TOWARDS NEURAL NETWORKS THAT PROVABLY KNOW WHEN THEY DON'T KNOW

Anonymous authors
Paper under double-blind review

ABSTRACT

It has recently been shown that ReLU networks produce arbitrarily over-confident predictions far away from the training data. Thus, ReLU networks do not know when they don't know. However, this is a highly important property in safety critical applications. In the context of out-of-distribution detection (OOD) there have been a number of proposals to mitigate this problem but none of them are able to make any mathematical guarantees. In this paper we propose a new approach to OOD which overcomes both problems. Our approach can be used with ReLU networks and provides provably low confidence predictions far away from the training data as well as the first certificates for low confidence predictions in a neighborhood of an out-distribution point. In the experiments we show that state-of-the-art methods fail in this worst-case setting whereas our model can guarantee its performance while retaining state-of-the-art OOD performance.

1 INTRODUCTION

Deep Learning Models are being deployed in a growing number of applications. As these include more and more systems where safety is a concern, it is important to guarantee that deep learning models work as one expects them to. One topic that has received a lot of attention in this area is the problem of adversarial examples, in which a model's prediction can be changed by introducing a small perturbation to an originally correctly classified sample. Achieving robustness against this type of perturbation is an active field of research. Empirically, adversarial training (Madry et al., 2018) performs well and provably robust models have been developed (Hein & Andriushchenko, 2017; Wong & Kolter, 2018; Raghunathan et al., 2018; Mirman et al., 2018; Cohen et al., 2019).

On the other end of the spectrum it is also important to study how deep learning models behave far away from the training samples. A simple property every classifier should satisfy is that far away from the training data, it should yield close to uniform confidence over the classes: it knows when it does not know. However, several cases of high confidence predictions far away from the training data have been reported for neural networks, e.g. fooling images (Nguyen et al., 2015), for out-of-distribution (OOD) images (Hendrycks & Gimpel, 2017) or in medical diagnosis (Leibig et al., 2017). Moreover, it has been observed that, even on the original task, neural networks often produce overconfident predictions (Guo et al., 2017). Very recently, it has been shown theoretically that the class of ReLU networks (all neural networks which use a piecewise affine activation function), which encompasses almost all standard models, produces predictions with arbitrarily high confidences far away from the training data (Hein et al., 2019). Unfortunately, this statements holds for almost all such networks and thus without a change in the architecture one cannot avoid this phenomenon.

Traditionally, the calibration of the confidence of predictions has been considered on the in-distribution (Guo et al., 2017; Lakshminarayanan et al., 2017). However these techniques cannot be used for OOD techniques (Leibig et al., 2017). Only recently the detection of OOD inputs (Hendrycks & Gimpel, 2017) has been tackled. The existing approaches are roughly of two types: first, postprocessing techniques that adjust the estimated confidence (DeVries & Taylor, 2018; Liang et al., 2018) which includes the baseline ODIN. Second, modification of the classifier training by integrating generative models like a VAE or GAN in order to discriminate out-distribution from in-distribution data (Lee et al., 2018a; Wang et al., 2018; Lee et al., 2018b) or approaches which enforce low confidence on OOD inputs during training (Hein et al., 2019; Hendrycks et al., 2019). Worst-case aspects of OOD detection have previously been studied in Nguyen et al. (2015); Schott et al. (2018);

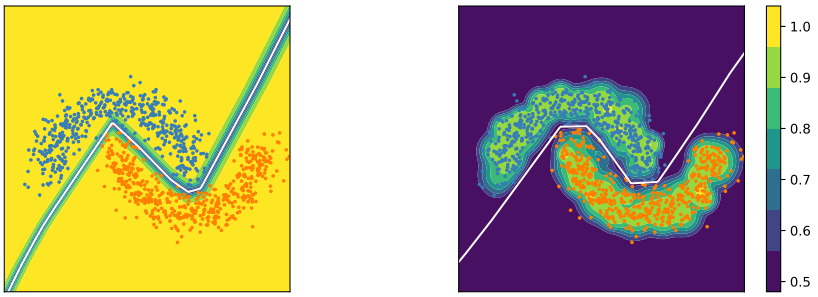

Figure 1: **Illustration on toy dataset:** We show the color-coded confidence in the prediction (yellow indicates high confidence $\max_y \hat{p}(y|x) \approx 1$, whereas dark purple regions indicate low confidence $\max_y \hat{p}(y|x) \approx 0.5$) for a normal neural network (left) and our CCU neural network (right). The decision boundary is shown in white which is similar for both models. Our CCU-model retains high-confidence predictions in regions close to the training data, whereas far away from the training the CCU-model outputs close to uniform confidence. In contrast the normal neural network is over-confident everywhere except very close to the decision boundary.

Hein et al. (2019); Sehwag et al. (2019), but no robustness guarantees have yet been proposed for this setting. In particular, none of those approaches are able to guarantee that neural networks produce low confidence predictions far away from the training data. We prove that our classifier satisfies this requirement even when we use ReLU networks as the classifier model - without loosing performance on either the prediction task on the in-distribution nor the OOD detection performance, see Figure 1 for an illustration. Moreover, our technique allows to give upper bounds on the confidence over a whole neighborhood around a point (worst-case guarantees). We show that most state-of-the-art OOD methods can be fooled by maximizing the confidence in this ball even when starting from uniform noise images, which should be trivial to identify.

2 A GENERIC MODEL FOR CLASSIFIERS WITH CERTIFIED LOW CONFIDENCE FAR AWAY FROM THE TRAINING DATA

The model which we propose in this paper assumes that samples from an out-distribution are given to us. In image recognition we could either see the set of all images as a sample from the out-distribution (Hendrycks et al., 2019) on all images or the agnostic case where we use use uniform noise on $[0, 1]^d$ as a maximally uninformative out-distribution. In both settings one tries to discriminate these out-distribution images from images coming from a particular image recognition task and the task is to get low confidence predictions on the out-distribution images vs. higher confidence on the images from the actual task. From the general model we derive under minimal assumptions a maximum-likelihood approach where one trains both a classifier for the actual task and density estimators for in- and out-distribution jointly. As all of these quantities are coupled in our model for the conditional distribution $\mathrm{P}(y|x)$ we get guarantees by controlling the density estimates far away from the training data. This is a crucial difference to the approaches of Lee et al. (2018a); Wang et al. (2018); Hendrycks et al. (2019) which empirically yield good OOD performance but are not able to certify the detection mechanism.

2.1 A PROBABILISTIC MODEL FOR IN- AND OUT-DISTRIBUTION DATA

We assume that there exists a joint probability distribution $p(y, x)$ over the in- and out-distribution data, where y are the labels in $\{1, \dots, M\}$, M is the number of classes, and $x \in \mathbb{R}^d$, where d is the input dimension. In the following, we denote the underlying probabilities/densities with $p(y|x)$ resp. $p(x)$ and the estimated quantities with $\hat{p}(y|x)$ and $\hat{p}(x)$. We are mainly interested in a discriminative framework, i.e. we want to estimate $p(y|x)$ which one can represent via the conditional distribution of the in-distribution $p(y|x, i)$ and out-distribution $p(y|x, o)$:

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

Note that at first it might seem strange to have a conditional distribution $p(y|x, o)$ for out-distribution data, but until now we have made no assumptions about what in-and out-distribution are. A realistic

scenario would be that at test time we are presented with instances x from other classes (out-distribution) for which we expect a close to uniform $p(y|x, o)$.

Our model for $\hat{p}(y|x)$ has the same form as $p(y|x)$

$$\hat{p}(y|x) = \frac{\hat{p}(y|x, i)\hat{p}(x|i)\hat{p}(i) + \hat{p}(y|x, o)\hat{p}(x|o)\hat{p}(o)}{\hat{p}(x|i)\hat{p}(i) + \hat{p}(x|o)\hat{p}(o)}. \tag{2}$$

Typically, out-distribution data has no relation to the actual task and thus we would like to have uniform confidence over the classes. Therefore we set in our model

$$\hat{p}(y|x, o) = \frac{1}{M} \quad \text{and} \quad \hat{p}(y|x, i) = \frac{e^{f_y(x)}}{\sum_{k=1}^{M} e^{f_k(x)}}, \quad y \in \{1, \dots M\}, \tag{3}$$

where $f : \mathbb{R}^d \to \mathbb{R}^M$ is the classifier function (logits). This framework is generic for classifiers trained with the cross-entropy (CE) loss (as the softmax function is the correct link function for the CE loss) and we focus in particular on neural networks. For a ReLU network the classifier function f is componentwise a continuous piecewise affine function and has been shown to produce asymptotically arbitrarily highly confident predictions (Hein et al., 2019), i.e. the classifier gets more confident in its predictions the further it moves away from its training data. One of the main goals of our proposal is to fix this behavior of neural networks in a provable way.

Note that with the choice of $\hat{p}(y|x, o)$ and non-zero priors for $\hat{p}(i), \hat{p}(o)$, the full model $\hat{p}(y|x)$ can be seen as a calibrated version of $\hat{p}(y|x, i)$, where $\hat{p}(y|x) \approx \hat{p}(y|x, i)$ for inputs with $\hat{p}(x|i) \gg \hat{p}(x|o)$ and $\hat{p}(y|x) \approx \frac{1}{M}$ if $\hat{p}(x|i) \ll \hat{p}(x|o)$. However, note that only the confidence in the prediction $\hat{p}(y|x)$ is affected, the classifier decision is still done according to $\hat{p}(y|x, i)$ as the calibration does not change the ranking. Thus even if the OOD data came from the classification task we would like to solve, the trained classifier's performance would be unaffected, only the confidence in the prediction would be damped.

For the marginal out-distribution $\hat{p}(x|o)$ there are two possible scenarios. In the first case one could concentrate on the worst case where we assume that $p(x|o)$ is maximally uniformative (maximal entropy). This means that $\hat{p}(x|o)$ is uniform for bounded domains e.g. for images which are in $[0, 1]^d$, $\hat{p}(x|o) = 1$ for all $x \in [0, 1]^d$, or $\hat{p}(x|o)$ is a Gaussian for the domain of $\mathbb{R}^d$ (the Gaussian has maximum entropy among all distributions of fixed variance). However, in this work we follow the approach of Hendrycks et al. (2019) where they used the 80 million tiny image dataset (Torralba et al., 2008) as a proxy of all possible images. Thus we estimate the density of $\hat{p}(x|o)$ using this data.

In order to get guarantees, the employed generative models for $\hat{p}(x|i)$ and $\hat{p}(x|o)$ have to be chosen in a way that allows one to control predictions far away from the training data. Variational autoencoders (VAEs) (Kingma & Welling, 2014; Rezende et al., 2014), normalizing flows (Dinh et al., 2016; Kingma & Dhariwal, 2018) and generative adversarial networks (GANs) (Goodfellow et al., 2014) are powerful generative models. However, there is no direct way to control the likelihood far away from the training data. Moreover, it has recently been discovered that VAEs, flows and GANs also suffer from overconfident likelihoods (Nalisnick et al., 2019; Hendrycks et al., 2019) far away from the data they are supposed to model as well as adversarial samples (Kos et al., 2017).

For $\hat{p}(x|o)$ and $\hat{p}(x|i)$ we use a Gaussian mixture model (GMM) which is less powerful than a VAE but has the advantage that the density estimates can be controlled far away from the training data:

$$\hat{p}(x|i) = \sum_{k=0}^{K_i} \alpha_k \exp\left(-\frac{d(x, \mu_k)^2}{2\sigma_k^2}\right), \qquad \hat{p}(x|o) = \sum_{l=0}^{K_o} \beta_l \exp\left(-\frac{d(x, \nu_l)^2}{2\theta_l^2}\right) \tag{4}$$

where $d : \mathbb{R}^d \times \mathbb{R}^d \to \mathbb{R}$ is the metric

$$d(x, y) = \left\| C^{-\frac{1}{2}}(x - y) \right\|_2.$$

with C being a positive definite matrix and

$$\alpha_k = \frac{1}{K_i} \frac{1}{(2\pi\sigma_k^2 \det C)^{\frac{d}{2}}}, \quad \beta_l = \frac{1}{K_o} \frac{1}{(2\pi\theta_l^2 \det C)^{\frac{d}{2}}}.$$

We later fix C as a slightly modified covariance matrix of the in-distribution data (see Section 4 for details). Thus one just has to estimate the centroids μ_k, ν_l and the variances σ_k^2, θ_l^2. The idea of this metric is to use distances adapted to the data-distribution. Note that equation 4 is a properly normalized density in $\mathbb{R}^d$.

2.2 MAXIMUM LIKELIHOOD ESTIMATION

Given models for $\hat{p}(y|x)$ and $\hat{p}(x)$ we effectively have a full generative model and apply maximum likelihood estimation to get the underlying classifier $\hat{p}(y|x,i)$ and the parameters of the Gaussian mixture models $\hat{p}(x|i), \hat{p}(x|o)$. The only free parameter left is the probability $\hat{p}(i), \hat{p}(o)$ which we write compactly as $\lambda = \frac{\hat{p}(o)}{\hat{p}(i)}$. In principle this parameter should be set considering the potential cost of over-confident predictions. In our experiments we simply fix it to $\lambda = 1$.

$$\mathop{\mathbb{E}}_{(x,y)\sim p(x,y)} \log\Big(\hat{p}(y,x)\Big) = \mathop{\mathbb{E}}_{(x,y)\sim p(x,y)} \log\big(\hat{p}(y|x)\big) + \log(\hat{p}(x)),$$

$$= \mathop{\mathbb{E}}_{(x,y)\sim p(x,y)} \log\left(\frac{\hat{p}(y|x,i)\hat{p}(x|i)\hat{p}(i) + \frac{1}{M}\hat{p}(x|o)\hat{p}(o)}{\hat{p}(x|i)\hat{p}(i) + \hat{p}(x|o)\hat{p}(o)}\right) + \log\big(\hat{p}(x|i)\hat{p}(i) + \hat{p}(x|o)\hat{p}(o)\big). \quad (5)$$

In practice, we have to compute empirical expectations from finite training data from the in-distribution $(x_i, y_i)_{i=1}^{n_i}$ and out-distribution $(z_j)_{j=1}^{n_o}$. Labels for the out-distribution could be generated randomly via $p(y|x,o) = \frac{1}{M}$, but we obtain an unbiased estimator with lower variance by averaging over all classes directly, as was done in Lee et al. (2018a); Hein et al. (2019); Hendrycks et al. (2019). Now we can estimate the classifier f and the mixture model parameters μ, ν, σ, θ via

$$\arg\max_{f,\mu,\nu,\sigma,\theta} \left\{ \frac{1}{n_i}\sum_{i=1}^{n_i} \log\big(\hat{p}(y_i|x_i)\big) + \frac{\lambda}{n_o}\sum_{j=1}^{n_o}\frac{1}{M}\sum_{m=1}^{M}\log\big(\hat{p}(m|z_j)\big) \right. \quad (6)$$

$$\left. + \frac{1}{n_i}\sum_{i=1}^{n_i}\log(\hat{p}(x_i)) + \frac{\lambda}{n_o}\sum_{j=1}^{n_o}\log(\hat{p}(z_j)) \right\}, \quad (7)$$

with

$$\hat{p}(y|x) = \frac{\hat{p}(y|x,i)\hat{p}(x|i) + \frac{\lambda}{M}\hat{p}(x|o)}{\hat{p}(x|i) + \lambda\hat{p}(x|o)} \quad \text{and} \quad \hat{p}(x) = \frac{1}{\lambda+1}\Big(\hat{p}(x|i) + \lambda\hat{p}(x|o)\Big). \quad (8)$$

Due to the bounds derived in Section 3, we denote our method by **Certified Certain Uncertainty (CCU)**. Note that if one uses a standard neural network model with softmax, i.e. $\hat{p}(y|x) = \hat{p}(y|x,i) = \frac{e^{f_y(x)}}{\sum_{m=1}^{M} e^{f_m(x)}}$, then the first term in equation 6 would be the cross-entropy loss for the in-distribution data and the second term the cross entropy loss for the out-distribution data with a uniform distribution over the classes. For this choice of $\hat{p}(y|x)$ and neglecting the terms for $\hat{p}(x)$ we recover the approach of Hein et al. (2019); Hendrycks et al. (2019) for training a classifier which outputs uniform confidence predictions on out-distribution data where $\frac{\hat{p}(i)}{\hat{p}(o)}$ corresponds to that regularization parameter λ. The key difference in our approach is that $\hat{p}(y|x) \neq \hat{p}(y|x,i)$ and the estimated densities for in- and out distribution $\hat{p}(x|i)$ and $\hat{p}(x|o)$ lead to a confidence calibration of $\hat{p}(y|x)$, and in turn the fit of the classifier influences the estimation of $\hat{p}(x|i)$ and $\hat{p}(x|o)$. The major advantage of our model is that we can give guarantees on the confidence of the classifier decision far away from the training data.

3 PROVABLE GUARANTEES FOR CLOSE TO UNIFORM PREDICTIONS FAR AWAY FROM THE TRAINING DATA

In this section we provide two types of guarantees on the confidence of a classifier trained according to our model in equation 6. The first one says that the classifier has provably low confidence far away from the training data, where an explicit bound on the minimal distance is provided, and the second provides an upper bound on the confidence in a ball around a given input point. The latter bound resembles robustness guarantees for adversarial samples (Hein & Andriushchenko, 2017; Wong & Kolter, 2018; Raghunathan et al., 2018; Mirman et al., 2018) and is quite different from the purely empirical evaluation done in OOD detection papers as we show in Section 4.

We provide our bounds for a more general mixture model which includes our GMM in equation 4 as a special case. Up to our knowledge, these are the first such bounds for neural networks and thus it is the first modification of a ReLU neural network so that it provably "knows when it does not know" (Hein et al., 2019) in the sense that far away from the training data the predictions are close to uniform over the classes.

Theorem 3.1. *Let $(x_i^{(i)}, y_i^{(i)})_{i=1}^n$ be the training set of the in-distribution and let the model for the conditional probability be given as*

$$\forall x \in \mathbb{R}^d,\, y \in \{1, \dots, M\}, \qquad \hat{p}(y|x) = \frac{\hat{p}(y|x,i)\hat{p}(x|i) + \frac{\lambda}{M}\hat{p}(x|o)}{\hat{p}(x|i) + \lambda \hat{p}(x|o)}, \tag{9}$$

where $\lambda = \frac{\hat{p}(o)}{\hat{p}(i)} > 0$ and let the model for the marginal density of the in-distribution $\hat{p}(x|i)$ and out-distribution $p(x|o)$ be given by the generalized GMMs

$$\hat{p}(x|i) = \sum_{k=0}^{K_i} \alpha_k \exp\left(-\frac{d(x,\mu_k)^2}{2\sigma_k^2}\right), \qquad \hat{p}(x|o) = \sum_{l=0}^{K_o} \beta_l \exp\left(-\frac{d(x,\nu_l)^2}{2\theta_l^2}\right)$$

with $\alpha_k, \beta_l > 0$ and $\mu_k, \nu_l \in \mathbb{R}^d \;\; \forall k = 1, \dots K_i,\; l = 1, \dots, K_o$ and $d : \mathbb{R}^d \times \mathbb{R}^d \to \mathbb{R}_+$ a metric. Let $z \in \mathbb{R}^d$ and define $k^* = \underset{k=1,\dots,K_i}{\arg\min} \frac{d(z,\mu_k)}{\sigma_k}$, $i^* = \underset{i=1,\dots,n}{\arg\min}\, d(z, x_i)$, $l^* = \underset{l=1,\dots,K_o}{\arg\min}\, \beta_l \exp\left(-\frac{d(z,\nu_l)^2}{2\theta_l^2}\right)$ and $\Delta = \frac{\theta_{l^*}^2}{\sigma_{k^*}^2} - 1$. For any $\epsilon > 0$, if $\min_l \theta_l > \max_k \sigma_k$ and

$$\min_{i=1,\dots,n} d(z, x_i) \geq d(x_{i^*}, \mu_{k^*}) + d(\mu_{k^*}, \nu_{l^*}) \Big[\frac{2}{\Delta} + \frac{1}{\sqrt{\Delta}}\Big] + \frac{\theta_{l^*}}{\sqrt{\Delta}} \sqrt{\log\Big(\frac{\epsilon\,\lambda}{M-1} \frac{\beta_{l^*}}{\sum_k \alpha_k}\Big)}, \tag{10}$$

then it holds for all $m \in \{1, \dots, M\}$ that

$$\hat{p}(m|z) \leq \frac{1}{M}\big(1 + \epsilon\big). \tag{11}$$

In particular, if $\min_i d(z, x_i) \to \infty$, then $\hat{p}(m|z) \to \frac{1}{M}$.

The proof is in the Appendix A. Theorem 3.1 holds for any multi-class classifier which defines for each input a probability distribution over the labels. Given the parameters of the GMM's it quantifies at which distance of an input z to the training set the classifier achieves close to uniform confidence. The theorem holds even if we use ReLU classifiers which in their unmodified form have been shown to produce arbitrarily high confidence far away from the training data Hein et al. (2019). This is a first step towards neural networks which provably know when they don't know.

In the next corollary, we provide an upper bound on the confidence over a ball around a given data point. This allows to give "confidence guarantees" for a whole volume and thus is much stronger than the usual pointwise evaluation of OOD methods.

Corollary 3.1. *Let $x_0 \in \mathbb{R}^d$ and $R > 0$, then with $\lambda = \frac{\hat{p}(o)}{\hat{p}(i)}$ it holds*

$$\max_{d(x,x_0) \leq R} \hat{p}(y|x) \;\leq\; \frac{1}{M} \frac{1 + M\frac{b}{\lambda}}{1 + \frac{b}{\lambda}}, \tag{12}$$

where $b = \dfrac{\sum_{k=1}^{K_i} \alpha_k \exp\Big(-\frac{\max\{d(x_0,\mu_k) - R, 0\}^2}{2\sigma_k^2}\Big)}{\sum_{l=1}^{K_o} \beta_l \exp\Big(-\frac{(d(x_0,\nu_l)+R)^2}{2\theta_l^2}\Big)}$.

The proof is in the Appendix B. We show in Section 4 that even though OOD methods achieve low confidence on noise images, the maximization of the confidence in a ball around a noise point (adversarial noise) yields high confidence predictions for OOD methods, whereas our classifier has provably low confidence, as certified by Corollary 3.1. The failure of OOD methods shows that the certification of entire regions is an important contribution of CCU which goes beyond the purely sampling-based evaluation.

4 Experiments

We evaluate the worst-case performance of various OOD detection methods within regions for which CCU yields guarantees and by standard OOD on MNIST (LeCun et al., 1998), FashionMNIST (Xiao et al., 2017), SVHN (Netzer et al., 2011) and CIFAR10 (Krizhevsky & Hinton, 2009). We show that all other OOD methods yield undesired high confidence predictions in the certified low confidence regions of CCU and thus would not detect these inputs as out-distribution. For calibrating hyper-parameters resp. training we use for all OOD methods the 80 Million Tiny Images (Torralba et al., 2008) as out-distribution Hendrycks et al. (2019) which yields a fair and realistic comparison.

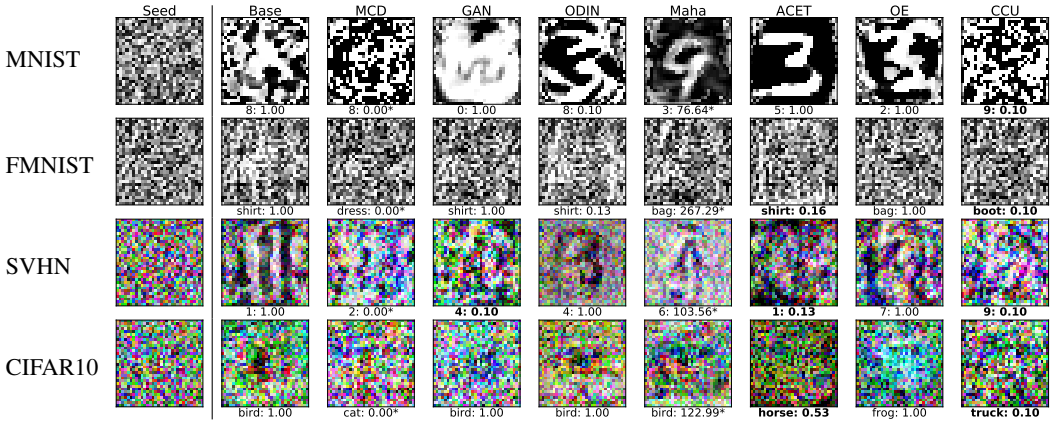

Figure 2: **Adversarial Noise:** We maximize the confidence of the OOD methods using PGD in the ball around a uniform noise sample (seed images, left) on which CCU is guaranteed by Corollary 3.1 to yield less than $1.1\frac{1}{M}$ maximal confidence. For each OOD method we report the image with the highest confidence. Maha and MCD use scores where lower is more confident (indicated by $*$). If we do *not* find a sample that has higher confidence/lower score than the mean of the in-distribution, we highlight this in boldface. All other OOD methods fail on some dataset, see Table 1 for a quantitative version. Arguably ACET produces good samples on MNIST, but the sample is classified as a 5. ODIN at high temperatures always returns low confidence, so a value of 0.1 is not informative.

CCU: As the Euclidean metric is known to be a relatively bad distance between two images we instead use the distance $d(x, y) = \left\| C^{-\frac{1}{2}}(x - y) \right\|$, where C is generated as follows. We calculate the covariance matrix C' on augmented in-distribution samples (see C.1). Let $(\lambda_i, u_i)_{i=1}^d$ be the eigenvalues/eigenvectors of C'. Then we set

$$C = \sum_{i=1}^{d} \max\{\lambda_i, 10^{-6} \max_j \lambda_j\} u_i u_i^T, \tag{13}$$

that is we fix a lower bound on the smallest eigenvalue so that C has full rank. This metric strongly penalizes moving away from the in-distribution. We choose $K_i = K_o = 100$ as the number of centroids for the GMMs. We initialize the in-GMM on augmented in-data using the EM algorithm with spherical covariance matrices in the transformed space, as in equation 4. For the out-distribution we use a subset of 20000 points for the initialization. While, initially it holds that $\forall k, l : \sigma_k < \theta_l$, as required in Theorem 3.1, this is not guaranteed during the optimization of equation 6. Thus, we enforce the constraint during training by setting: $\theta_l \mapsto \max\{\theta_l, 2 \max_k \sigma_k\}$ at every gradient step. Since the "classifier" and "density" terms in equation 6 have very different magnitudes we choose a small learning rate of $1e-5$ for the parameters in the GMMs. It is also crucial to not apply weight decay to these parameters. The other hyperparameters are chosen as in the base model below.

Benchmarks: For all OOD methods we use LeNet on MNIST and a Resnet18 (for GAN and MCD we use VGG) otherwise. The hyperparameters used during training can be found in Appendix C. The AUC is computed by treating in-distribution versus out-distribution as a two-class problem using the confidence/score of the method as criterion. ***MCD:*** Monte-Carlo Dropout (Gal & Ghahramani, 2016) uses dropout at train and at test time. Since it is not clear where to put the dropout layers in a ResNet, we use VGG instead. We take the softmax from 7 forward passes (Shafaei et al., 2018) and use the mean of the output for prediction and the variance as score. ***GAN:*** The framework of confidence-calibrated classifiers (Lee et al., 2017) relies on training a GAN alongside a classifier such that the GAN's generator is encouraged to generate points close to but not on the in-distribution. On these points one then enforces uniform confidence. We used their provided code to train a VGG this way, as we were unable to adapt the method to a ResNet with an acceptable test error (e.g. TE$< 30\%$ on SVHN). ***ODIN:*** ODIN (Liang et al., 2017) consists of two parts: a temperature T by which one rescales the logits before the softmax layer $\frac{e^{f_n/T}}{\sum_k e^{f_k/T}}$ and a preprocessing step that applies a single FGSM-step (Goodfellow et al., 2015) of length ϵ before evaluating the input. The two

		Base	MCD	GAN	ODIN	Maha	ACET	OE	CCU
MNIST	TE	0.5	0.4	0.8	0.5	0.5	0.5	0.7	0.4
	SR	100.0	100.0	62.5	100.0	99.5	90.5	100.0	**0.0**
	AUC	4.1	5.3	41.2	0.0	22.9	28.3	35.2	**100.0**
FMNIST	TE	4.9	5.6	6.3	4.9	4.9	4.9	5.7	4.9
	SR	100.0	99.5	100.0	100.0	100.0	**0.0**	100.0	**0.0**
	AUC	0.0	33.5	35.9	0.0	24.0	**100.0**	35.7	**100.0**
SVHN	TE	3.0	3.9	4.2	3.0	3.0	3.0	4.1	3.0
	SR	100.0	99.0	**0.0**	100.0	100.0	**0.0**	100.0	**0.0**
	AUC	0.0	23.3	**100.0**	0.0	0.0	**100.0**	13.5	**100.0**
CIFAR10	TE	5.7	12.0	11.7	5.7	5.7	5.6	4.7	5.8
	SR	100.0	98.0	100.0	100.0	100.0	**0.0**	100.0	**0.0**
	AUC	0.0	17.2	25.3	0.0	0.0	**100.0**	0.4	**100.0**

Table 1: Worst-case performance of different OOD methods in neighborhoods around uniform noise points certified by CCU. We report the clean test error (TE) on the in-distribution (GAN and MCD use VGG). The success rate (SR) is the fraction of adversarial noise points for which the confidence/score inside the ball is higher than the median of the in-distribution's confidence/score. The AUC quantifies detection of adversarial noise versus in-distribution. All values in %.

parameters are calibrated on the out-distribution. ***Maha:*** The approach in Lee et al. (2018c) is based on computing a class-conditional Mahalanobis distance in feature space and applying an ODIN-like preprocessing step for each layer. Following Ren et al. (2019) we use a single-layer version of this on our networks' penultimate layers because the multi-layer version in the original code does not support gradient-based attacks. ***OE:*** Outlier exposure (Hendrycks et al., 2019) enforces uniform confidence on a large out-distribution. We use their provided code to train a model with our chosen architecture. ***ACET:*** Adversarial confidence enhanced training (ACET) (Hein et al., 2019) enforces uniform confidence on a ball around points from an out-distribution by running adversarial attacks during training. In order to make the comparison with OE more meaningful we use 80M tiny images to draw the seeds rather than smoothed uniform noise as in Hein et al. (2019).

Some of the above OOD papers optimize their hyperparameters on a validation set for each out-distribution they test on. However, this leads to different classifiers for each out-distribution dataset which seems unrealistic as we want to have good generic OOD performance and not for a particular dataset. Thus we keep the comparison realistic and fair by calibrating the hyperparameters of all methods on a subset of 80M tiny images and then evaluating on the other unseen distributions.

Certified robustness against adversarial noise: We sample uniform noise images as they are obviously out-distribution for all tasks and certify using Corollary 3.1 the largest ball around the uniform noise sample on which CCU attains at most $1.1\cdot$ uniform confidence, that is 11% on all datasets. We describe how to compute the radius of this ball in Appendix D. We construct adversarial noise samples for all OOD methods by maximizing the confidence/minimizing the score via a PGD attack with 500 steps and 50 random restarts on this ball. Further details of the attack can be found in Appendix C.2. In Table 1 we show the results of running this attack on the different models. We used 200 noise images and we report clean test error on the in-distribution, the success rate (SR) (fraction of adversarial noise points for which the confidence resp. score inside the ball is higher resp. lower than the median of the in-distribution's confidence/score) and the AUC for the separation of the generated adversarial noise images and the in-distribution based on confidence/score. By construction, see Corollary 3.1, our method provably makes no overconfident predictions but we nevertheless run the attack on CCU as well. We note that only CCU performs perfectly on this task for all datasets. We also see that ACET achieves very robust performance which is maybe expected as it does some kind of adversarial training for OOD detection. Nevertheless ACET can be broken on MNIST and has no guarantees. We illustrate the generated adversarial noise images for all methods in Figure 2.

OOD performance: For each dataset and method we report the AUC for the binary classification problem of discriminating in- and out-distribution based on confidence resp. score. The results are shown in Table 2. The list of datasets we use for OOD detection can be seen in Table 2. LSUN_CR

		Base	MCD	GAN	ODIN	Maha	ACET	OE	CCU
MNIST	FMNIST	97.1	77.5	99.4	98.3	97.2	**100.0**	99.9	99.9
	EMNIST	89.5	74.1	92.8	88.7	92.4	**96.6**	95.8	92.3
	GrayCIFAR10	99.6	75.2	99.1	99.9	97.7	**100.0**	**100.0**	**100.0**
	Noise	**100.0**	77.1	99.3	**100.0**	97.7	**100.0**	**100.0**	**100.0**
	UniformNoise	91.9	77.4	99.9	96.8	99.5	**100.0**	**100.0**	**100.0**
FMNIST	MNIST	97.0	74.9	**99.7**	99.2	97.1	96.9	96.3	97.6
	EMNIST	97.7	79.9	**99.9**	99.4	97.8	97.4	99.3	99.5
	GrayCIFAR10	87.7	87.6	91.1	90.4	97.2	**100.0**	**100.0**	**100.0**
	Noise	95.9	89.7	99.4	98.4	98.8	99.8	**100.0**	**100.0**
	UniformNoise	96.6	89.4	81.2	98.3	98.4	**100.0**	97.6	**100.0**
SVHN	CIFAR10	95.3	87.6	96.8	95.9	96.5	94.7	**100.0**	**100.0**
	CIFAR100	94.9	87.3	96.1	95.5	96.0	94.7	**100.0**	**100.0**
	LSUN_CR	95.1	84.5	99.0	96.1	95.9	98.0	**100.0**	**100.0**
	Imagenet-	94.7	87.7	97.7	95.5	96.1	98.0	**100.0**	**100.0**
	Noise	96.5	88.6	96.3	87.0	97.4	95.7	**97.8**	97.3
	UniformNoise	97.9	88.5	**100.0**	98.9	97.4	**100.0**	**100.0**	**100.0**
CIFAR10	SVHN	93.3	76.6	83.9	94.6	94.8	93.0	**98.8**	98.0
	CIFAR100	86.9	74.9	82.9	86.9	68.7	86.7	**95.3**	94.1
	LSUN_CR	91.7	76.2	89.9	92.8	68.4	91.9	**98.6**	98.0
	Imagenet-	86.3	74.8	83.9	86.5	65.7	90.1	**94.7**	93.1
	Noise	94.7	74.2	81.8	96.0	81.2	94.6	**97.3**	**97.3**
	UniformNoise	90.0	77.4	73.0	92.1	56.7	**100.0**	98.8	**100.0**

Table 2: AUC (in- versus out-distribution based on confidence/score) in percent for different OOD methods and datasets (higher is better). OE and CCU have the best OOD performance.

refers to only the classroom class of LSUN and Imagenet- is a subset of 10000 resized Imagenet validation images, that have no overlap with CIFAR10/CIFAR100 classes. The noise dataset was obtained as in Hein et al. (2019) by first shuffling the pixels of the test images in the in-distribution and then smoothing them by a Gaussian filter of uniformly random width, followed by a rescaling so that the images have full range. We see that OE and CCU have the best OOD performance. MCD is worse than the base model which confirms the results found in Leibig et al. (2017) that MCD is not useful for OOD. The performance of Maha is worse than what has been reported in Lee et al. (2018c) which can have two reasons. We just use their version where one uses the scores only from the last layer and we do not calibrate hyperparameters for each test set separately but just once on the Tiny Image dataset. Especially on CIFAR10 we found that the results depend strongly on the step size. The results of ACET, GAN and ODIN are mixed but clearly outperform the baseline. Comparing Table 1 and Table 2 we see that most models perform well when evaluating on uniform noise but fail when finding the worst case in a small neighborhood around the noise point. Thus we think that such worst-case analysis should become standard in OOD evaluation.

5 Conclusion

In Hein et al. (2019) it has recently been shown that ReLU networks produce arbitrarily highly confident predictions far away from the training data, which could only be resolved by a modification of the network architecture. With CCU we present such a modification which explicitly integrates a generative model and provably show that the resulting neural network produces close to uniform predictions far away from the training data. Moreover, CCU is the only OOD method which can guarantee low confidence predictions over a whole volume rather than just pointwise and we show that all other OOD methods fail in this worst-case setting. CCU achieves this without loss in test accuracy or OOD performance. In the future it would be interesting to use more powerful generative models for which one can also guarantee their behavior far away from the training data. This is currently not the case for VAEs and GANs (Nalisnick et al., 2019; Hendrycks et al., 2019).

REFERENCES

H. H. Bauschke and J. M. Borwein. On projection algorithms for solving convex feasibility problems. *SIAM Review*, 38:367–426, 1996.

Jeremy M Cohen, Elan Rosenfeld, and J Zico Kolter. Certified adversarial robustness via randomized smoothing. arXiv:1902.02918, 2019.

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

Laurent Dinh, Jascha Sohl-Dickstein, and Samy Bengio. Density estimation using real nvp. *arXiv:1605.08803*, 2016.

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

I. J. Goodfellow, J. Shlens, and C. Szegedy. Explaining and harnessing adversarial examples. In *ICLR*, 2015.

I.J. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio. Generative adversarial nets. In *NeurIPS*, 2014.

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

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

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.

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

D. Hendrycks, M. Mazeika, and T. Dietterich. Deep anomaly detection with outlier exposure. In *ICLR*, 2019.

Diederik P Kingma and Max Welling. Auto-encoding variational bayes. In *ICLR*, 2014.

Durk P Kingma and Prafulla Dhariwal. Glow: Generative flow with invertible 1x1 convolutions. In *NeurIPS*, pp. 10215–10224, 2018.

J. Kos, I. Fischer, and D. Song. Adversarial examples for generative models. In *ICLR Workshop*, 2017.

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

B. Lakshminarayanan, A. Pritzel, and C. Blundell. Simple and scalable predictive uncertainty estimation using deep ensembles. In *NeurIPS*, 2017.

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

K. Lee, H. Lee, K. Lee, and J. Shin. Training confidence-calibrated classifiers for detecting out-of-distribution samples. In *ICLR*, 2018a.

K. Lee, H. Lee, K. Lee, and J. Shin. A simple unified framework for detecting out-of-distribution samples and adversarial attacks. In *NeurIPS*, 2018b.

Kimin Lee, Honglak Lee, Kibok Lee, and Jinwoo Shin. Training confidence-calibrated classifiers for detecting out-of-distribution samples. *arXiv:1711.09325*, 2017.

Kimin Lee, Kibok Lee, Honglak Lee, and Jinwoo Shin. A simple unified framework for detecting out-of-distribution samples and adversarial attacks. In *NeurIPS*, pp. 7167–7177, 2018c.

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.

S. Liang, Y. Li, and R. Srikant. Enhancing the reliability of out-of-distribution image detection in neural networks. In *ICLR*, 2018.

Shiyu Liang, Yixuan Li, and R Srikant. Enhancing the reliability of out-of-distribution image detection in neural networks. *arXiv:1706.02690*, 2017.

A. Madry, A. Makelov, L. Schmidt, D. Tsipras, and A. Valdu. Towards deep learning models resistant to adversarial attacks. In *ICLR*, 2018.

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

E. Nalisnick, A. Matsukawa, Y. Whye Teh, D. Gorur, and B. Lakshminarayanan. Do deep generative models know what they don't know? In *ICLR*, 2019.

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

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

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

Jie Ren, Peter J Liu, Emily Fertig, Jasper Snoek, Ryan Poplin, Mark A DePristo, Joshua V Dillon, and Balaji Lakshminarayanan. Likelihood ratios for out-of-distribution detection. In *NeurIPS*, 2019.

Danilo Jimenez Rezende, Shakir Mohamed, and Daan Wierstra. Stochastic backpropagation and approximate inference in deep generative models. 2014.

L. Schott, J. Rauber, M. Bethge, and W. Brendel. Towards the first adversarially robust neural network model on mnist. preprint, arXiv:1805.09190v3, 2018.

Vikash Sehwag, Arjun Nitin Bhagoji, Liwei Song, Chawin Sitawarin, Daniel Cullina, Mung Chiang, and Prateek Mittal. Better the devil you know: An analysis of evasion attacks using out-of-distribution adversarial examples. *preprint, arXiv:1905.01726*, 2019.

Alireza Shafaei, Mark Schmidt, and James J Little. Does your model know the digit 6 is not a cat? a less biased evaluation of" outlier" detectors. *arXiv:1809.04729*, 2018.

Antonio Torralba, Rob Fergus, and William 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.

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

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

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

A APPENDIX - PROOF OF THEOREM 3.1

Theorem A.1. *Let $(x_i, y_i)_{i=1}^n$ be the training set of the training distribution. We define the model for the conditional probability over the classes $y \in \{1, \dots, M\}$ given x as*

$$\hat{p}(y|x) = \frac{\hat{p}(y|x,i)\hat{p}(x|i) + \frac{\lambda}{M}\hat{p}(x|o)}{\hat{p}(x|i) + \lambda\hat{p}(x|o)}, \tag{14}$$

where $\lambda = \frac{\hat{p}(o)}{\hat{p}(i)} > 0$ and $M > 1$. Further, let the model for the marginal density of the in-distribution $\hat{p}(x|i)$ and out-distribution $p(x|o)$ be given by the generalized GMMs

$$\hat{p}(x|i) = \sum_{k=0}^{K_i} \alpha_k \exp\left(-\frac{d(x,\mu_k)^2}{2\sigma_k^2}\right), \qquad \hat{p}(x|o) = \sum_{l=0}^{K_o} \beta_l \exp\left(-\frac{d(x,\nu_l)^2}{2\theta_l^2}\right)$$

with $\alpha_k, \beta_l > 0$ and $\mu_k, \nu_l \in \mathbb{R}^d \quad \forall k = 1, \dots K_i,\ l = 1, \dots, K_o$ and $d : \mathbb{R}^d \times \mathbb{R}^d \to \mathbb{R}_+$ a metric. Let $z \in \mathbb{R}^d$ and define $k^* = \underset{k=1,\dots,K_i}{\arg\min} \frac{d(z,\mu_k)}{\sigma_k}$, $i^* = \underset{i=1,\dots,n}{\arg\min}\, d(z, x_i)$, $l^* = \underset{l=1,\dots,K_o}{\arg\min}\, \beta_l \exp\left(-\frac{d(z,\nu_l)^2}{2\theta_l^2}\right)$ and $\Delta = \frac{\theta_{l^*}^2}{\sigma_{k^*}^2} - 1$. For any $\epsilon > 0$, if $\min_l \theta_l > \max_k \sigma_k$ and

$$\min_{i=1,\dots,n} d(z, x_i) \geq d(x_{i^*}, \mu_{k^*}) + d(\mu_{k^*}, \nu_{l^*})\Big[\frac{2}{\Delta} + \frac{1}{\sqrt{\Delta}}\Big] + \frac{\theta_{l^*}}{\sqrt{\Delta}}\sqrt{\log\Big(\frac{\epsilon\,\lambda}{M-1}\frac{\beta_{l^*}}{\sum_k \alpha_k}\Big)}, \tag{15}$$

then it holds for all $m \in \{1, \dots, M\}$ that

$$\hat{p}(m|z) \leq \frac{1}{M}\big(1 + \epsilon\big). \tag{16}$$

In particular, if $\min_i d(z, x_i) \to \infty$, then $\hat{p}(m|z) \to \frac{1}{M}$.

Proof. The proof essentially hinges on upper bounding $\frac{\hat{p}(z|i)}{\hat{p}(z|o)}$ using the specific properties of the Gaussian mixture model. We note that

$$\hat{p}(y|x) = \frac{\hat{p}(y|x,i)\hat{p}(x|i) + \frac{\lambda}{M}\hat{p}(x|o)}{\hat{p}(x|i) + \lambda\hat{p}(x|o)} = \frac{1}{M}\frac{1 + \frac{M}{\lambda}\frac{\hat{p}(x|i)}{\hat{p}(x|o)}}{1 + \frac{1}{\lambda}\frac{\hat{p}(x|i)}{\hat{p}(x|o)}} \leq \frac{1}{M}\left(1 + \frac{M-1}{\lambda}\frac{\hat{p}(x|i)}{\hat{p}(x|o)}\right)$$

The last step holds because the function $g(\xi) = \frac{1+M\xi}{1+\xi}$ is monotonically increasing

$$\frac{\partial g}{\partial \xi} = \frac{M-1}{(1+\xi)^2} \quad \text{and} \quad \frac{\partial^2 g}{\partial \xi^2} = -2\frac{M-1}{(1+\xi)^3}. \tag{17}$$

As the second deriviative is negative for $\xi \geq 0$, g is concave for $\xi \geq 0$ and thus

$$\frac{1+M\xi}{1+\xi} = g(\xi) \leq g(0) + \frac{\partial g}{\partial \xi}\Big|_{\xi=0}(\xi - 0) = 1 + (M-1)\xi. \tag{18}$$

In order to achieve the required result we need to show that $\frac{M-1}{\lambda}\frac{\hat{p}(x|i)}{\hat{p}(x|o)} \leq \epsilon$ for x sufficiently far away from the training data.

We note that

$$\begin{aligned}\frac{\hat{p}(x|i)}{\hat{p}(x|o)} &= \frac{\sum_k \alpha_k \exp\left(-\frac{d(x,\mu_k)^2}{2\sigma_k^2}\right)}{\sum_l \beta_l \exp\left(-\frac{d(x,\nu_l)^2}{2\theta_l^2}\right)} \leq \frac{\max_k \exp\left(-\frac{d(x,\mu_k)^2}{2\sigma_k^2}\right)\sum_k \alpha_k}{\max_l \beta_l \exp\left(-\frac{d(x,\nu_l)^2}{2\theta_l^2}\right)} \\ &= \frac{\sum_k \alpha_k}{\beta_{l^*}} \exp\left(-\frac{d(x,\mu_{k^*})^2}{2\sigma_{k^*}^2} + \frac{d(x,\nu_{l^*})^2}{2\theta_{l^*}^2}\right)\end{aligned}$$

where $k^* = \underset{k}{\arg\min} \frac{d(x,\mu_k)^2}{2\sigma_k^2}$ and $l^* = \underset{l}{\arg\min}\, \beta_l \exp(-\frac{d(x,\nu_l)^2}{2\theta_l^2})$. Using triangle inequality, $d(x, \nu_{l^*}) \leq d(x, \mu_{k^*}) + d(\mu_{k^*}, \nu_{l^*})$, we get the desired condition as

$$\frac{\sum_k \alpha_k}{\beta_{l^*}} \exp\left(-d(x,\mu_{k^*})^2\left(\frac{1}{2\sigma_{k^*}^2} - \frac{1}{2\theta_{l^*}^2}\right) + \frac{d(\mu_{k^*},\nu_{l^*})d(x,\mu_{k^*})}{\theta_{l^*}^2} + \frac{d(\mu_{k^*},\nu_{l^*})^2}{2\theta_{l^*}^2}\right) \leq \frac{\epsilon\,\lambda}{M-1}$$

Thus we get with $a = \left(\frac{1}{2\sigma_{k^*}^2} - \frac{1}{2\theta_{l^*}^2}\right)$, $b = \frac{d(\mu_{k^*},\nu_{l^*})}{\theta_{l^*}^2}$ and $c = \frac{d(\mu_{k^*},\nu_{l^*})^2}{2\theta_{l^*}^2}$, $d = \log\left(\frac{\epsilon\,\lambda}{M-1}\frac{\beta_{l^*}}{\sum_k \alpha_k}\right)$, the quadratic inequality

$$-d(x,\mu_{k^*})^2 a + d(x,\mu_{k^*})b + \frac{b^2}{2} \leq d,$$

where $d < 0$ for sufficiently small ϵ. We get the solution

$$d(x,\mu_{k^*}) \geq \frac{b}{2a} + \sqrt{\max\left\{0, \frac{c-d}{a} + \frac{b^2}{4a^2}\right\}}.$$

It holds, using $\sqrt{a+b} \leq \sqrt{a} + \sqrt{b}$ for $a, b > 0$,

$$\frac{b}{2a} + \sqrt{\max\left\{0, \frac{c-d}{a} + \frac{b^2}{4a^2}\right\}} \leq \frac{b}{a} + \sqrt{\frac{c}{a}} + \sqrt{\frac{d}{a}}.$$

One can simplify

$$\frac{b}{a} = 2\frac{\sigma_{k^*}^2\theta_{l^*}^2}{\theta_{l^*}^2 - \sigma_{k^*}^2}\frac{d(\mu_{k^*},\nu_{l^*})}{\theta_{l^*}^2} = 2\frac{\sigma_{k^*}^2\, d(\mu_{k^*},\nu_{l^*})}{\theta_{l^*}^2 - \sigma_{k^*}^2} = 2\frac{d(\mu_{k^*},\nu_{l^*})}{\frac{\theta_{l^*}^2}{\sigma_{k^*}^2} - 1}$$

$$\frac{c}{a} = 2\frac{\sigma_{k^*}^2\theta_{l^*}^2}{\theta_{l^*}^2 - \sigma_{k^*}^2}\frac{d(\mu_{k^*},\nu_{l^*})^2}{2\theta_{l^*}^2} = \frac{\sigma_{k^*}^2\, d(\mu_{k^*},\nu_{l^*})^2}{\theta_{l^*}^2 - \sigma_{k^*}^2} = \frac{d(\mu_{k^*},\nu_{l^*})^2}{\frac{\theta_{l^*}^2}{\sigma_{k^*}^2} - 1}$$

Noting that $d(x,\mu_{k^*}) \geq |d(x,x_{i^*}) - d(x_{i^*},\mu_{k^*})|$ we get that

$$d(x,x_{i^*}) \geq d(x_{i^*},\mu_{k^*}) + \frac{b}{2a} + \frac{b}{a} + \sqrt{\frac{c}{a}} + \sqrt{\frac{d}{a}},$$

implies $\frac{M-1}{\lambda}\frac{\hat{p}(x|i)}{\hat{p}(x|o)} \leq \epsilon$. The last statement follows directly by noting that by assumption $a > 0$ (independently of the choice of l^* and k^*) and $b, c, d(x_{i^*},\mu_{k^*})$ are bounded as K_i, K_o, n are finite. With $\Delta = \frac{\theta_{l^*}^2}{\sigma_{k^*}^2} - 1$ we can rewrite the required condition as

$$d(x,x_{i^*}) \geq d(x_{i^*},\mu_{k^*}) + d(\mu_{k^*},\nu_{l^*})\Big[\frac{2}{\Delta} + \frac{1}{\sqrt{\Delta}}\Big] + \frac{\theta_{l^*}}{\sqrt{\Delta}}\sqrt{\log\Big(\frac{\epsilon\,\lambda}{M-1}\frac{\beta_{l^*}}{\sum_k \alpha_k}\Big)}.$$

□

B Appendix - Proof of Corollary 3.1

Corollary B.1. *Let $x_0 \in \mathbb{R}^d$ and $R > 0$, then with $\lambda = \frac{\hat{p}(o)}{\hat{p}(i)}$ it holds*

$$\max_{d(x,x_0)\leq R} \hat{p}(y|x) \;\leq\; \frac{1}{M}\frac{1 + M\frac{b}{\lambda}}{1 + \frac{b}{\lambda}}, \tag{19}$$

where $b = \dfrac{\sum_{k=1}^{K_i} \alpha_k \exp\Big(-\frac{\max\{d(x_0,\mu_k)-R,0\}^2}{2\sigma_k^2}\Big)}{\sum_{l=1}^{K_o} \beta_l \exp\Big(-\frac{(d(x_0,\nu_l)+R)^2}{2\theta_l^2}\Big)}$.

Proof. From the previous section we already know that $\hat{p}(y|x) \leq \frac{1}{M}\frac{1+M\frac{b}{\lambda}}{1+\frac{b}{\lambda}}$ as long as $\frac{p(x|i)}{p(x|o)} \leq b$. Now we can separately bound the numerator and denominator within a ball of radius R around x_0. For the numerator we have

$$\max_{d(x,x_0)\leq R} \hat{p}(x|i) \leq \sum_{k=1}^{K} \alpha_k \max_{d(x,x_0)\leq R} e^{-\frac{d(x,\mu_k)^2}{2\sigma_k^2}} \tag{20}$$

$$\leq \sum_{k=1}^{K} \alpha_k \exp\left(-\frac{\min\limits_{d(x,x_0)\leq R} d(x,\mu_k)^2}{2\sigma_k^2}\right)$$

$$\leq \sum_{k=1}^{K} \alpha_k \exp\left(-\frac{\left(\max\left\{d(\mu_k,x_0) - R, 0\right\}\right)^2}{2\sigma_k^2}\right), \tag{21}$$

where we have lower bounded $\min_{d(x,x_0)\leq R} d(x,\mu_k)$ via the reverse triangle inequality

$$\begin{aligned}\min_{d(x,x_0)\leq R} d(x,\mu_k) &\geq \min_{d(x,x_0)\leq R} |d(x_0,\mu_k) - d(x,x_0)|, \\ &\geq \max\left\{\min_{d(x,x_0)\leq R}(d(x_0,\mu_k) - d(x_0,\mu_k)), 0\right\}, \\ &\geq \max\left\{d(x_0,\mu_k) - r, 0\right\}. \end{aligned} \tag{22}$$

The denominator can similarly be bounded via

$$\min_{d(x,x_0)\leq R} \hat{p}(x|o) \geq \sum_{l=1}^{K_o} \beta_l \min_{d(x,x_0)\leq R} e^{-\frac{d(x,\nu_l)^2}{2\theta_k^2}} \tag{23}$$

$$\geq \sum_{l=1}^{K_o} \beta_l \exp\left(-\frac{\max_{d(x,x_0)\leq R} d(x,\nu_l)^2}{2\theta_l^2}\right)$$

$$\geq \sum_{l=1}^{K_o} \beta_l \exp\left(-\frac{(d(x_0,\nu_l)+R)^2}{2\theta_l^2}\right). \tag{24}$$

With both of these bounds in place the conclusion immediately follows. □

C Appendix - Experimental Details

Unless specified otherwise we use ADAM on MNIST with a learning rate of $1e-3$ and SGD with learning rate 0.1 for the other datasets. The learning rate for the GMM is always set to $1e-5$. We decrease all learning rates by a factor of 10 after 50, 75 and 90 epochs. Our batch size is 128, the total number of epochs 100 and weight decay is set to $5e-4$.

When training ACET, OE and CCU with 80 million tiny images we pick equal batches of in- and out-distribution data (corresponding to $p(i) = p(o)$) and concatenate them into a batches of size 256. Note that during the 100 epochs only a fraction of the 80 million tiny images are seen and so there is no risk of over-fitting.

C.1 Data Augmentation

Our data augmentation scheme uses random crops with a padding of 2 pixels on MNIST and FMNIST. On SVHN and CIFAR10 the padding width is 4 pixels. For SVHN we fill the padding with the value at the boundary and for CIFAR10 we apply reflection at the boundary pixels. On top of this we include random horizontal flips on CIFAR10. For MNIST and FMNIST we generate 60000 such samples and for SVHN and CIFAR10 50000 samples by drawing from the clean dataset without replacement. This augmented data is used to calculate the covariance matrix from equation 13. During the actual training we use the same data augmentation scheme in a standard fashion.

C.2 Attack details

We begin with a step size of 3 and for each of the 50 restarts we randomly initialize at some point in the ellipsoid. Whenever a gradient step successfully decreases the losses we increase the step size by a factor of 1.1. Whenever the loss increases instead we use backtracking and decrease the step size by a factor of 2. We apply normal PGD using the l_2-norm in the transformed space to ensure that we stay on the ellipsoid and after each gradient step we rotate back into the original space to project onto the box $[0,1]^d$. The result is not guaranteed be on the ellipsoid so after the 500 steps we use the alternating projection algorithm (Bauschke & Borwein, 1996) for 10 steps which is guaranteed to converge to a point in the intersection of the ellipsoid and the box because both of these sets are convex.

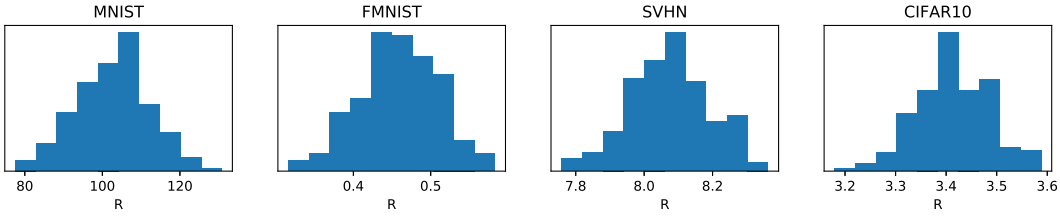

Figure 3: **Histograms of bounds:** Certified radius in transformed space for different datasets.

D APPENDIX - FINDING A THE CERTIFIABLE RADIUS

Since Corollary 3.1 does not explicitly give a radius, one has to numerically invert the bound. The bound

$$b(R) = \frac{\sum_{k=1}^{K_i} \alpha_k \exp\left(-\frac{\max\{d(x_0,\mu_k)-R,0\}^2}{2\sigma_k^2}\right)}{\sum_{l=1}^{K_o} \beta_l \exp\left(-\frac{(d(x_0,\nu_l)+R)^2}{2\theta_l^2}\right)} \tag{25}$$

is monotonically increasing in R. Thus, for a given sample x_0 one can fix a desired bound $\max_{d(x,x_0)\leq R} \hat{p}(x|i) \leq \frac{1}{M}\nu$, where $\nu \in (1, M)$ and then find the unique solution

$$b(R) = \frac{\nu - 1}{M - \nu}\lambda \tag{26}$$

for R via bisection. This radius $\hat{R}$ will then represent the maximal radius, that one can certify using Corollary 3.1. The presumption is, of course, that for $R = 0$ one has a sufficiently low bound in the first place, i.e. that a solution exists. In our experiments on uniform noise we did not encounter a single counterexample to this assumption. We show the radii for the different datasets in Figure 3.