text{OUT}}; 0). \quad (10)$$

During training we do this ordering on the part of each batch consisting of out-distribution images. On CIFAR-10, where the out-distribution dataset 80M Tiny Images is closer to the in-distribution, the quantile GOOD-loss allows us to choose the trade-off between clean and guaranteed AUC for OOD detection, similar to the trade-off between clean and robust accuracy in adversarial robustness.

## 4 Experiments

We provide experimental results for image recognition tasks with MNIST [22], SVHN [30] and CIFAR-10 [21] as in-distribution datasets. We first discuss the training details, hyperparameters and evaluation before we present the results of GOOD and competing methods. Code is available under <https://gitlab.com/Bitterwolf/GOOD>.

### 4.1 Model architectures, training procedure and evaluation

**Model architectures and data augmentation.** For all experiments, we use deep convolutional neural networks consisting of convolutional, affine and ReLU layers. For MNIST, we use the large architecture from [13], and for SVHN and CIFAR-10 a similar but deeper and wider model. The layer structure is laid out in Table 2 in the appendix. Data augmentation is applied to both in- and out-distribution images during training. For MNIST we use random crops to size  $28 \times 28$  with padding 4 and for SVHN and CIFAR-10 random crops with padding 4 as well as the quite aggressive augmentation AutoAugment [9]. Additionally, we apply random horizontal flips for CIFAR-10.

**GOOD training procedure.** As it is the case with IBP training [13] for certified adversarial robustness, we have observed that the inclusion of IBP bounds can make the training unstable or cause it to fail completely. This can happen for our GOOD training despite the logarithmic damping in the  $\mathcal{L}_{\text{CUB}}$  loss in (8). Thus, in order to further stabilize the training similar to [13], we use linear ramp up schedules for  $\epsilon$  and  $\kappa$ , which are detailed in Appendix D. As radii for the  $l_\infty$ -perturbation model on the out-distribution we use  $\epsilon = 0.3$  for MNIST and  $\epsilon = 0.01$  for SVHN and CIFAR-10 (note that  $0.01 > \frac{2}{255} \approx 0.0078$ ). The chosen  $\epsilon = 0.01$  for SVHN/CIFAR-10 is so small that the changes are hardly visible (see Figure ??). As parameter  $\kappa$  for the trade-off between cross-entropy loss and the GOOD regularizer in (9) and (10), we set  $\kappa = 0.3$  for MNIST and  $\kappa = 1$  for SVHN and CIFAR-10.

In order to explore the potential trade-off between the separation of in- and out-distribution for clean and perturbed out-distribution inputs (clean AUCs vs guaranteed AUCs - see below), we train GOOD models for different quantiles  $q \in [0, 1]$  in (10) which we denote as  $\text{GOOD}_Q$  in the following. Here,  $Q = 100q$  is the percentage of out-distribution training samples for which we minimize the guaranteed upper bounds on the confidence of the neural network in the  $l_\infty$ -ball of radius  $\epsilon$  around the out-distribution point during training. Note that  $\text{GOOD}_{100}$  corresponds to (9) where we minimize the guaranteed upper bound on the worst-case confidence for all out-distribution samples, whereas

GOOD<sub>0</sub> can be seen as a variant of OE or CEDA. A training batch consists of 128 in- and 128 out-distribution samples. Examples of OOD training batches with the employed augmentation and their quantile splits for a GOOD<sub>60</sub> model are shown in Table 3 in the appendix.

For the training out-distribution, we use 80 Million Tiny Images (80M) [37], which is a large collection of natural images associated to nouns in wordnet [11]. All methods get the same out-distribution for training and we are *neither* training *nor* adapting hyperparameters for each OOD dataset separately as in some previous work. Since CIFAR-10 and CIFAR-100 are subsets of 80M, we follow [19] and filter them out. Even after the filtering process we have observed that the remaining dataset still contains images from the CIFAR-10 and CIFAR-100 classes. Thus we have further excluded all samples for which a CIFAR-10 CEDA model has confidence above 11%, altogether removing 4.25M images. As can be seen in the example batches in Table 3, even this reduced dataset contains still images from CIFAR-10 classes, which explains why our quantile-based loss is essential to get good performance on CIFAR-10. We take a subset of 50 million images as OOD training set. Since the size of the training set of the in-distribution datasets (MNIST: 60,000; SVHN: 73,257; CIFAR-10: 50000) is small compared to 50 million, typically an OOD image appears only once during training.

**Evaluation.** For each method, we compute the test accuracy on the in-distribution task, and for various out-distribution datasets (not seen during training) we report the area under the receiver operating characteristic curve (AUC) as a measure for the separation of in- from out-distribution samples based on the predicted confidences on the test sets. As OOD evaluation sets we use FashionMNIST [39], the Letters of EMNIST [5], grayscale CIFAR-10, and Uniform Noise for MNIST, and CIFAR-100 [21], CIFAR-10/SVHN, LSUN Classroom [40], and Uniform Noise for SVHN/CIFAR-10. Further evaluation on other OOD datasets can be found in Appendix H.

We are particularly interested in the worst case OOD detection performance of all methods under the  $l_\infty$ -perturbation model for the out-distribution. For this purpose, we compute the **adversarial AUC (AAUC)** and the **guaranteed AUC (GAUC)**. These AUCs are based on the maximal confidence in the  $l_\infty$ -ball of radius  $\epsilon$  around each out-distribution image. For the adversarial AUC, we compute a lower bound on the maximal confidence in the  $l_\infty$ -ball by using Auto-PGD [8] for maximizing the confidence of the classifier inside the intersection of the  $l_\infty$ -ball and the image domain  $[0, 1]^d$ . Auto-PGD uses an automatic stepsize selection scheme and has been shown to outperform PGD. We use an adaptation to our setting (described in Appendix A) with 500 steps and 5 restarts on 1000 points from each test set. On MNIST, gradient masking poses a significant challenge so we use an additional attack discussed in Appendix A and report the worst case. For the guaranteed AUC, we compute an upper bound on the confidence in the intersection of the  $l_\infty$ -ball with the image domain  $[0, 1]^d$  via IBP using (6) for the full test set. These worst case/guaranteed confidences for the out-distributions are then used for the AUC computation.

**Competitors.** We compare a normally trained model (Plain), the state-of-the-art OOD detection method Outlier Exposure (OE) [19], CEDA [16] and Adversarial Confidence Enhanced Training (ACET) [16], which we adjusted to the given task as described in the appendix. As CEDA performs very similar to OE, we omit it in the figures for better readability. The  $\epsilon$ -radii for the  $l_\infty$ -balls are the same for ACET and GOOD. So far the only method which could provide robustness guarantees for OOD detection is Certified Certain Uncertainty (CCU) with a data-dependent Mahalanobis-type  $l_2$  threat model. We use their publicly available code to train a CCU model with our architecture and we evaluate their guarantees for our  $l_\infty$  threat model. In Appendix B, we provide details and explain why their guarantees turn out to be vacuous in our setting.

## 4.2 Results

In Table 1 we present the results on all datasets.

**GOOD is provably better than OE/CEDA with regard to worst case OOD detection.** We note that for almost all OOD datasets GOOD achieves non-trivial GAUCs. Thus the guarantees generalize from the training out-distribution 80M to the test OOD datasets. For the easier in-distributions MNIST and SVHN, which are more clearly separated from the out-distribution, the best results are achieved for GOOD<sub>100</sub> whereas for CIFAR-10 the best guarantees are given by GOOD<sub>90</sub> or GOOD<sub>95</sub>. However, if taking clean AUCs into account, arguably the best trade-off is achieved for GOOD<sub>80</sub>. Note that the guaranteed AUC (GAUC) of these models is always better than the adversarial AUC (AAUC) of OE/CEDA (except for EMNIST). Thus it is fair to say that the worst-case OOD detection

Table 1: Accuracies as well as AUC, adversarial AUC (AAUC) and guaranteed AUC (GAUC) values for the MNIST, SVHN and CIFAR-10 in-distributions with respect to several unseen out-distributions. GOOD is the only method with non-zero GAUC which for GOOD<sub>100</sub> on MNIST/SVHN and GOOD<sub>80</sub> on CIFAR-10 on almost all OOD datasets (except EMNIST) is better than the AAUC of OE and CEDA. Thus GOOD is provably better than OE and CEDA w.r.t. worst-case OOD detection. GOOD achieves this without significant loss in accuracy. Especially on SVHN, GOOD<sub>100</sub> has very good accuracy and almost perfect provable worst-case OOD detection performance.

		AUC	AAUC	GAUC	AUC	AAUC	GAUC	AUC	AAUC	GAUC	AUC	AAUC	GAUC
PLAIN	<b>99.5</b>	97.7	1.3	0.0	87.9	0.5	0.0	98.5	0.1	0.0	99.4	0.0	0.0
CEDA	<b>99.5</b>	<b>100.0</b>	78.4	0.0	92.8	17.9	0.0	<b>100.0</b>	86.4	0.0	<b>100.0</b>	<b>100.0</b>	0.0
OE	<b>99.5</b>	99.9	69.7	0.0	92.8	13.5	0.0	<b>100.0</b>	93.2	0.0	<b>100.0</b>	<b>100.0</b>	0.0
ACET	<b>99.5</b>	<b>100.0</b>	<b>99.5</b>	0.0	96.4	62.6	0.0	<b>100.0</b>	<b>100.0</b>	0.0	<b>100.0</b>	<b>100.0</b>	0.0
CCU	<b>99.5</b>	<b>100.0</b>	76.6	0.0	92.9	3.1	0.0	<b>100.0</b>	98.9	0.0	<b>100.0</b>	<b>100.0</b>	0.0
GOOD <sub>0</sub>	<b>99.5</b>	99.9	78.6	0.0	92.8	15.7	0.0	<b>100.0</b>	98.6	0.0	<b>100.0</b>	<b>100.0</b>	0.0
GOOD <sub>20</sub>	99.2	99.8	88.0	9.6	95.2	47.4	0.0	<b>100.0</b>	<b>100.0</b>	26.7	<b>100.0</b>	<b>100.0</b>	<b>100.0</b>
GOOD <sub>40</sub>	99.1	99.8	94.4	28.7	95.7	58.1	0.0	<b>100.0</b>	<b>100.0</b>	65.2	<b>100.0</b>	<b>100.0</b>	<b>100.0</b>
GOOD <sub>60</sub>	98.9	99.9	95.2	43.3	96.6	64.6	0.2	<b>100.0</b>	<b>100.0</b>	85.2	<b>100.0</b>	<b>100.0</b>	<b>100.0</b>
GOOD <sub>80</sub>	99.0	99.9	96.1	57.0	97.8	69.4	4.2	<b>100.0</b>	<b>100.0</b>	95.0	<b>100.0</b>	<b>100.0</b>	<b>100.0</b>
GOOD <sub>90</sub>	98.9	99.9	96.4	67.4	98.2	<b>69.6</b>	<b>6.1</b>	<b>100.0</b>	<b>100.0</b>	97.8	<b>100.0</b>	<b>100.0</b>	<b>100.0</b>
GOOD <sub>95</sub>	98.8	99.9	96.3	73.1	98.6	66.0	4.7	<b>100.0</b>	<b>100.0</b>	98.8	<b>100.0</b>	<b>100.0</b>	<b>100.0</b>
GOOD <sub>100</sub>	98.7	<b>100.0</b>	97.1	<b>79.2</b>	<b>98.9</b>	59.4	3.3	<b>100.0</b>	<b>100.0</b>	<b>99.4</b>	<b>100.0</b>	<b>100.0</b>	<b>100.0</b>

		IN: SVHN						$\epsilon = 0.01$					
METHOD	ACC.	CIFAR-100			CIFAR-10			LSUN CLASSROOM			UNIFORM NOISE		
		AUC	AAUC	GAUC	AUC	AAUC	GAUC	AUC	AAUC	GAUC	AUC	AAUC	GAUC
PLAIN	95.5	94.9	57.1	0.0	95.2	59.3	0.0	95.7	69.2	0.0	99.4	91.4	0.0
CEDA	95.4	99.9	98.7	0.0	99.9	97.3	0.0	<b>100.0</b>	99.7	0.0	<b>100.0</b>	99.9	0.0
OE	95.5	<b>100.0</b>	98.2	0.0	<b>100.0</b>	98.1	0.0	<b>100.0</b>	99.7	0.0	<b>100.0</b>	<b>100.0</b>	0.0
ACET	95.6	<b>100.0</b>	<b>99.8</b>	0.0	<b>100.0</b>	<b>99.9</b>	0.0	<b>100.0</b>	<b>100.0</b>	0.0	99.7	96.8	0.0
CCU	95.7	<b>100.0</b>	98.0	0.0	<b>100.0</b>	97.8	0.0	<b>100.0</b>	99.7	0.0	<b>100.0</b>	<b>100.0</b>	0.0
GOOD <sub>0</sub>	<b>97.0</b>	<b>100.0</b>	97.4	0.0	<b>100.0</b>	97.6	0.0	<b>100.0</b>	99.8	0.0	<b>100.0</b>	<b>100.0</b>	0.0
GOOD <sub>20</sub>	96.5	99.9	99.3	19.1	99.9	99.4	17.9	99.9	99.9	23.6	99.9	<b>100.0</b>	99.9
GOOD <sub>40</sub>	96.1	99.9	99.3	43.2	<b>100.0</b>	99.4	48.1	<b>100.0</b>	99.9	60.0	<b>100.0</b>	<b>100.0</b>	<b>100.0</b>
GOOD <sub>60</sub>	95.8	99.9	99.5	64.8	<b>100.0</b>	99.6	71.0	<b>100.0</b>	99.9	85.0	<b>100.0</b>	<b>100.0</b>	99.9
GOOD <sub>80</sub>	95.9	99.6	99.4	87.6	99.6	99.4	91.2	99.6	99.6	97.5	99.6	99.6	98.7
GOOD <sub>90</sub>	96.3	99.9	99.7	95.8	99.9	99.8	97.7	<b>100.0</b>	<b>100.0</b>	99.7	<b>100.0</b>	99.9	99.3
GOOD <sub>95</sub>	96.2	99.8	99.6	97.8	99.8	99.7	98.9	99.8	99.8	99.7	99.8	99.8	99.6
GOOD <sub>100</sub>	96.6	99.9	99.7	<b>99.4</b>	<b>100.0</b>	<b>99.9</b>	<b>99.7</b>	<b>100.0</b>	<b>100.0</b>	<b>99.9</b>	<b>100.0</b>	<b>100.0</b>	99.9

		IN: CIFAR-10						$\epsilon = 0.01$					
METHOD	ACC.	CIFAR-100			SVHN			LSUN CLASSROOM			UNIFORM NOISE		
		AUC	AAUC	GAUC	AUC	AAUC	GAUC	AUC	AAUC	GAUC	AUC	AAUC	GAUC
PLAIN	90.1	84.3	6.3	0.0	87.7	10.1	0.0	88.9	9.8	0.0	90.6	56.8	0.0
CEDA	87.6	90.9	35.0	0.0	97.6	42.4	0.0	98.0	47.1	0.0	97.8	81.9	0.0
OE	90.5	92.2	20.3	0.0	<b>98.1</b>	17.7	0.0	98.9	38.9	0.0	97.4	72.6	0.0
ACET	89.1	90.5	<b>76.1</b>	0.0	96.9	<b>91.0</b>	0.0	98.5	91.7	0.0	97.8	96.3	0.0
CCU	<b>91.6</b>	<b>93.0</b>	23.3	0.0	97.1	14.8	0.0	<b>99.3</b>	38.2	0.0	<b>100.0</b>	<b>100.0</b>	0.0
GOOD <sub>0</sub>	89.6	<b>93.0</b>	28.2	0.0	97.7	35.5	0.0	98.1	67.5	0.0	97.7	95.1	0.0
GOOD <sub>20</sub>	86.4	88.3	45.8	9.4	94.3	51.3	19.5	97.6	65.4	1.7	94.8	92.2	90.9
GOOD <sub>40</sub>	89.6	89.6	62.1	25.3	96.1	89.1	25.1	96.4	<b>92.1</b>	28.8	93.2	91.7	91.4
GOOD <sub>60</sub>	89.8	88.7	63.2	34.8	95.8	80.3	28.5	96.5	90.6	53.2	91.5	90.3	89.7
GOOD <sub>80</sub>	90.0	85.9	67.3	46.3	95.6	84.8	41.0	96.2	90.7	<b>61.9</b>	90.5	87.5	87.0
GOOD <sub>90</sub>	90.2	81.8	64.2	51.6	91.0	76.4	<b>53.3</b>	94.5	85.7	61.5	93.8	92.4	91.9
GOOD <sub>95</sub>	90.3	77.5	60.0	<b>52.0</b>	89.6	71.2	47.9	84.6	66.0	57.5	98.4	98.1	98.0
GOOD <sub>100</sub>	90.3	73.9	45.7	30.0	74.3	30.3	4.7	82.7	63.4	54.2	99.1	98.7	<b>98.6</b>

Plain	4: 99.4	8: 100.0	0: 100.0	0: 45.2	8: 65.9	4: 51.6	9: 99.9	4: 100.0	1: 100.0	5: 99.9
OE	4: 95.5	8: 100.0	0: 99.7	4: 72.5	8: 65.8	4: 48.6	9: 99.9	4: 100.0	1: 99.9	5: 99.8
CCU	4: 96.6	8: 100.0	0: 99.9	4: 70.9	8: 56.3	4: 67.3	9: 100.0	4: 100.0	1: 100.0	5: 99.2
ACET	4: 92.3	8: 98.5	0: 94.2	4: 35.0	8: 58.6	7: 58.1	9: 96.1	4: 99.8	1: 99.8	5: 68.7
GOOD <sub>100</sub>	2: 10.0	3: 21.8	0: 22.7	6: 14.7	2: 10.0	7: 16.6	9: 12.8	2: 10.0	1: 80.8	5: 25.3




Figure 2: Random samples from 10 letters in the out-distribution dataset EMNIST. The predictions and confidences of all methods trained on MNIST are shown on top. GOOD<sub>100</sub> is the only method which is **not** overconfident (e.g. “H”) unless the letter is indistinguishable from a digit (“I”).

performance of GOOD is provably better than that of OE/CEDA. As expected, ACET yields good AAUCs but has no guarantees. The failure of CCU regarding guarantees is discussed in Appendix B. It is notable that GOOD<sub>100</sub> has basically perfect guaranteed OOD detection performance for MNIST on CIFAR-10/uniform noise and for SVHN on **all** out-distribution datasets. In Appendix I we show that the guarantees of GOOD partially hold even at larger radii than used during training.

**GOOD achieves certified OOD performance with almost no loss in accuracy.** While there is a small drop in clean accuracy for MNIST, on SVHN, GOOD<sub>100</sub> has with 96.6% a better clean accuracy than all competing methods. On CIFAR-10 GOOD<sub>80</sub> achieves an accuracy of 90.0% which is better than ACET and only slightly worse than Plain and OE. This is remarkable as we are not aware of any model with certified *adversarial robustness on the in-distribution* which gets even close to this range; e.g. IBP [13] achieves an accuracy of 85.2% on SVHN with  $\epsilon = 0.01$  (we have 96.6%), on CIFAR-10 with  $\epsilon = \frac{2}{255}$  they get 71.2% (we have 90.0%). Previous certified methods had even worse clean accuracy. Since a significant loss in prediction performance is usually not acceptable, certified methods have not yet had much practical impact. Thus we think it is an encouraging and interesting observation that properties different from adversarial robustness like worst-case out-of-distribution detection can be certified without suffering much in accuracy. In particular, it is quite surprising that certified methods can be trained effectively with aggressive data augmentation like AutoAugment.

**Trade-off between clean and guaranteed AUC via Quantile-GOOD.** As discussed above, even after filtering, our training out-distribution contains in-distribution images from CIFAR-10 classes. This seems to be the reason why GOOD<sub>100</sub> suffers from a significant drop in clean and guaranteed AUC, as the only way to ensure small loss  $\mathcal{L}_{\text{CUB}}$ , if in- and out-distribution can partially not be distinguished, is to reduce also the confidence on the in-distribution. This conflict is then resolved via GOOD<sub>80</sub> and GOOD<sub>90</sub> which both have better clean and guaranteed AUCs. It is an interesting open question if similar trade-offs are potentially also useful for certified adversarial robustness.

**EMNIST: distinguishing letters from digits without ever having seen letters.** GOOD<sub>100</sub> achieves an excellent AUC of 98.9% for the letters of EMNIST which is, up to our knowledge, state-of-the-art. Indeed, an AUC of 100% should not be expected as even for humans some letters like i and l are indistinguishable from digits. This result is quite remarkable as GOOD<sub>100</sub> has never seen letters during training. Moreover, as the AUC just distinguishes the separation of in-and out-distribution based on the confidence, we provide the mean confidence on all datasets in the Appendix in Table 4 and in Figure 2 (see also Figure 3 in the Appendix) we show some samples from EMNIST together with their prediction/confidences for all models. GOOD<sub>100</sub> has a mean confidence of 98.3% on MNIST but only 27.8% on EMNIST in contrast to ACET with 71.3%, OE 87.7% and Plain 91.3%. This shows that while the AUC’s of ACET and OE are good for EMNIST, these methods are still highly overconfident on EMNIST. Only GOOD<sub>100</sub> produces meaningful higher confidences on EMNIST, when the letter has clear features of the corresponding digit.

## 5 Conclusion

We propose GOOD, a novel training method to achieve guaranteed OOD detection in a worst-case setting. GOOD provably outperforms OE, the state-of-the-art in OOD detection, in worst case OOD detection and has state-of-the-art performance on EMNIST which is a particularly challenging out-distribution dataset. As the test accuracy of GOOD is comparable to the one of normal training, this shows that certified methods have the potential to be useful in practice even for more complex tasks. In future work it will be interesting to explore how close certified methods can get to state-of-the-art test performance.



## Broader Impact

In order to use machine learning in safety-critical systems it is required that the machine learning system correctly flags its uncertainty. As neural networks have been shown to be overconfident far away from the training data, this work aims at overcoming this issue by not only enforcing low confidence on out-distribution images but even guaranteeing low confidence in a neighborhood around it. As a neural network should not flag that it knows when it does not know, we see only positive implications of this work for our society.

## Acknowledgements

The authors acknowledge support from the German Federal Ministry of Education and Research (BMBF) through the Tübingen AI Center (FKZ: 01IS18039A) and from the Deutsche Forschungsgemeinschaft (DFG, German Research Foundation) under Germany’s Excellence Strategy (EXC number 2064/1, Project number 390727645). The authors thank the International Max Planck Research School for Intelligent Systems (IMPRS-IS) for supporting Alexander Meinke.

## References

- [1] A. Athalye, N. Carlini, and D. Wagner. Obfuscated gradients give a false sense of security: Circumventing defenses to adversarial examples. In *ICML*, 2018.
- [2] B. Biggio, I. Corona, D. Maiorca, B. Nelson, N. Šrndić, P. Laskov, G. Giacinto, and F. Roli. Evasion attacks against machine learning at test time. In *ECML/PKDD*, 2013.
- [3] N. Carlini and D. Wagner. Adversarial examples are not easily detected: Bypassing ten detection methods. In *ACM Workshop on Artificial Intelligence and Security*, 2017.
- [4] N. Carlini and D. Wagner. Towards evaluating the robustness of neural networks. In *IEEE Symposium on Security and Privacy*, 2017.
- [5] G. Cohen, S. Afshar, J. Tapson, and A. Van Schaik. Emnist: Extending mnist to handwritten letters. In *IJCNN*, 2017.
- [6] J. M. Cohen, E. Rosenfeld, and J. Z. Kolter. Certified adversarial robustness via randomized smoothing. In *ICML*, 2019.
- [7] F. Croce and M. Hein. Sparse and imperceivable adversarial attacks. In *ICCV*, 2019.
- [8] F. Croce and M. Hein. Reliable evaluation of adversarial robustness with an ensemble of diverse parameter-free attacks. preprint, arXiv:2003.01690, 2020.
- [9] E. D. Cubuk, B. Zoph, D. Mane, V. Vasudevan, and Q. V. Le. Autoaugment: Learning augmentation strategies from data. In *CVPR*, 2019.
- [10] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei. Imagenet: A large-scale hierarchical image database. In *CVPR*, 2009.
- [11] C. Fellbaum. Wordnet. *The encyclopedia of applied linguistics*, 2012.
- [12] R. Geirhos, C. R. Temme, J. Rauber, H. H. Schütt, M. Bethge, and F. A. Wichmann. Generalisation in humans and deep neural networks. In *NeurIPS*, 2018.
- [13] S. Gowal, K. Dvijotham, R. Stanforth, R. Bunel, C. Qin, J. Uesato, R. Arandjelovic, T. Mann, and P. Kohli. On the effectiveness of interval bound propagation for training verifiably robust models. *arXiv:1810.12715*, 2018.
- [14] C. Guo, G. Pleiss, Y. Sun, and K. Weinberger. On calibration of modern neural networks. In *ICML*, 2017.
- [15] M. Hein and M. Andriushchenko. Formal guarantees on the robustness of a classifier against adversarial manipulation. In *NeurIPS*, 2017.
- [16] M. Hein, M. Andriushchenko, and J. Bitterwolf. Why ReLU networks yield high-confidence predictions far away from the training data and how to mitigate the problem. In *CVPR*, 2019.

- [17] D. Hendrycks and T. Dietterich. Benchmarking neural network robustness to common corruptions and perturbations. In *ICLR*, 2019.
- [18] D. Hendrycks and K. Gimpel. A baseline for detecting misclassified and out-of-distribution examples in neural networks. In *ICLR*, 2017.
- [19] D. Hendrycks, M. Mazeika, and T. Dietterich. Deep anomaly detection with outlier exposure. In *ICLR*, 2019.
- [20] D. P. Kingma and J. Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.
- [21] A. Krizhevsky and G. Hinton. Learning multiple layers of features from tiny images. Technical report, Citeseer, 2009.
- [22] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.
- [23] K. Lee, H. Lee, K. Lee, and J. Shin. Training confidence-calibrated classifiers for detecting out-of-distribution samples. In *ICLR*, 2018.
- [24] K. Lee, K. Lee, H. Lee, and J. Shin. A simple unified framework for detecting out-of-distribution samples and adversarial attacks. In *NeurIPS*, 2018.
- [25] S. Liang, Y. Li, and R. Srikant. Enhancing the reliability of out-of-distribution image detection in neural networks. In *ICLR*, 2018.
- [26] S. Liu, R. Garrepalli, T. Dietterich, A. Fern, and D. Hendrycks. Open category detection with PAC guarantees. In *PMLR*, 2018.
- [27] A. Madry, A. Makelov, L. Schmidt, D. Tsipras, and A. Valdu. Towards deep learning models resistant to adversarial attacks. In *ICLR*, 2018.
- [28] A. Meinke and M. Hein. Towards neural networks that provably know when they don’t know. In *ICLR*, 2020.
- [29] M. Mirman, T. Gehr, and M. Vechev. Differentiable abstract interpretation for provably robust neural networks. In *ICML*, 2018.
- [30] Y. Netzer, T. Wang, A. Coates, A. Bissacco, B. Wu, and A. Y. Ng. Reading digits in natural images with unsupervised feature learning. In *NeurIPS Workshop on Deep Learning and Unsupervised Feature Learning*, 2011.
- [31] A. Nguyen, J. Yosinski, and J. Clune. Deep neural networks are easily fooled: High confidence predictions for unrecognizable images. In *CVPR*, 2015.
- [32] 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. *Journal of Machine Learning Research*, 12:2825–2830, 2011.
- [33] A. Ragunathan, J. Steinhardt, and P. Liang. Certified defenses against adversarial examples. In *ICLR*, 2018.
- [34] L. Schott, J. Rauber, M. Bethge, and W. Brendel. Towards the first adversarially robust neural network model on mnist. In *ICLR*, 2019.
- [35] V. Sehwag, A. N. Bhagoji, L. Song, C. Sitawarin, D. Cullina, M. Chiang, and P. Mittal. Better the devil you know: An analysis of evasion attacks using out-of-distribution adversarial examples. *preprint, arXiv:1905.01726*, 2019.
- [36] C. Szegedy, W. Zaremba, I. Sutskever, J. Bruna, D. Erhan, I. Goodfellow, and R. Fergus. Intriguing properties of neural networks. In *ICLR*, 2014.
- [37] A. Torralba, R. Fergus, and W. T. Freeman. 80 million tiny images: A large data set for nonparametric object and scene recognition. *IEEE transactions on pattern analysis and machine intelligence*, 30(11):1958–1970, 2008.
- [38] E. Wong and J. Z. Kolter. Provable defenses against adversarial examples via the convex outer adversarial polytope. In *ICML*, 2018.

- [39] H. Xiao, K. Rasul, and R. Vollgraf. Fashion-MNIST: a novel image dataset for benchmarking machine learning algorithms. preprint, arXiv:1708.07747, 2017.
- [40] F. Yu, Y. Zhang, S. Song, A. Seff, and J. Xiao. Lsun: Construction of a large-scale image dataset using deep learning with humans in the loop. *CoRR*, abs/1506.03365, 2015.
- [41] H. Zhang, H. Chen, C. Xiao, S. Gowal, R. Stanforth, B. Li, D. Boning, and C.-J. Hsieh. Towards stable and efficient training of verifiably robust neural networks. In *ICLR*, 2020.
- [42] H. Zhang, Y. Yu, J. Jiao, E. P. Xing, L. E. Ghaoui, and M. I. Jordan. Theoretically principled trade-off between robustness and accuracy. In *ICML*, 2019.

## APPENDIX

### A Adversarial attacks on OOD detection

It has been demonstrated [2, 36, 4, 7] that without strong countermeasures, DNNs are very susceptible to adversarial attacks changing the classification result. The goal of adversarial attacks in our setting is to fool the OOD detection which is based on the confidence in the prediction. Thus the attacker aims at maximizing the confidence in a neighborhood around a given out-distribution input  $x$  so that the adversarially modified image will be wrongly assigned to the in-distribution. In this paper, we regard as threat model/neighborhood an  $l_\infty$ -ball of a given radius  $\epsilon$ , that is  $\{z \in [0, 1]^d \mid \|z - x\|_\infty \leq \epsilon\}$ ; note that in our case the disturbed inputs have to be valid images, hence the additional constraint  $z \in [0, 1]^d$ .

For evaluation, we use Auto-PGD [8], which is a state-of-the-art implementation of PGD (projected gradient descent), with backtracking, adaptive step sizes and random restarts. Since Auto-PGD has been designed for finding adversarial samples around the in-distribution, we change the objective of Auto-PGD to be the confidence of the classifier. We use Auto-PGD with 500 steps and 5 random restarts which is a quite strong attack. By default, the random initialization is drawn uniformly from the  $\epsilon$ -ball. However, we found that for MNIST the attack very often got stuck for our GOOD models, because a large random perturbation of size 0.3 would move the sample directly into a region of the input space where the model is completely flat and thus no gradients are available (in this sense adversarial attacks on OOD inputs are more difficult than usual adversarial attacks on the in-distribution). We instead use a modified version of the attack for MNIST which starts within short distance of the original point. Thus we use as initialization a random perturbation from  $[-0.01, 0.01]^d$  (note that for our evaluation on SVHN and CIFAR10, this choice coincides with the default settings).

Nevertheless, for MNIST most out-distribution points lie in regions where the predictions of our GOOD models are flat, i.e. the gradients are exactly zero. Because of this, Auto-PGD is unable to effectively explore the search space around those points. Thus, for MNIST we created an adaptive attack which partially circumvents these issues. First, we use an initialization scheme that mitigates lack of gradients by increasing the contrast as much as the threat model allows. All pixel values  $x_i$  that lie above  $1 - \epsilon$  get set to  $x_i = 1$  and all values  $x_i \leq 1 - \epsilon$  get set to  $\max\{0, x_i - \epsilon\}$ . In our experience these points are more likely to yield gradients, so we use them as initialization for a 200-step PGD attack with backtracking, adaptive step size selection and momentum of 0.9. Concretely, we use a step size of 0.1, and whenever a PGD step does not increase the confidence we backtrack and halve the step size. After every successful gradient step we multiply the step size by 1.1. Using backtracking and adaptive step size is necessary because otherwise one can easily step into regions where gradient information is no longer available. Additionally, to further mitigate the problem of gradient-masking at initialization, we use the adversarial images that Auto-PGD finds for models without significant gradient masking (Plain, OE, GOOD<sub>0</sub>) as initialization for the same monotone PGD for models which show significant gradient masking (CEDA, ACET).

### B A review of robust OOD detection

**ACET** A method that was proposed in order to achieve adversarially robust low confidence on OOD data is Adversarial Confidence Enhancing Training (ACET) [16] which is based on adversarial training on the out-distribution. However, similar to adversarial training on the in-distribution, this does typically not lead to any guarantees, whereas our goal is to get guarantees on the confidences of worst-case out-distribution inputs. ACET has the following objective:

$$\frac{1}{N} \sum_{i=1}^N \mathcal{L}_{\text{CE}}(x_i^{\text{IN}}, y_i^{\text{IN}}) + \frac{\kappa}{M} \sum_{j=1}^M \max_{\|\hat{x} - x_j^{\text{OUT}}\|_\infty \leq \epsilon} \mathcal{L}_{\text{OUT}}(\hat{x}). \quad (11)$$

They use  $\mathcal{L}_{\text{OUT}} = \log \text{Conf}_f$  with low frequency noise as their training out-distribution. We found firstly that training an ACET model with 80M as out-distribution yields much better results than the smoothed uniform noise used in [16] and secondly using the cross-entropy loss with respect to the uniform prediction instead of  $\log \text{Conf}_f$  also leads to improvements. For training ACET models, we employ a standard PGD attack with 40 steps of size  $\frac{2\epsilon}{41}$  with initialization at the target input for maximizing the loss around  $x_j^{\text{OUT}}$ . As usual for a  $l_\infty$ -attack, we use the sign of the gradient as direction and project onto the intersection of the image domain  $[0, 1]^d$  and the  $l_\infty$ -ball of radius  $\epsilon$  around the target. Finally, the attack returns the image with the highest confidence found during the iterations. For the attack at training time we use no backtracking or adaptive stepsizes. ACET does not provide any guaranteed confidence bounds.

**CCU** Certified Certain Uncertainty (CCU) [28] gives low confidence guarantees around certain OOD data that is far away from the training dataset in a specific metric. Those bounds do hold on such far-away datasets, but do not generalize to inputs relatively close to the in distribution, like for example CIFAR-10 vs. CIFAR-100. Moreover, even in the regime where CCU yields meaningful guarantees, they are given in terms of a

data-dependent Mahalanobis distance rather than the  $l_\infty$ -distance. However, due to norm equivalences, one can still extract  $l_\infty$ -guarantees from CCU and we evaluated the CCU guarantees as follows. We use the corollary 3.1 from [28] which states that for a CCU model that is written as

$$p(y|x) = \frac{p(y|x, i)p(x|i) + \frac{1}{K}p(x|o)}{p(x|i) + p(x|o)} \quad (12)$$

with  $p(y|x, i)$  being the softmax output of a neural network and  $p(x|i)$  and  $p(x|o)$  Gaussian mixture models for in- and out-distribution, one can bound the confidence in a certain neighborhood around any point  $x \in \mathbb{R}^d$  via

$$\max_{d_M(\hat{x}, x) \leq R} p(y|x) \leq \frac{1}{K} \frac{1 + K b(x, R)}{1 + b(x, R)}. \quad (13)$$

Here  $b : \mathbb{R}^d \times \mathbb{R}_+ \rightarrow \mathbb{R}_+$  is a positive function that increases monotonically in the radius  $R$  and that depends on the parameters of the Gaussian mixture models (details in [28]). The metric  $d_M : \mathbb{R}^d \times \mathbb{R}^d \rightarrow \mathbb{R}_+$  that they used for their CCU model is given as

$$d_M(x, y) = \left\| C^{-\frac{1}{2}}(x - y) \right\|, \quad (14)$$

where  $C$  is a regularized version of the covariance matrix, calculated on the augmented in-distribution data. Note that this Mahalanobis metric is strongly equivalent to the metric induced by the  $l_2$ -norm and consequently to the metric induced by the  $l_\infty$ -norm. By computing the equivalence constants between these metrics we can extract the  $l_\infty$ -guarantees that are implicit in the CCU model. Geometrically speaking, we compute the size of an ellipsoid (its shape determined by the eigenvalues of  $C$ ) that is large enough to fit a cube inside it with a radius given by our threat model  $r = 0.3$  or  $r = 0.01$ , respectively. Via norm equivalences one has

$$d_M(x, y) \leq \sqrt{\lambda_1} d_2(x, y) \leq \sqrt{d \lambda_1} d_\infty(x, y) \leq \sqrt{d \lambda_1} r, \quad (15)$$

where  $\lambda_1$  is the largest eigenvalue of  $C$ . This means that the confidence upper bounds from (13) on a Mahalanobis-ball of radius  $R = (d \lambda_1)^{\frac{1}{2}} r$  automatically apply to an  $l_\infty$ -ball of radius  $r$ . However, the covariance matrix  $C$  is highly ill-conditioned, which means that  $\lambda_1$  is fairly high. On top of that, in high dimensions  $\sqrt{d}$  is big as well so that in practice the required radius  $R$  becomes too large for CCU to certify meaningful guarantees. Even on uniform noise, the upper bounds were larger than the highest confidence on the in-distribution test set, with the consequence that there are no lower-bounds on the AAUC. However, we want to stress that at least for uniform noise the lack of guarantees of CCU is due to the incompatibility of the threat models used in our paper and [28].

Another type of guarantee that certifies a detection rate for OOD samples by applying probably approximately correct (PAC) learning considerations has been proposed in [26]. Their problem setting and nature of guarantees are not directly comparable to ours, since their guarantees handle behaviour on whole distributions while our guarantees are given for individual datapoints.

## C AUC and Conservative AUC

As a measure for the separation of in- vs. out-distribution data we use the Area Under the Receiver Operating Characteristic curve (AUROC or AUC) using the confidence of the classifier as the feature. The AUC is equal to the empirical probability of a random in-sample to be assigned a higher confidence than a random out-sample, plus one half times the probability of the confidences being equal. Thus, the standard way (as e.g. implemented in scikit-learn [32]) to calculate the AUC from given confidence values on sets of in- and out-distribution samples  $S_{in}$  and  $S_{out}$  is

$$\begin{aligned} \text{AUC}(f, S_{in}, S_{out}) = & \frac{1}{|S_{in}||S_{out}|} \left( |\{x_{in} \in S_{in}, x_{out} \in S_{out} \mid \text{Conf}_f(x_{in}) > \text{Conf}_f(x_{out})\}| \right. \\ & \left. + \frac{1}{2} |\{x_{in} \in S_{in}, x_{out} \in S_{out} \mid \text{Conf}_f(x_{in}) = \text{Conf}_f(x_{out})\}| \right), \end{aligned} \quad (16)$$

where for a set  $S$ ,  $|S|$  indicates the number of its elements. The half-weighted equality term gives this definition certain symmetry properties. However, it assigns a positive score to some completely uninformed functions  $f$ . For example, a constant uniform classifier with  $p_k(x) = \frac{1}{K}$  receives an AUC value of 50%. Similarly, a classifier that assigns 100% confidence to most in-distribution inputs would have positive AUC and even GAUC statistics, even if it fails to have confidence below 100% on any OOD inputs. In order to regard only example pairs where the distributions are positively distinguished, we define the **Conservative AUC** (cAUC) by dropping the equality term:

$$\text{cAUC}(f, S_{in}, S_{out}) := \frac{1}{|S_{in}||S_{out}|} |\{x_{in} \in S_{in}, x_{out} \in S_{out} \mid \text{Conf}_f(x_{in}) > \text{Conf}_f(x_{out})\}|. \quad (17)$$

While in general  $\text{cAUC}(f, S_{in}, S_{out}) \leq \text{AUC}(f, S_{in}, S_{out})$ , the confidences of all models presented in the paper are differentiated enough so that for all shown numbers actually  $\text{cAUC} = \text{AUC}$ . However, we have experienced that one can have models where the confidences (uniform or one-hot predictions) cannot be distinguished due to limited numerical precision. In these cases the normal AUC definition would indicate a certain discrimination where it is actually impossible to discriminate the confidences.

## D Experimental details

The layer compositions of the architectures used for all GOOD and baseline models are laid out in Table 2. No normalization of inputs or activations is used. Weight decay ( $l_2$ ) is set to 0.05 for MNIST and 0.005 for SVHN and CIFAR-10. For all runs, we use a batch size of 128 samples from both the in- and the out-distribution (where applicable). At <https://gitlab.com/Bitterwolf/GOOD> you can find the exact implementation.

Table 2: Model architectures used for MNIST (L), SVHN (XL) and CIFAR-10 (XL) experiments. Each convolutional and non-final affine layer is followed by a ReLU activation. All convolutions use a kernel size of 3, a padding of 1, and stride of 1, except for the third convolution which has stride=2.

L	XL
CONV2D(64)	CONV2D(128)
CONV2D(64)	CONV2D(128)
CONV2D(128) <sub>s=2</sub>	CONV2D(256) <sub>s=2</sub>
CONV2D(128)	CONV2D(256)
CONV2D(128)	CONV2D(256)
LINEAR(512)	LINEAR(512)
LINEAR(10)	LINEAR(512)
	LINEAR(10)

For the MNIST experiments, we use as optimizer SGD with 0.9 Nesterov momentum, with an initial learning rate of  $\frac{0.005}{128}$  that is divided by 5 after 50, 100, 200, 300 and 350 epochs, with a total number of 420 training epochs. For the GOOD, CEDA and OE runs, the first two epochs only use in-distribution  $\mathcal{L}_{CE}$ ; over the next 100 epochs, the value of  $\kappa$  is ramped up linearly from zero to its final value of 0.3 for GOOD/OE and 1.0 for CEDA, where it stays for the remaining 318 epochs. The  $\epsilon$  value in the  $\mathcal{L}_{CUB}$  loss for GOOD is also increased linearly, starting at epoch 10 and reaching its final value of 0.3 on epoch 130. CCU is trained using the publicly available code from [28], where we modify the architecture, learning rate schedule and data augmentation to be the same as OE. The initial learning rate for the Gaussian mixture models is  $1e-5/\text{batchsize}$  and gets dropped at the same epochs as the neural network learning rate. Our more aggressive data augmentation implies that our underlying Mahalanobis metric is not the same as they used in [28]. The ACET model for MNIST is warmed up with two epochs on the in-distribution only, then four with  $\kappa = 1.0$  and  $\epsilon = 0$ , and the full ACET loss with  $\kappa = 1.0$  and  $\epsilon = 0.3$  for the remaining epochs. The reason why we chose a smaller  $\kappa$  of 0.3 for the MNIST GOOD runs is that considering the large  $\epsilon$  for which guarantees are enforced, training with higher  $\kappa$  values makes training unstable without improving any validation results.

For the SVHN and CIFAR-10 baseline models, we used the ADAM optimizer [20] with initial learning rate  $\frac{0.01}{128}$  for SVHN and  $\frac{0.1}{128}$  for CIFAR-10 that was divided by 5 after 30 and 100 epochs, with a total number of 420 training epochs. For OE,  $\kappa$  is increased linearly from zero to one between epochs 60 and 360. The same holds for CCU which again uses the same hyperparameters as OE. Again, ACET is warmed up with two in-distribution-only and four OE epochs. Then it is trained with  $\kappa = 1.0$  and  $\epsilon = 0.01$ , with a shorter training time of 100 epochs (the same number as used in [16]).

In line with the experiences reported in [13] and [41], for GOOD training on SVHN and CIFAR-10 longer training schedules with slower ramping up of the  $\mathcal{L}_{CUB}$  loss are necessary, as adding the out-distribution loss defined in Equation (8) to the training objective at once will overwhelm the in-distribution cross-entropy loss and cause the model to collapse to uniform predictions for all inputs, without recovery. In order to reduce warm-up time, we use a pre-trained CEDA model for initialization and train for 900 epochs. The learning rate is  $10^{-4}$  in the beginning and is divided by 5 after epochs 450, 750 and 850. Due to the pre-training, we begin training with a small  $\kappa$  and already start with non-zero  $\epsilon$  after epoch 4. Then,  $\epsilon$  is increased linearly to its final value of 0.01 which is reached at epoch 204. Simultaneously,  $\kappa$  is increased linearly with a virtual starting point at epoch -2 to its final value of 1.0 at epoch 298.

Due to the tendency of IBP based training towards instabilities, the selection of hyper-parameters was based on finding settings where training is reliably stable while guaranteed bounds over meaningful  $\epsilon$  radii are possible.

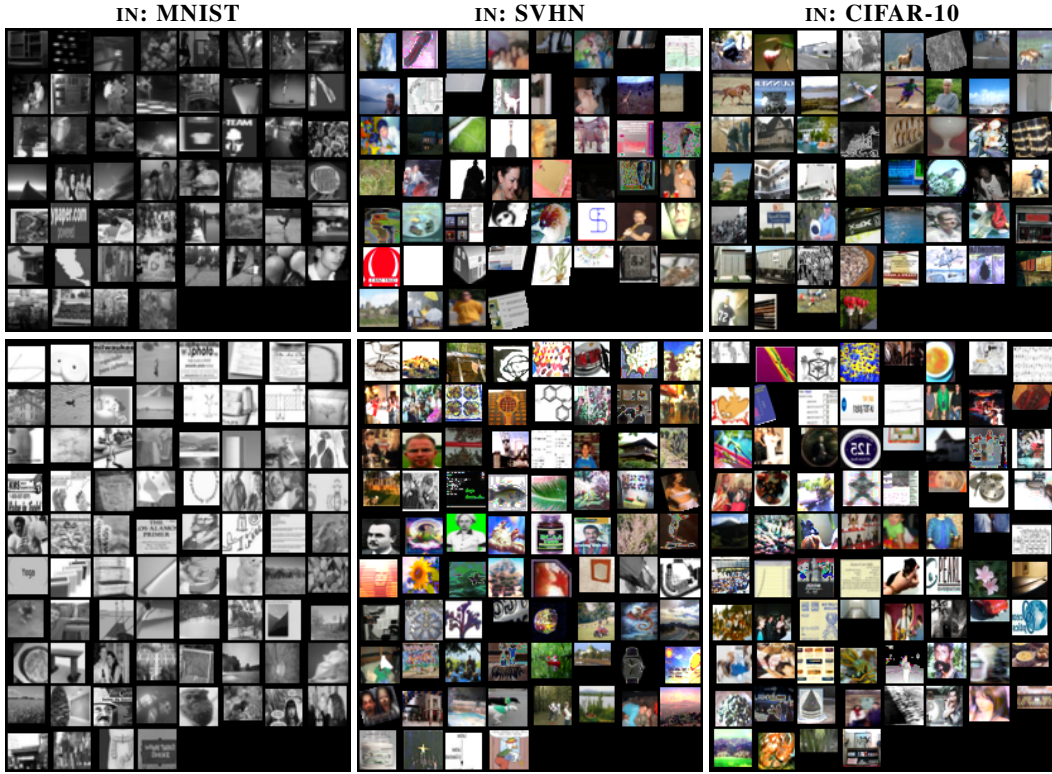
For the accuracy, AUC and GAUC evaluations in Table 1 the test splits of each (non-noise) dataset were used, with the following numbers of samples: 10,000 for MNIST, FashionMNIST, CIFAR-10, CIFAR-100 and Uniform Noise; 20,800 for EMNIST Letters; 26,032 for SVHN; 300 for LSUN Classroom. Due to the computational cost of the employed attacks, the AAUC values are based on subsets of 1000 samples for each dataset.

All experiments were run on Nvidia Tesla P100 and V100 GPUs, with GPU memory requirement below 16GB.

## E Depiction of GOOD Quantile-loss

In Quantile-GOOD training, the out-distribution part of each batch is split up into “harder” and “easier” parts, since trying to enforce low confidence guarantees on inputs that are very close to the in-distribution leads to low confidences in general, even on the in-distribution. In Table 3, we show example batches of GOOD<sub>60</sub> models with MNIST, SVHN and CIFAR-10 as in-distribution near the end of training (from epochs 410, 890 and 890, respectively). Even though many CIFAR-like images were filtered out, some are still present. For the CIFAR-10 model, such samples (among others) get sorted above the quantile. For MNIST, lower brightness images appear to be more difficult, while for SVHN images with fewer objects seem to be comparably hardest to distinguish from the house numbers of the in-distribution.

Table 3: Exemplary batch of out-distribution 80M Tiny Images (after augmentation) towards the end of training of GOOD<sub>60</sub> models. **Top:** The 52 Images with highest confidence upper bound. On these, loss is based on standard output. **Bottom:** The remaining 76 Images with lowest confidence upper bound. Here, loss is based on upper bounds within the  $\epsilon$ -ball.



## F Confidences on EMNIST

Figure 3 shows samples of the letters “k” to “z” together with the predictions and confidences of the GOOD<sub>100</sub> MNIST model and four baseline models, complementing Figure 2. Also on these samples we see that GOOD<sub>100</sub> only produces high confidences for letters when they show digit-specific features (“l”, “q”, “s”). All other methods including ACET also produce high confidences for letters which are quite distinct from digits (“m”, “n”, “p”, “y”).

The mean confidence values of the same selection of MNIST models for each letter of the alphabet for EMNIST are plotted in Figure 4. We observe that the mean confidence is mostly aligned with the intuitive likeness of a letter with some digit: GOOD<sub>100</sub> has the highest mean confidence on the letter inputs “i” and “l”, which in many cases do look like the digit “1”. Curiously, the confidence of GOOD<sub>100</sub> on the letter “o”, which even humans often cannot distinguish from a digit “0”, is generally low.

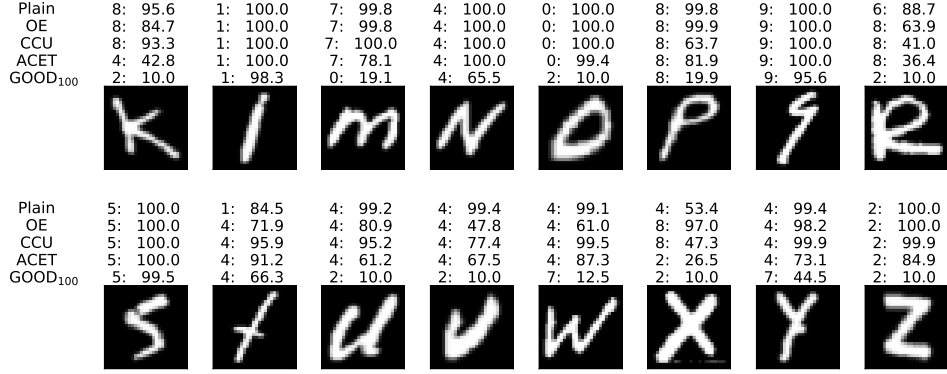


Figure 3: Continuation of Figure 2. Random samples from the remaining letters in the out-distribution dataset EMNIST. The predictions and confidences of different methods trained on MNIST are shown on top.

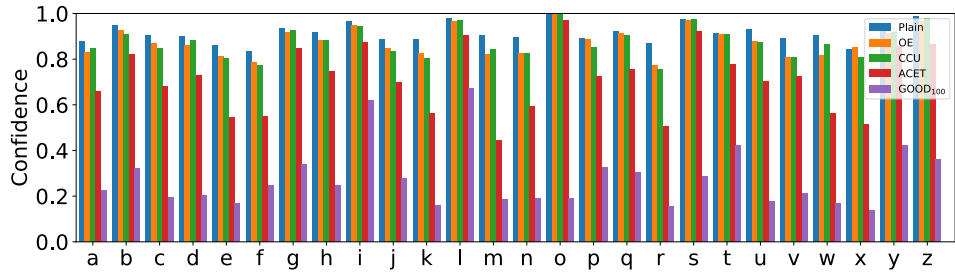


Figure 4: Mean confidence of different models across the classes of EMNIST-Letters. GOOD<sub>100</sub> only has high mean confidence on letters that can easily be mistaken for digits.

## G Distributions of confidences and confidence upper bounds

Table 4 shows the mean confidences of all models on the in-distribution as well as the mean confidences and the mean guaranteed upper bounds on the worst-case confidences on the evaluated out-distributions. As discussed, GOOD<sub>100</sub> training can reduce the confidence on the in-distribution, with a particularly strong effect for CIFAR-10. By adjusting the loss quantile, this effect can be significantly reduced while maintaining non-trivial guarantees.

The histograms of mean confidences on the in-distribution and mean guaranteed upper bounds on the worst-case confidences on the samples from the evaluated out-distribution test sets for seven models are shown in Tables 5 (MNIST), 6 (SVHN) and 7 (CIFAR-10). A higher GOOD loss quantile generally shifts the distribution of the upper bounds on the worst-case confidence towards smaller values, but in some cases, especially for GOOD<sub>100</sub> on CIFAR-10, strongly lowers confidences in in-distribution predictions as well.



Table 4: Mean confidence on the in-distribution and mean confidence / mean upper bounds on the confidence within the  $l_\infty$ -balls of radius  $\epsilon$  on the evaluated out-distribution datasets.

IN: MNIST $\epsilon = 0.3$					
METHOD	MNIST	FASHIONMNIST	EMNIST LETTERS	CIFAR-10	UNIFORM NOISE
PLAIN	99.7	80.3 / 100.0	91.3 / 100.0	79.3 / 100.0	73.4 / 100.0
CEDA	99.7	17.3 / 100.0	87.6 / 100.0	10.0 / 100.0	10.0 / 100.0
OE	99.7	24.3 / 100.0	87.7 / 100.0	10.1 / 100.0	10.0 / 100.0
ACET	99.4	11.0 / 100.0	71.5 / 100.0	10.0 / 100.0	10.0 / 100.0
CCU	99.7	17.5 / 100.0	87.4 / 100.0	10.0 / 100.0	10.0 / 100.0
GOOD <sub>0</sub>	99.7	20.6 / 100.0	87.8 / 100.0	10.0 / 100.0	10.0 / 100.0
GOOD <sub>20</sub>	99.5	19.8 / 93.2	69.7 / 100.0	10.0 / 78.1	10.0 / 10.0
GOOD <sub>40</sub>	99.3	19.2 / 77.4	58.7 / 100.0	10.0 / 43.9	10.0 / 10.0
GOOD <sub>60</sub>	99.2	15.7 / 64.6	49.4 / 99.9	10.0 / 24.9	10.0 / 10.0
GOOD <sub>80</sub>	99.1	15.4 / 54.1	40.3 / 98.4	10.0 / 15.3	10.0 / 10.0
GOOD <sub>90</sub>	98.8	15.5 / 47.3	36.1 / 97.8	10.0 / 12.7	10.0 / 10.0
GOOD <sub>95</sub>	98.8	12.7 / 42.9	32.6 / 98.6	10.0 / 11.7	10.0 / 10.0
GOOD <sub>100</sub>	98.3	11.1 / 39.5	27.8 / 99.2	10.0 / 11.0	10.0 / 10.0

IN: SVHN $\epsilon = 0.01$					
METHOD	SVHN	CIFAR-100	CIFAR-10	LSUN CLASSROOM	UNIFORM NOISE
PLAIN	97.7	70.8 / 100.0	70.5 / 100.0	66.8 / 100.0	40.5 / 100.0
CEDA	97.4	10.3 / 100.0	10.2 / 100.0	10.0 / 100.0	10.0 / 100.0
OE	97.2	10.9 / 100.0	10.6 / 100.0	10.3 / 100.0	10.2 / 100.0
ACET	95.5	10.2 / 100.0	10.1 / 100.0	10.1 / 100.0	15.1 / 100.0
CCU	97.2	10.8 / 100.0	10.6 / 100.0	10.4 / 100.0	10.0 / 100.0
GOOD <sub>0</sub>	98.7	10.0 / 100.0	10.0 / 100.0	10.0 / 100.0	10.0 / 100.0
GOOD <sub>20</sub>	97.9	10.1 / 83.0	10.1 / 84.0	10.0 / 79.0	10.0 / 10.0
GOOD <sub>40</sub>	97.7	10.1 / 61.5	10.1 / 57.1	10.0 / 46.6	10.0 / 10.0
GOOD <sub>60</sub>	97.4	10.1 / 42.3	10.1 / 36.6	10.0 / 23.8	10.0 / 10.2
GOOD <sub>80</sub>	97.4	10.1 / 25.9	10.1 / 21.5	10.0 / 13.5	10.0 / 13.6
GOOD <sub>90</sub>	97.2	10.1 / 17.6	10.0 / 14.9	10.0 / 10.9	10.0 / 11.6
GOOD <sub>95</sub>	96.9	10.1 / 14.9	10.0 / 13.1	10.0 / 10.4	10.0 / 11.0
GOOD <sub>100</sub>	95.7	10.2 / 12.6	10.0 / 11.5	10.0 / 10.3	10.0 / 10.0

IN: CIFAR-10 $\epsilon = 0.01$					
METHOD	CIFAR-10	CIFAR-100	SVHN	LSUN CLASSROOM	UNIFORM NOISE
PLAIN	95.1	79.0 / 100.0	75.8 / 100.0	73.9 / 100.0	73.2 / 100.0
CEDA	85.5	28.6 / 100.0	12.3 / 100.0	10.4 / 100.0	10.9 / 100.0
OE	86.9	31.7 / 100.0	16.1 / 100.0	13.7 / 100.0	18.0 / 100.0
ACET	75.1	25.9 / 100.0	16.1 / 100.0	13.3 / 100.0	15.1 / 100.0
CCU	89.4	32.5 / 100.0	20.5 / 100.0	12.6 / 100.0	10.0 / 100.0
GOOD <sub>0</sub>	80.7	18.7 / 100.0	10.7 / 100.0	10.1 / 100.0	10.0 / 100.0
GOOD <sub>20</sub>	78.4	27.0 / 92.5	14.1 / 84.8	11.2 / 98.7	11.0 / 13.0
GOOD <sub>40</sub>	81.0	25.1 / 85.3	11.1 / 85.6	10.6 / 90.6	11.7 / 12.3
GOOD <sub>60</sub>	71.9	21.4 / 75.8	11.5 / 80.9	10.8 / 62.5	13.1 / 13.6
GOOD <sub>80</sub>	62.8	21.9 / 63.7	11.8 / 66.2	11.3 / 45.4	15.0 / 15.3
GOOD <sub>90</sub>	56.4	24.2 / 54.1	15.0 / 52.0	12.9 / 41.4	13.5 / 13.6
GOOD <sub>95</sub>	45.7	26.1 / 44.4	17.6 / 47.5	21.2 / 40.1	11.5 / 11.5
GOOD <sub>100</sub>	28.7	24.4 / 28.3	25.2 / 31.0	22.7 / 26.7	13.9 / 14.0

Table 5: Histograms of the confidences on the **MNIST** in-distribution and guaranteed upper bounds on the confidences on OOD datasets within the  $l_\infty$ -ball of radius 0.3. Each histogram uses 50 bins between 0.1 and 1.0. For better readability, the scale is zoomed in by a factor 10 for numbers below one fifth of the total number of datapoints of the shown datasets. The vertical dotted line shows the mean value of the histogram’s data.

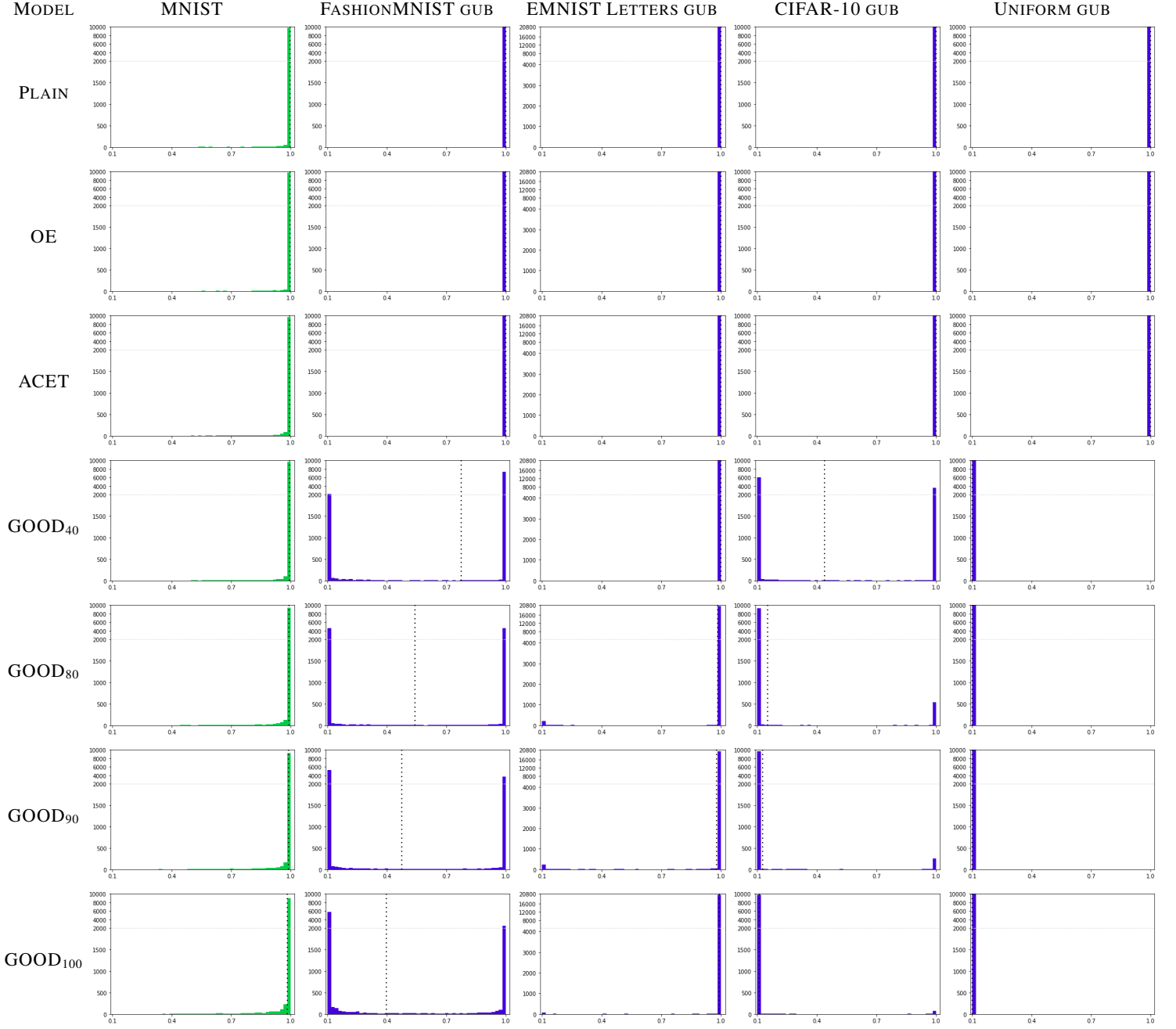


Table 6: Histograms of the confidences on the **SVHN** in-distribution and guaranteed upper bounds on the confidences on OOD datasets within the  $l_\infty$ -ball of radius 0.01. Each histogram uses 50 bins between 0.1 and 1.0. For better readability, the scale is zoomed in by a factor 10 for numbers below one fifth of the total number of datapoints of the shown datasets. The vertical dotted line shows the mean value of the histogram’s data.

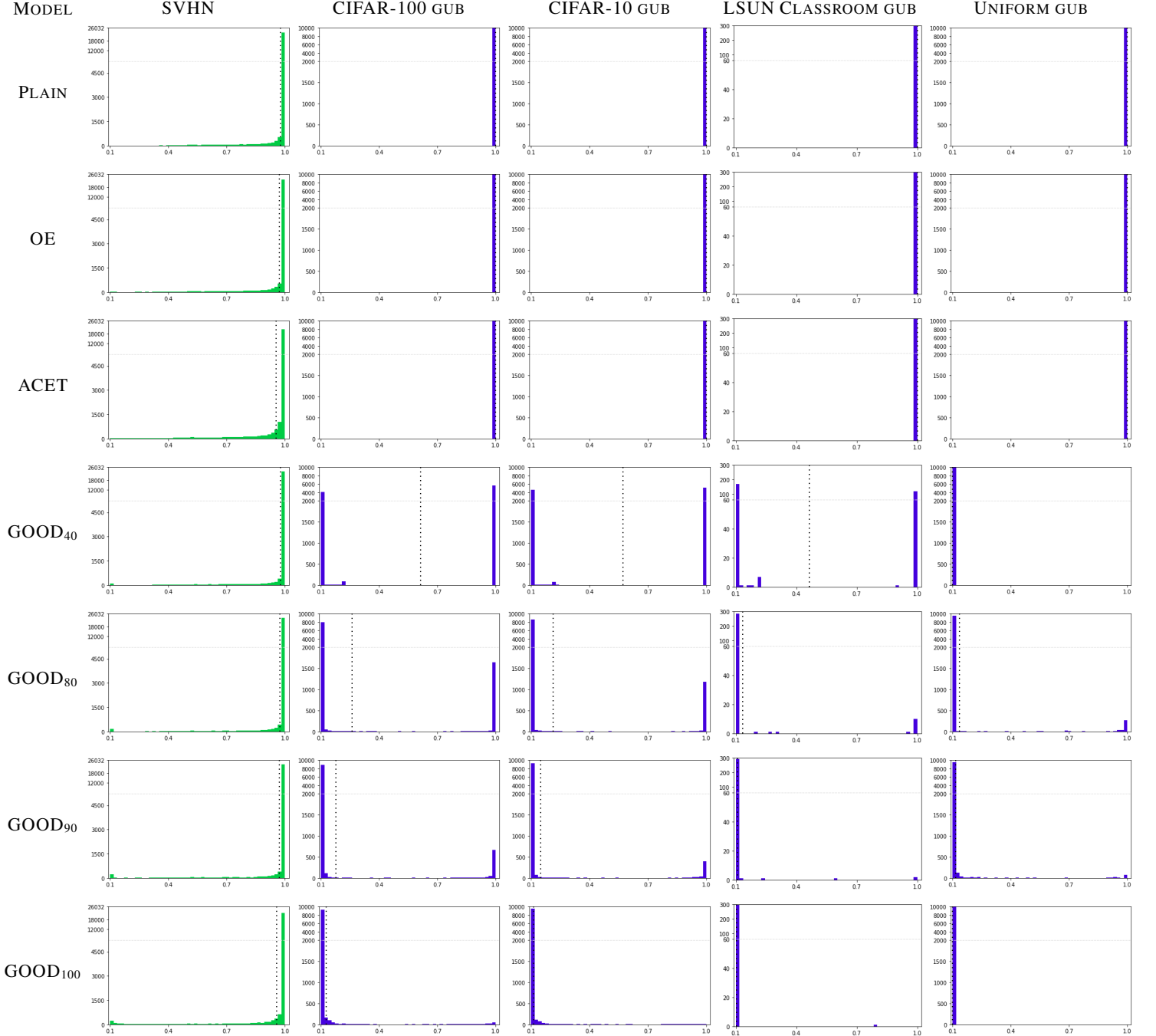
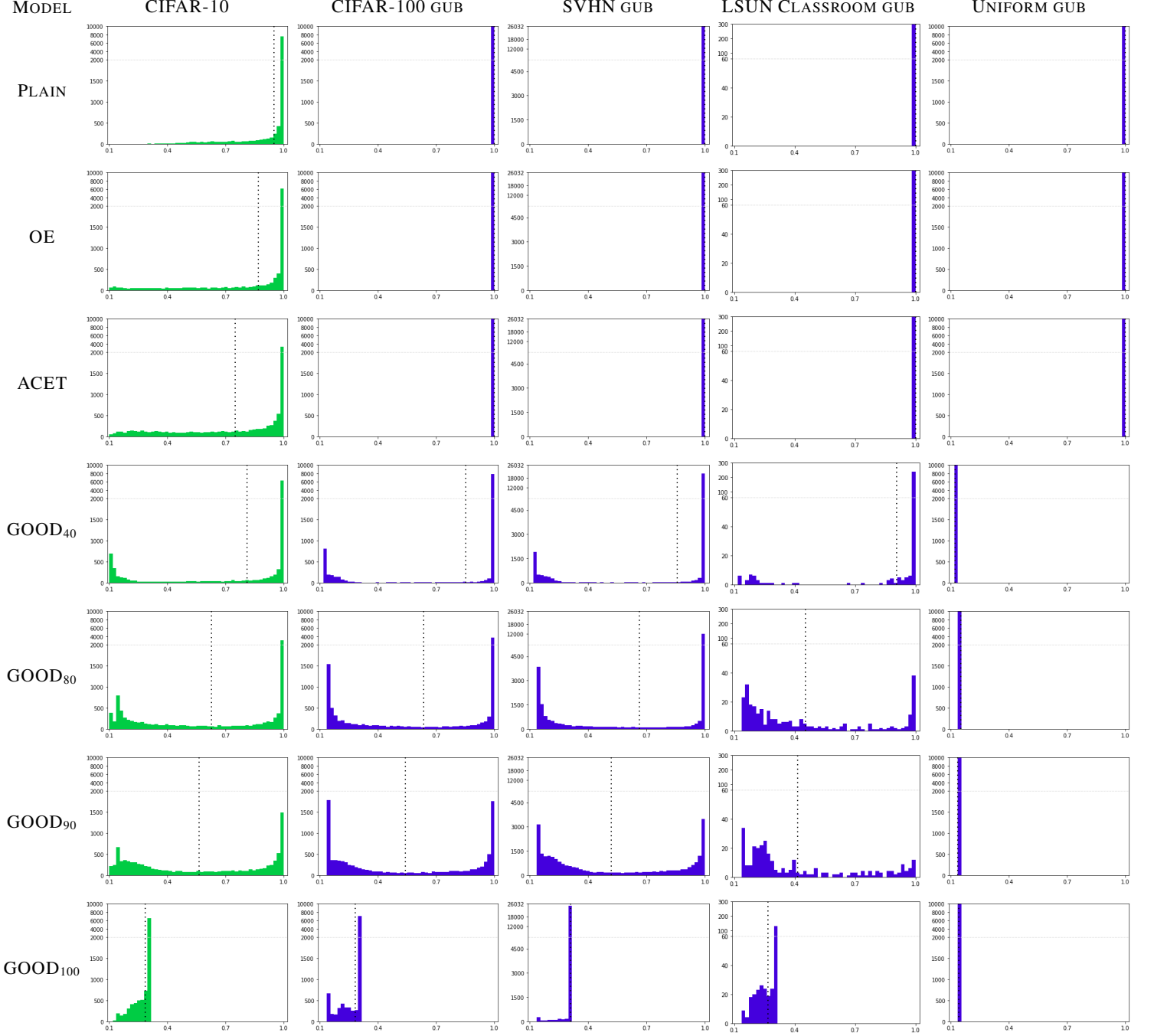


Table 7: Histograms of the confidences on the **CIFAR-10** in-distribution and guaranteed upper bounds on the confidences on OOD datasets within the  $l_\infty$ -ball of radius 0.01. Each histogram uses 50 bins between 0.1 and 1.0. For better readability, the scale is zoomed in by a factor 10 for numbers below one fifth of the total number of datapoints of the shown datasets. The vertical dotted line shows the mean value of the histogram’s data.



## H Evaluation on additional datasets

Table 8: A continuation of Table 1 for additional out-distributions. As in Table 1 the guaranteed AUCs (GAUC) of the highlighted GOOD models are in general better than the adversarial (AAUC) of OE (with the exception of Omniglot for MNIST).

IN: MNIST $\epsilon = 0.3$										
METHOD	ACC.	80M TINY IMAGES			OMNIGLOT			NOTMNIST		
		AUC	AAUC	GAUC	AUC	AAUC	GAUC	AUC	AAUC	GAUC
PLAIN	99.5	98.6	0.3	0.0	97.6	1.8	0.0	89.3	1.6	0.0
CEDA	99.5	<b>100.0</b>	88.0	0.0	98.5	23.5	0.0	99.9	98.4	0.0
OE	99.5	<b>100.0</b>	70.7	0.0	98.1	15.8	0.0	99.9	97.4	0.0
ACET	99.5	<b>100.0</b>	96.7	0.0	<b>99.3</b>	<b>79.6</b>	0.0	<b>100.0</b>	99.6	0.0
CCU	99.4	<b>100.0</b>	75.0	0.0	98.1	3.4	0.0	99.9	96.1	0.0
GOOD <sub>0</sub>	99.5	<b>100.0</b>	95.8	0.0	98.3	16.6	0.0	99.9	98.1	0.0
GOOD <sub>20</sub>	99.2	<b>100.0</b>	98.0	32.3	96.3	40.3	0.0	<b>100.0</b>	99.6	19.2
GOOD <sub>40</sub>	99.1	<b>100.0</b>	98.1	59.2	95.9	57.5	0.0	<b>100.0</b>	99.6	43.6
GOOD <sub>60</sub>	98.9	<b>100.0</b>	98.7	77.6	95.5	55.3	0.0	<b>100.0</b>	99.7	75.6
GOOD <sub>80</sub>	99.0	<b>100.0</b>	99.5	89.9	96.6	73.0	0.6	<b>100.0</b>	<b>100.0</b>	97.7
GOOD <sub>90</sub>	98.9	<b>100.0</b>	<b>99.7</b>	94.4	97.8	73.3	<b>1.4</b>	<b>100.0</b>	<b>100.0</b>	99.3
GOOD <sub>95</sub>	98.8	<b>100.0</b>	<b>99.7</b>	96.1	98.1	73.3	1.3	<b>100.0</b>	<b>100.0</b>	<b>99.5</b>
GOOD <sub>100</sub>	98.7	<b>100.0</b>	99.4	<b>97.9</b>	98.6	67.7	1.2	<b>100.0</b>	99.9	99.3

IN: SVHN $\epsilon = 0.01$										
METHOD	ACC.	80M TINY IMAGES			IMAGENET-			SMOOTH NOISE		
		AUC	AAUC	GAUC	AUC	AAUC	GAUC	AUC	AAUC	GAUC
PLAIN	95.5	95.3	60.7	0.0	95.5	61.0	0.0	95.8	36.7	0.0
CEDA	95.4	99.9	98.4	0.0	<b>100.0</b>	99.3	0.0	97.0	46.3	0.0
OE	95.5	<b>100.0</b>	97.8	0.0	<b>100.0</b>	98.7	0.0	96.7	44.7	0.0
ACET	95.6	<b>100.0</b>	<b>99.8</b>	0.0	<b>100.0</b>	<b>99.9</b>	0.0	<b>98.9</b>	77.1	0.0
CCU	95.7	<b>100.0</b>	97.9	0.0	<b>100.0</b>	99.3	0.0	97.1	67.3	0.0
GOOD <sub>0</sub>	97.0	<b>100.0</b>	95.9	0.0	<b>100.0</b>	97.9	0.0	98.1	38.2	0.0
GOOD <sub>20</sub>	96.5	99.9	99.0	20.5	99.9	99.7	33.3	97.9	63.2	0.0
GOOD <sub>40</sub>	96.1	99.9	98.9	44.1	<b>100.0</b>	99.7	62.3	98.0	69.2	0.0
GOOD <sub>60</sub>	95.8	99.9	99.5	66.1	<b>100.0</b>	<b>99.9</b>	82.2	98.2	74.9	0.0
GOOD <sub>80</sub>	95.9	99.6	99.4	88.4	99.6	99.6	95.6	98.1	72.1	0.0
GOOD <sub>90</sub>	96.3	99.9	99.7	96.1	<b>100.0</b>	<b>99.9</b>	99.0	98.4	79.9	2.3
GOOD <sub>95</sub>	96.2	99.8	99.7	98.0	99.8	99.8	99.5	97.9	73.1	3.4
GOOD <sub>100</sub>	96.6	99.9	<b>99.8</b>	<b>99.5</b>	99.9	<b>99.9</b>	<b>99.8</b>	97.5	<b>82.5</b>	<b>65.5</b>

IN: CIFAR-10 $\epsilon = 0.01$										
METHOD	ACC.	80M TINY IMAGES			IMAGENET-			SMOOTH NOISE		
		AUC	AAUC	GAUC	AUC	AAUC	GAUC	AUC	AAUC	GAUC
PLAIN	90.1	85.9	12.0	0.0	83.5	9.7	0.0	90.2	10.2	0.0
CEDA	87.6	96.3	61.1	0.0	89.3	31.5	0.0	99.1	90.4	0.0
OE	90.5	97.4	33.7	0.0	90.3	20.6	0.0	99.4	37.3	0.0
ACET	89.1	<b>96.8</b>	<b>89.1</b>	0.0	89.3	75.6	0.0	<b>99.5</b>	<b>96.8</b>	0.0
CCU	91.6	<b>96.8</b>	33.7	0.0	<b>92.0</b>	30.0	0.0	<b>99.5</b>	38.0	0.0
GOOD <sub>0</sub>	89.6	96.7	47.7	0.0	91.0	22.8	0.0	97.3	30.6	0.0
GOOD <sub>20</sub>	86.4	95.5	63.7	15.6	87.0	44.3	5.6	96.6	81.6	42.0
GOOD <sub>40</sub>	89.6	95.0	82.8	38.4	88.3	60.5	25.2	96.9	<b>96.8</b>	34.9
GOOD <sub>60</sub>	89.8	94.8	81.4	49.5	87.0	62.6	36.4	97.5	96.6	49.5
GOOD <sub>80</sub>	90.0	94.0	83.2	59.4	84.0	<b>65.3</b>	47.5	97.7	95.8	54.8
GOOD <sub>90</sub>	90.2	92.1	78.5	<b>64.1</b>	79.6	61.7	51.9	95.3	89.5	81.2
GOOD <sub>95</sub>	90.3	86.7	72.0	<b>64.1</b>	75.5	58.4	<b>52.0</b>	88.7	68.1	46.2
GOOD <sub>100</sub>	90.3	77.5	49.5	35.4	74.1	45.7	33.0	95.6	93.2	<b>92.8</b>

Extending the evaluation results presented in Table 1, we provide AUC, AAUC and GAUC values for additional out-distribution datasets in Table 8. These datasets are:

- 80M Tiny Images, the out-distribution that was used during training. While it is the same *distribution* as seen during training, the test set consists of 30,000 samples that are not part of the training set.
- Omniglot (Lake, B. M., Salakhutdinov, R., and Tenenbaum, J. B. (2015). Human-level concept learning through probabilistic program induction. *Science*, 350(6266), 1332-1338.) is a dataset of hand drawn characters. We use the evaluation split consisting of 13180 characters from 20 different alphabets.
- notMNIST is a dataset of the letters A to J taken from different publicly available fonts. The dataset was retrieved from <https://yaroslavvb.blogspot.com/2011/09/notmnist-dataset.html>. We evaluate on the hand cleaned subset of 18724 images,
- ImageNet- [16], which is a subset of ImageNet [10] without images labelled as classes equal or similar to CIFAR-10 classes.
- Smooth Noise is generated as described by [16]. First, a uniform noise image is generated. Then, a Gaussian filter with  $\sigma$  drawn uniformly at random between 1.0 and 2.5 is applied. Finally, the image is re-scaled such that the minimal pixel value is 0.0 and the maximal one is 1.0. We evaluate AUC and GAUC on 30,000 samples.

For MNIST, GOOD<sub>100</sub> has an excellent GAUC for the training out-distribution 80M Tiny images as well as for notMNIST. For Omniglot, GOOD<sub>100</sub> is again better than OE/CEDA (similar to EMNIST) in terms of clean AUC's but here ACET is slightly better. However, again it is very difficult to provide any guarantees for this dataset even though non-trivial adversarial AUC's are possible.

For SVHN, the detection of smooth noise turns out to be the most difficult of the evaluated tasks. There, the clean AUCs of all non-plain methods are lower than the perfect scores we see on other out-distributions but still very high, and only GOOD<sub>100</sub> can give some guarantees. An explanation might be that the image features of SVHN house numbers and of this kind of synthetic noise are similarly smooth. For 80M Tiny Images and Imagenet-, on the other hand, the SVHN high quantile GOOD models, particularly GOOD<sub>100</sub>, are able to provide almost perfect guaranteed AUCs.

For CIFAR-10, on all three out-distributions we again observe the trade-off between clean and guaranteed AUC that comes with the choice of the loss quantile. Overall, the GOOD<sub>80</sub> model again retains reasonable AUC values for the clean data while also providing useful guaranteed AUCs.

## I Generalization of provable confidence bounds to a larger radius

In Table 9, we evaluate the generalization of empirical worst case and guaranteed upper bound for the confidence within a larger  $l_\infty$ -ball around OOD samples than what the model was trained for.

As expected, the adversarial AUC's (AAUC) degrade for the larger radius  $\epsilon$ . However, we suspect that the seemingly stronger robustness of CEDA compared to OE could be partially due to the lack of gradients at the initialization points. As mentioned above in general attacking all OOD models is difficult and requires adaptive and transfer attacks to be successful. That said, the relative differences of the AAUC should still be meaningful. However, this shows even more that for worst-case OOD detection provable guarantees are particularly needed.

On MNIST, GOOD<sub>100</sub> not only still has a perfect guaranteed GAUC for uniform noise for an  $\epsilon$  of 0.4 but even on FashionMNIST and CIFAR-10 it still has substantial guarantees. Moreover, GOOD<sub>100</sub> has with the exception of FashionMNIST a better AAUC than that of ACET.

For SVHN the excellent guarantees of GOOD<sub>100</sub> for the radius of  $\epsilon = 0.01$  models do not generalize well to the significantly larger radius of  $\epsilon = 8/255$  (but note that this is more than three times as large as at training time). This is in particular the case for uniform noise where there are basically no guarantees anymore. Nevertheless the adversarial AAUC is still very high and better than that of ACET.

In contrast, for CIFAR-10 the generalization of the bounds of GOOD<sub>80</sub> to the larger radius of  $\epsilon = 8/255$  is surprisingly good: for all out-distributions, we only see an at most moderate drop of the GAUC value compared to Table 1. The same is true for the AAUC which is now significantly better than that of ACET whereas for the training radius of  $\epsilon = 0.01$  ACET had a better AAUC.

In summary, GOOD in most cases still achieves reasonable guarantees for the larger threat model at test time. Interestingly, the AAUC for the GOOD models is, with the exception of FashionMNIST, always better than that of ACET and thus our guaranteed IBP training shows in this regard a better generalization to larger evaluation radii than adversarial training on the out-distribution.

Table 9: Complementing Table 1, an evaluation of the generalization of worst-case OOD detection, that is AAUC and GAUC, for  $\epsilon$ -values larger than those of the threat models used during training.

IN: MNIST $\epsilon = 0.4$													
METHOD	ACC.	FASHIONMNIST			EMNIST LETTERS			CIFAR-10			UNIFORM NOISE		
		AUC	AAUC	GAUC	AUC	AAUC	GAUC	AUC	AAUC	GAUC	AUC	AAUC	GAUC
PLAIN	99.5	97.7	0.2	0.0	87.9	0.2	0.0	98.5	0.1	0.0	99.4	0.0	0.0
CEDA	99.5	100.0	71.0	0.0	92.8	8.9	0.0	100.0	80.2	0.0	100.0	<b>100.0</b>	0.0
OE	99.5	99.9	47.7	0.0	92.8	5.7	0.0	100.0	69.0	0.0	100.0	<b>100.0</b>	0.0
ACET	99.5	100.0	<b>96.7</b>	0.0	96.4	47.7	0.0	100.0	97.5	0.0	100.0	<b>100.0</b>	0.0
CCU	99.5	100.0	62.0	0.0	92.9	2.7	0.0	100.0	97.6	0.0	100.0	<b>100.0</b>	0.0
GOOD <sub>0</sub>	99.5	99.9	57.6	0.0	92.8	5.9	0.0	100.0	80.1	0.0	100.0	<b>100.0</b>	0.0
GOOD <sub>20</sub>	99.2	99.8	79.7	3.6	95.2	36.5	0.0	100.0	91.7	5.9	100.0	<b>100.0</b>	99.8
GOOD <sub>40</sub>	99.1	99.8	86.7	18.1	95.7	48.0	0.0	100.0	94.3	26.1	100.0	<b>100.0</b>	<b>100.0</b>
GOOD <sub>60</sub>	98.9	99.9	89.5	32.1	96.6	55.0	0.0	100.0	94.7	55.1	100.0	<b>100.0</b>	<b>100.0</b>
GOOD <sub>80</sub>	99.0	99.9	93.9	42.6	97.8	61.1	1.3	100.0	97.8	77.6	100.0	<b>100.0</b>	<b>100.0</b>
GOOD <sub>90</sub>	98.9	99.9	93.5	46.1	98.2	<b>61.4</b>	<b>1.9</b>	100.0	98.5	79.2	100.0	<b>100.0</b>	<b>100.0</b>
GOOD <sub>95</sub>	98.8	99.9	93.0	48.1	98.6	56.4	1.3	100.0	98.8	<b>79.4</b>	100.0	<b>100.0</b>	<b>100.0</b>
GOOD <sub>100</sub>	98.7	100.0	94.4	<b>49.6</b>	98.9	48.6	0.8	100.0	<b>99.0</b>	73.8	100.0	<b>100.0</b>	<b>100.0</b>

IN: SVHN $\epsilon = 8/255$													
METHOD	ACC.	CIFAR-100			CIFAR-10			LSUN CLASSROOM			UNIFORM NOISE		
		AUC	AAUC	GAUC	AUC	AAUC	GAUC	AUC	AAUC	GAUC	AUC	AAUC	GAUC
PLAIN	95.5	94.9	4.3	0.0	95.2	9.9	0.0	95.7	6.5	0.0	99.4	57.9	0.0
CEDA	95.4	99.9	76.1	0.0	99.9	80.8	0.0	100.0	89.9	0.0	100.0	99.9	0.0
OE	95.5	100.0	57.4	0.0	100.0	61.7	0.0	100.0	76.4	0.0	100.0	98.8	0.0
ACET	95.6	100.0	92.2	0.0	100.0	93.2	0.0	100.0	98.0	0.0	99.7	70.5	0.0
CCU	95.7	100.0	63.5	0.0	100.0	67.4	0.0	100.0	79.6	0.0	100.0	99.2	0.0
GOOD <sub>0</sub>	97.0	100.0	43.1	0.0	100.0	43.6	0.0	100.0	39.3	0.0	100.0	88.7	0.0
GOOD <sub>20</sub>	96.5	99.9	78.7	0.2	99.9	84.1	0.0	99.9	90.8	0.0	99.9	99.9	0.0
GOOD <sub>40</sub>	95.8	99.7	84.0	0.7	99.7	89.2	0.2	99.7	94.4	0.3	99.7	99.7	0.0
GOOD <sub>60</sub>	95.8	99.9	92.6	1.5	100.0	94.3	0.8	100.0	96.6	0.3	100.0	<b>100.0</b>	0.0
GOOD <sub>80</sub>	95.9	99.6	<b>98.5</b>	3.2	99.6	<b>98.9</b>	2.0	99.6	99.6	1.3	99.6	99.6	0.0
GOOD <sub>90</sub>	96.3	99.9	96.7	11.8	99.9	98.0	11.3	100.0	<b>99.9</b>	11.5	100.0	99.6	0.0
GOOD <sub>95</sub>	96.2	99.8	97.7	17.8	99.8	<b>98.9</b>	16.1	99.8	99.8	16.4	99.8	99.8	0.0
GOOD <sub>100</sub>	96.6	99.9	98.0	<b>40.3</b>	100.0	98.8	<b>41.3</b>	100.0	99.8	<b>40.3</b>	100.0	<b>100.0</b>	<b>1.5</b>

IN: CIFAR-10 $\epsilon = 8/255$													
METHOD	ACC.	CIFAR-100			SVHN			LSUN CLASSROOM			UNIFORM NOISE		
		AUC	AAUC	GAUC	AUC	AAUC	GAUC	AUC	AAUC	GAUC	AUC	AAUC	GAUC
PLAIN	90.1	84.3	0.1	0.0	87.7	8.6	0.0	88.9	0.0	0.0	90.7	3.4	0.0
CEDA	87.6	90.9	13.9	0.0	97.6	14.1	0.0	98.0	22.0	0.0	98.0	41.1	0.0
OE	90.5	92.2	1.3	0.0	98.1	9.8	0.0	98.9	0.3	0.0	97.6	6.2	0.0
ACET	89.1	90.5	39.1	0.0	96.9	53.6	0.0	98.5	51.0	0.0	97.9	68.7	0.0
CCU	91.6	93.0	1.6	0.0	97.1	11.2	0.0	99.3	0.4	0.0	100.0	<b>100.0</b>	0.0
GOOD <sub>0</sub>	89.6	93.0	10.4	0.0	97.7	22.8	0.0	98.1	29.0	0.0	97.7	51.9	0.0
GOOD <sub>20</sub>	86.4	88.3	16.1	2.3	94.3	29.7	2.8	97.6	5.0	0.0	94.8	90.7	0.0
GOOD <sub>40</sub>	89.6	89.6	55.5	15.5	96.1	<b>83.2</b>	16.9	96.4	<b>89.3</b>	11.2	93.2	91.7	91.4
GOOD <sub>60</sub>	89.8	88.7	56.7	25.7	95.8	74.4	22.3	96.5	87.5	35.9	91.5	90.2	89.3
GOOD <sub>80</sub>	90.0	85.9	<b>61.3</b>	36.7	95.6	79.0	34.4	96.2	86.6	48.4	90.4	87.5	86.9
GOOD <sub>90</sub>	90.2	81.8	57.5	<b>42.2</b>	91.0	65.9	<b>43.8</b>	94.5	81.3	<b>50.1</b>	93.8	92.3	91.9
GOOD <sub>95</sub>	90.3	77.5	52.8	41.7	89.6	56.8	38.8	84.6	56.4	43.1	98.4	98.1	97.6
GOOD <sub>100</sub>	90.3	73.9	45.5	23.4	74.3	29.2	2.1	82.7	57.0	43.3	99.1	98.7	<b>98.6</b>