Provably Adversarially Robust Detection of Out-of-Distribution Data (Almost) for Free

Alexander Meinke*
University of Tübingen
Tübingen AI Center

Julian Bitterwolf
University of Tübingen
Tübingen AI Center

Matthias Hein
University of Tübingen
Tübingen AI Center

Abstract

The application of machine learning in safety-critical systems requires a reliable assessment of uncertainty. However, deep neural networks are known to produce highly overconfident predictions on out-of-distribution (OOD) data. Even if trained to be non-confident on OOD data, one can still adversarially manipulate OOD data so that the classifier again assigns high confidence to the manipulated samples. We show that two previously published defenses can be broken by better adapted attacks, highlighting the importance of robustness guarantees around OOD data. Since the existing method for this task is hard to train and significantly limits accuracy, we construct a classifier that can simultaneously achieve provably adversarially robust OOD detection and high clean accuracy. Moreover, by slightly modifying the classifier's architecture our method provably avoids the asymptotic overconfidence problem of standard neural networks. We provide code for all our experiments.†

1 Introduction

Deep neural networks have achieved state-of-the-art performance in many application domains. However, the widespread usage of deep neural networks in safety-critical applications, e.g. in healthcare, autonomous driving/aviation, manufacturing, raises concerns as deep neural networks have problematic deficiencies. Among these deficiencies, overconfident predictions on non-task related inputs [44, 21] have recently attracted a lot of interest. Even theoretically derived weaknesses like ReLU networks provably being overconfident far away from the training data [20] are yet to be fixed. Meanwhile, reliable confidences of the classifier on the classification task (in-distribution) [19] as well as on the out-distribution [21, 20] are important to be able to detect when the deep neural network is working outside of its specification, which can then be used to either involve a human operator or to fall back into a "safe state". Thus, solving this problem is of high importance for trustworthy ML systems. Crucially, a detection method needs to generalize to novel test out-distributions that are not available during training, since one does not know which unknown inputs can be expected.

Many approaches have been proposed for OOD detection, [21, 33, 31, 32, 22, 48, 20, 41, 9, 46, 35, 36]. In this paper we focus on confidence based OOD detection, i.e. the probability of the predicted class is used to decide whether to reject or accept the sample, because of its straightforward interpretation and because it has been shown to perform no worse than other scores for OOD detection [7]. One of the currently best performing methods enforces low confidence during training ("outlier exposure" (OE)) on a large and diverse set of out-distribution images [22] which leads to strong separation of in- and out-distribution based on the confidence of the classifier.

*Corresponding author - alexander.meinke@uni-tuebingen.de
†https://github.com/AlexMeinke/Provable-OOD-Detection

36th Conference on Neural Information Processing Systems (NeurIPS 2022).

Table 1: **ProoD combines desirable properties of existing (adversarially robust) OOD detection methods.** It has high test accuracy and standard OOD detection performance (as [22]) and has worst-case guarantees if the out-distribution samples are adversarially perturbed in an l_∞-neighborhood to maximize the confidence (see Section 4.2). Similar to CCU [41] it avoids the problem of asymptotic overconfidence far away from the training data.

	OE [22]	CCU [41]	ACET/ATOM [20, 9]	GOOD [8]	ProoD
High accuracy	✓	✓	✓		✓
High clean OOD detection performance	✓	✓	✓		✓
Adv. OOD l_∞-robustness			(✓)	✓	✓
Adv. OOD l_∞-certificates				✓	✓
Provably not asympt. overconfident		✓			✓

A remaining robustness problem of standard OOD detection methods is that they are vulnerable to adversarial perturbations, i.e. small modifications of OOD inputs can lead to large confidence of the classifier on the manipulated samples [44, 20, 50]. Of course, an OOD input, which by definition is semantically far away from the in-distribution, should not be able to be moved into a region that is considered in-distribution by the detection model if the movement is imperceptibly small. On the other hand, a slightly perturbed in-distribution input can still be considered in-distribution for some perturbations (e.g. if the perturbation resembles standard camera noise), but for other perturbations it might be highly atypical and therefore should arguably rather be seen as OOD. Furthermore, adversarial robustness on the in-distribution is known to come at the cost of clean accuracy [53] which hinders the adoption of such methods in practice. We aim to provide a method that does not harm the in-distribution performance in any way and thus, like previous OOD-detection methods, we do *not* consider adversarially manipulated in-distribution samples and focus on ensuring that OOD samples remain OOD under adversarial attacks.

While different methods for adversarially robust OOD detection have been proposed [20, 50, 41, 9, 8] there is little work on *provably* adversarially robust OOD detection [41, 8, 25, 5]. For the standard empirical evaluation of adversarial robustness, for each input one runs an array of different attacks that conform to the assumed threat model and records the output for the worst found perturbation. This means that there is no guarantee that a more malign perturbation does not exist, as only a lower bound on the adversarial robustness is established. Provable adversarial robustness on OOD data, which we provide in this paper, yields a mathematically deduced upper bound on the worst-case confidence around each OOD sample. For our guaranteed upper bounds on the confidence of an OOD sample, it is certain that no applicable manipulation raises the confidence above the certified value.

In [25] they apply randomized smoothing to obtain guarantees wrt. l_2-perturbations for Dirchlet-based models [39, 40, 51] which already show quite some gap in terms of AUC-ROC to SOTA OOD detection methods even without attacks. Interval bound propagation (IBP) [17, 42, 56, 23] has been shown to be one of the most effective techniques in certified adversarial robustness on the in-distribution when applied during training. In GOOD [8] they use IBP to compute upper bounds on the confidence in an l_∞-neighborhood of the input and minimize these upper bounds on a training out-distribution. This yields classifiers with pointwise guarantees for adversarially robust OOD detection even for "close" out-distribution inputs which generalize to novel OOD test distributions. However, the employed architectures of the neural network are restricted to rather shallow networks as otherwise the bounds of IBP are loose. Thus, they obtain low classification accuracy which is far from the state-of-the-art, e.g. 91% on CIFAR10, and their approach does not scale to more complex tasks like ImageNet. In particular, despite its low accuracy the employed network architecture is quite large and has higher memory consumption than a ResNet50. The authors of [5] use SOTA verification techniques [15] and get guarantees for OOD detection wrt. l_∞-perturbations for ACET models [20] that were not specifically trained to be verifiable but the guarantees obtained by training the models via IBP in [8] are significantly better.

A *different* type of guaranteed low confidence on OOD data pertains to the asymptotic behavior of a classifier. Since standard ReLU networks provably have increasing confidence in almost all directions far away from the training data [20], one has to modify the architecture in order to solve this problem. In CCU [41] the authors append density estimators based on Gaussian mixture models for in- and out-distribution to the softmax layer. By also enforcing low confidence on a training out-distribution, they achieve similar OOD detection performance to [22] but can guarantee that the classifier shows

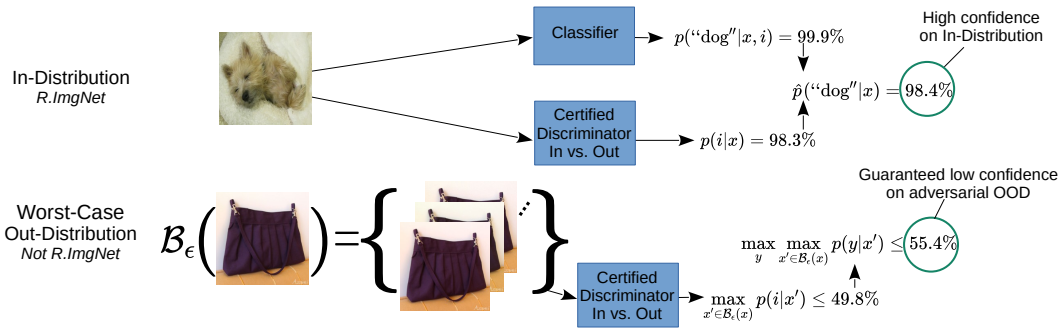

Figure 1: **ProoD's Architecture:** Combining the output of a classifier and a certified discriminator, see Eq. (2), we achieve high confidence on the in-distribution sample of a dog (R.ImgNet).The certified discriminator, see Eq. (4), yields an upper bound on the confidence in a ℓ_∞-neighborhood of the shown OOD sample not belonging to any classes of R.ImgNet. ProoD achieves provable guarantees on adversarial OOD detection without loss in accuracy or clean OOD detection.

decreasing confidence as one moves away from the training data. However, for close in-distribution inputs this approach yields no guarantee as the Gaussian mixture models are not powerful enough for complex image classification tasks. In [27, 28] similar asymptotic guarantees are derived for Bayesian neural networks but without any robustness guarantees.

In this paper we propose ProoD which merges a certified binary discriminator for in-versus out-distribution with a classifier for the in-distribution task in a principled fashion into a joint classifier. This combines the advantages of CCU [41] and GOOD [8] without suffering from their downsides. In particular, ProoD simultaneously achieves the following:

- Guaranteed adversarially robust OOD detection via certified upper bounds on the confidence in l_∞-balls around OOD samples.
- Additionally, it provably prevents the asymptotic overconfidence of deep neural networks.
- It can be used with arbitrary architectures and has no loss in prediction performance and standard OOD detection performance.

Thus, we get provable guarantees for adversarially robust OOD detection, fix the asymptotic overconfidence (almost) for free as we have (almost) no loss in prediction and standard OOD detection performance. We qualitatively compare the properties of our model to prior approaches in Table 1.

2 Provably Robust Detection of Out-of-distribution Data

In the following we consider feedforward networks for classification, $f : \mathbb{R}^d \to \mathbb{R}^K$, with K classes defined with input $x^{(0)} = x$ and layers $l = 1, \dots L-1$ as

$$x^{(l)} = \sigma^{(l)}\left(W^{(l)}x^{(l-1)} + b^{(l)}\right), \qquad f(x) = W^{(L)}x^{(L-1)} + b^{(L)}, \tag{1}$$

where $W^{(l)}$ and $b^{(l)}$ are weights and biases and $\sigma^{(l)}$ is either the ReLU or leaky ReLU activation function of layer l . We refer to the output of f as the *logits* and get a probability distribution over the classes via $\hat{p}(y|x) = \frac{e^{f_y(x)}}{\sum_k^K e^{f_k(x)}}$ for $y = 1, \dots, K$. We define the confidence as $\mathrm{Conf}(f(x)) = \max_{y=1,\dots,K} \hat{p}(y|x)$.

2.1 Joint Model for OOD Detection and Classification

In our joint model we assume that there exists an in- and out-distribution where the out-distribution samples are unrelated to the in-distribution task. Thus, we can formally write the conditional distribution on the input as

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

where $\hat{p}(i|x)$ is the conditional distribution that sample x belongs to the in-distribution and $\hat{p}(y|x,i)$ is the conditional distribution for the in-distribution. We assume that OOD samples are unrelated and thus maximally un-informative to the in-distribution task, i.e. we fix $\hat{p}(y|x,o) = \frac{1}{K}$, so that the classifier can be written as

$$\hat{p}(y|x) = \hat{p}(y|x,i)\hat{p}(i|x) + \frac{1}{K}(1 - \hat{p}(i|x)). \tag{3}$$

We train the binary classifier $\hat{p}(i|x)$ in a certified robust fashion wrt. an l_∞-threat model so that even adversarially manipulated OOD samples are detected. In order to avoid confusion with the multi-class classifier, we will refer to $\hat{p}(i|x)$ as a binary discriminator. In an l_∞-ball of radius ϵ around $x \in \mathbb{R}^d$ and for all y we get the upper bound on the confidence of the final classifier in Eq. (3):

$$\max_{\|x'-x\|_\infty \leq \epsilon} \hat{p}(y|x') \leq \max_{\|x'-x\|_\infty \leq \epsilon} \hat{p}(i|x') + \frac{1}{K}\big(1 - \hat{p}(i|x')\big) = \frac{K-1}{K} \max_{\|x'-x\|_\infty \leq \epsilon} \hat{p}(i|x') + \frac{1}{K}, \tag{4}$$

where we have used that $p(y|x,i) \leq 1 \forall x, y$, so we can defer the certification "work" to the binary discriminator. Using a particular constraint on the weights of the binary discriminator, we get similar asymptotic properties as in [41] but additionally get certified adversarial robustness for close out-distribution samples as in [8]. In contrast to [8], this comes without loss in test accuracy or non-adversarial OOD detection performance since in our model the neural network used for the in-distribution classification task $\hat{p}(y|x,i)$ is independent of the binary discriminator. Thus, we have the advantage that the classifier can use arbitrary deep neural networks and is not constrained to certifiable networks. We call our approach **Pr**ovable **o**ut-**o**f-**D**istribution detector (ProoD) and visualize its components in Figure 1. The intuitive idea of why ProoD can achieve adversarially robust OOD detection without loss in clean OOD detection can be explained with the behavior of the predicted probability distribution provided in Equation (3).

- **For clean OOD:** the classifier $\hat{p}(y|x,i)$ (trained similar to Outlier Exposure) already enforces low confidence on out-of-distribution points and thus irrespective of the values $\hat{p}(i|x)$, the resulting output of $\hat{p}(y|x)$ will be close to uniform as well and thus ProoD performs similar to Outlier Exposure.
- **For adversarial OOD:** the classifier confidence $\max_y \hat{p}(y|x,i)$ is potentially corrupted but now the binary discriminator $\hat{p}(i|x)$ kicks in and ensures that the resulting prediction $\hat{p}(y|x)$ is close to uniform.

This explains why the combination of certified discriminator and classifier works much better than the individual parts and the use of this "redundancy" is the key idea of ProoD.

Certifiably Robust Binary Discrimination of In- versus Out-Distribution The first goal is to get a certifiably adversarially robust OOD detector $\hat{p}(i|x)$. We train this binary discriminator independently of the overall classifier as the training schedules for certified robustness are incompatible with the standard training schedules of normal classifiers. For this binary classification problem we use a logistic model $\hat{p}(i|x) = \frac{1}{1+e^{-g(x)}}$, where $g : \mathbb{R}^d \to \mathbb{R}$ are logits of a neural network (we denote the weights and biases of g by W_g and b_g in order to differentiate it from the classifier f introduced in the next paragraph). Let $(x_r, y_r)_{r=1}^N$ be our in-distribution training data (we use the class encoding $+1$ for the in-distribution and -1 for the out-distribution) and $(z_s)_{s=1}^M$ be our training out-distribution data. Then the optimization problem associated to the binary classification problem becomes:

$$\min_{\substack{g \\ W_g^{(L_g)} < 0}} \frac{1}{N} \sum_{r=1}^{N} \log\Big(1 + e^{-g(x_r)}\Big) + \frac{1}{M} \sum_{s=1}^{M} \log\Big(1 + e^{\bar{g}(z_s)}\Big), \tag{5}$$

where we minimize over the parameters of the neural network g under the constraint that the weights of the output layer $W_g^{(L_g)}$ are componentwise negative and $\bar{g}(z) \geq \max_{u \in B_p(z,\epsilon)} g(u)$ is an upper bound on the output of g around OOD samples for a given l_p-threat model $B_p(z,\epsilon) = \{u \in [0,1]^d \mid \|u - z\|_p \leq \epsilon\}$. In this paper we always use an l_∞-threat model. This upper bound could, in principle, be computed using any certification technique but we will use interval bound propagation (IBP) since it is simple, fast and has been shown to produce SOTA results [17]. Note that this is not standard adversarial training for a binary classification problem as here we have

an asymmetric situation: we want to be (certifiably) robust to adversarial manipulation on the out-distribution data but *not* on the in-distribution and thus the upper bound is only used for out-distribution samples. The negativity of the output layer's weights $W_g^{(L_g)}$ is enforced by using the parameterization $(W_g^{(L_g)})_j = -e^{h_j}$ componentwise and optimizing over h_j. In Section 3 we show how the negativity of $W_g^{(L_g)}$ allows us to control the asymptotic behavior of the joint classifier.

For the reader's convenience we quickly present the upper $\overline{x}^{(l)}$ and lower $\underline{x}^{(l)}$ bounds on the output of layer l in a feedforward neural network produced by IBP:

$$\overline{x}^{(l)} = \sigma\left(W_+^{(l)}\overline{x}^{(l-1)} + W_-^{(l)}\underline{x}^{(l-1)} + b^{(l)}\right), \quad \underline{x}^{(l)} = \sigma\left(W_+^{(l)}\underline{x}^{(l-1)} + W_-^{(l)}\overline{x}^{(l-1)} + b^{(l)}\right), \tag{6}$$

where $W_+ = \max(0, W)$ and $W_- = \min(0, W)$ (min/max used componentwise). For an l_∞-threat model one starts with the upper and lower bounds for the input layer $\overline{x}^{(0)} = x + \epsilon$ and $\underline{x}^{(0)} = x - \epsilon$ and then iteratively computes the layerwise upper and lower bounds $\overline{x}^{(l)}$, $\underline{x}^{(l)}$ which fulfill

$$\underline{x}^{(l)} \;\leq \min_{\|x'-x\|_\infty \leq \epsilon} x^{(l)}(x') \leq \max_{\|x'-x\|_\infty \leq \epsilon} x^{(l)}(x') \;\leq\; \overline{x}^{(l)}. \tag{7}$$

While in [8] they also used IBP to upper bound the confidence of the classifier this resulted in a bound that took into account all $\mathcal{O}(K^2)$ logit differences between all classes. In contrast, our loss in Eq. (5) is significantly simpler as we just have a binary classification problem and therefore only need a single bound. Thus, our approach easily scales to tasks with a large number of classes and training the binary discriminator with IBP turns out to be significantly more stable than the approach in [8].

(Semi)-Joint Training of the Final Classifier Given the certifiably robust model $\hat{p}(i|x)$ for the binary classification task between in- and out-distribution, we need to determine the final predictive distribution $\hat{p}(y|x)$ in Eq. (2). On top of the provable OOD performance that we get from Eq. (4), we also want to achieve SOTA performance on unperturbed OOD data. In principle we could independently train a model for the predictive in-distribution task $\hat{p}(y|x, i)$, e.g. using outlier exposure (OE) [22] or any other state-of-the-art OOD detection method and simply combine it with our $\hat{p}(i|x)$. While this does lead to models with high OOD performance that also have guarantees, it completely ignores the interaction between $\hat{p}(i|x)$ and $\hat{p}(y|x, i)$ during training. Instead we propose to train $\hat{p}(y|x, i)$ by optimizing our final predictive distribution $\hat{p}(y|x)$. Note that in order to retain the guarantees of $\hat{p}(i|x)$ we only train the parameters of the neural network $f : \mathbb{R}^d \to \mathbb{R}^K$ and need to keep $\hat{p}(i|x)$ resp. g fixed. Because g stays fixed we call this semi-joint training. We use OE [22] for training $\hat{p}(y|x)$ with the cross-entropy loss and use the softmax-function in order to obtain the predictive distribution $\hat{p}_f(y|x, i) = \frac{e^{f_y(x)}}{\sum_k e^{f_k(x)}}$ from f:

$$\begin{aligned}
&\min_f -\frac{1}{N}\sum_{r=1}^{N} \log\left(\hat{p}(y_r|x_r)\right) - \frac{1}{M}\sum_{s=1}^{M}\frac{1}{K}\sum_{l=1}^{K}\log\left(\hat{p}(l|z_s)\right) \\
=&\min_f -\frac{1}{N}\sum_{r=1}^{N}\log\Big(\hat{p}_f(y_r|x_r, i)\hat{p}(i|x_r) + \frac{1}{K}\big(1 - \hat{p}(i|x_r)\big)\Big) \\
&- \frac{1}{M}\sum_{s=1}^{M}\frac{1}{K}\sum_{l=1}^{K}\log\Big(\hat{p}_f(l|z_s, i)\hat{p}(i|z_s) + \frac{1}{K}\big(1 - \hat{p}(i|z_s)\big)\Big),
\end{aligned} \tag{8}$$

where the first term is the standard cross-entropy loss on the in-distribution but now for our joint model for $\hat{p}(y|x)$ and the second term enforces uniform confidence on out-distribution samples. In App. B we show that semi-joint training leads to stronger guarantees than separate training.

The loss in Eq. (5) implicitly weighs the in-distribution and worst-case out-distribution equally, which amounts to the assumption $p(i) = \frac{1}{2} = p(o)$. This highly conservative choice simplifies training the binary discriminator but may not reflect the expected frequency of OOD samples at test time and in effect means that $\hat{p}(i|x)$ tends to be quite low. This typically yields good guaranteed AUCs but can have a negative impact on the standard out-distribution performance. In order to better explore the trade-off of guaranteed and standard OOD detection, we repeat the above semi-joint training with different shifts of the offset parameter in the output layer

$$b' = b_g^{(L_g)} + \Delta, \tag{9}$$

where $\Delta \geq 0$ leads to increasing $\hat{p}(i|x)$. This shift has a direct interpretation in terms of the probabilities $p(i)$ and $p(o)$. Under the assumption that our binary discriminator g is perfect, that is

$$p(i|x) = \frac{p(x|i)p(i)}{p(x|i)p(i) + p(x|o)p(o)} = \frac{1}{1 + e^{-g(x)}}, \tag{10}$$

then it holds that $e^{g(x)} = \frac{p(x|i)p(i)}{p(x|o)p(o)}$. A change of the prior probabilities $\tilde{p}(i)$ and $\tilde{p}(o)$ without changing $p(x|i)$ and $p(x|o)$ then corresponds to a novel classifier

$$e^{\tilde{g}(x)} = \frac{p(x|i)\tilde{p}(i)}{p(x|o)\tilde{p}(o)} = \frac{p(x|i)p(i)}{p(x|o)p(o)} \frac{p(o)\tilde{p}(i)}{p(i)\tilde{p}(o)} = e^{g(x)} e^{\Delta} \tag{11}$$

with $\Delta = \log\left(\frac{p(o)\tilde{p}(i)}{p(i)\tilde{p}(o)}\right)$. Note that $\tilde{p}(i) > p(i)$ corresponds to positive shifts. In practice, this parameter can be chosen based on the priors for a particular application. Since no such priors are available in our case we determine a suitable shift by evaluating on the training out-distribution (see Section 4.2). Note that we explicitly do not train the shift parameter since this way the guarantees would get lost as the classifier implicitly learns a large Δ in order to maximize the confidence on the in-distribution, thus converging to a normal outlier exposure-type classifier without any guarantees.

3 Guarantees on Asymptotic Confidence

In this section we show that our specific construction provably avoids the issue of asymptotic overconfidence that was pointed out in [20]. Note that the resulting guarantee (as stated in Theorem 1) is different from and in addition to the robustness guarantees discussed in the previous section (see Eq. (4)). The previous section dealt with providing confidence upper bounds on neighborhoods around OOD samples whereas this section deals with ensuring that a classifier's confidence decreases asymptotically as one moves away from all training data.

We note that a ReLU neural network $f : \mathbb{R}^d \to \mathbb{R}^K$ as defined in Eq. (1) using ReLU or leaky ReLU as activation functions, potential max-or average pooling and skip connection yields a piece-wise affine function [2], i.e. there exists a finite set of polytopes $Q_r \subset \mathbb{R}^d$ with $r = 1, \ldots, R$ such that $\cup_{r=1}^{R} Q_r = \mathbb{R}^d$ and f restricted to each of the polytopes is an affine function. Since there are only finitely many polytopes some of them have to extend to infinity and on these ones the neural network is essentially an affine classifier. This fact has been used in [20] to show that ReLU networks are almost always asymptotically overconfident in the sense that if one moves to infinity the confidence of the classifier approaches 1 (instead of converging to the desirable $1/K$ as in these regions the classifier has never seen any data). The following theorem now shows that, in contrast to standard ReLU networks, our proposed joint classifier gets provably less confident in its decisions as one moves away from the training data which is a desirable property of any reasonable classifier.

Theorem 1. *Let $x \in \mathbb{R}^d$ with $x \neq 0$ and let $g : \mathbb{R}^d \to \mathbb{R}$ be the ReLU-network of the binary discriminator (with the last activation being a non-leaky ReLU). Denote by $\{Q_r\}_{r=1}^{R}$ the finite set of polytopes on which g is affine (exists by Lemma 1 in App. C). Denote by Q_t the polytope such that $\beta x \in Q_t$ for all $\beta \geq \alpha$ and let $x^{(L-1)}(z) = Uz + d$ with $U \in \mathbb{R}^{n_{L-1} \times d}$ and $d \in \mathbb{R}^{n_{L-1}}$ be the output of the pre-logit layer of g for $z \in Q_t$. If $Ux \neq 0$, then $\lim_{\beta \to \infty} \hat{p}(y|\beta x) = \frac{1}{K}$.*

The proof is in App. C. In App. A we show that the condition $Ux \neq 0$ is not restrictive, as this property holds in all cases where we checked it. The negativity condition on the weights $W_g^{(L_g)}$ of the output layer of the in-vs. out-distribution discriminator g is crucial for the proof. This may seem restrictive, but we did not encounter any negative influence of this constraint on test accuracy, guaranteed or standard OOD detection performance. Thus, the asymptotic guarantees come essentially for free.

4 Experiments

4.1 Training of ProoD

We provide experiments on CIFAR10, CIFAR100 [29] and Restricted Imagenet (R.ImgNet) [53]. The latter consists of ImageNet images (ILSVRC2012) [16, 49] belonging to 9 types of animals.

Training the Binary Discriminator We train the binary discriminator between in-and out-distribution using the loss in Eq. (5) with the bounds over an l_∞-ball of radius $\epsilon = 0.01$ for the out-distribution following [8]. We use relatively shallow CNNs with only 5 layers plus pooling layers, see App. D. For the training out-distribution, we could follow previous work and use 80M Tiny Images [52] for CIFAR10/100. However, there have been concerns over the use of this dataset [6] due to offensive class labels. Although we do not use any of the class labels, we choose to use OpenImages [30] as training OOD instead. In order to ensure a fair comparison with prior work we also present results that were obtained using 80M Tiny Images in App. E. For R.ImgNet we use the ILSVRC2012 train images that do not belong to R.ImgNet as training out-distribution (NotR.ImgNet).

Semi-Joint Training For the classifier we use a ResNet18 architecture on CIFAR and a ResNet50 on R.ImgNet. Note that the architecture of our binary discriminator is over an order of magnitude smaller than the one in [8] (11MB instead of 135MB) and thus the memory overhead for the binary discriminator is less than a third of that of the classifier. All schedules, hardware and hyperparameters are described in App. D. As discussed in Section 2.1, when training the binary discriminator one implicitly assumes that in- and (worst-case) out-distribution samples are equally likely. It seems very unlikely that one would be presented with such a large number of OOD samples in practice but as discussed in Section 2.1, we can adjust the weight of the losses after training the discriminator (but before training the classifier) by shifting the bias $b_g^{(L_g)}$ in the output layer of the binary discriminator. We train several ProoD models for binary shifts in $\{0, 1, 2, 3, 4, 5, 6\}$ and then evaluate the AUC and guaranteed AUC (see 4.2) on a subset of the training out-distribution OpenImages (resp. NotR.ImgNet). For all bias shifts we use the same fixed provably trained binary discriminator and only train the classifier part. As our goal is to have provable guarantees with minimal or no loss on the standard OOD detection task, among all solutions which have better AUC than outlier exposure (OE) [22] we choose the one with the highest guaranteed AUC on OpenImages (on CIFAR10/CIFAR100) resp. NotR.ImgNet (on R.ImgNet). If none of the solutions has better AUC than OE on the training out-distribution we take the one with the highest AUC. We show the trade-off curves for the example in App. D.

4.2 Evaluation

Setup For OOD evaluation for CIFAR10/100 we use the test sets from CIFAR100/10, SVHN [43], the classroom category of downscaled LSUN [55] (LSUN_CR) as well as smooth noise as suggested in [20] and described in App. D. For R.ImgNet we use Flowers [45], FGVC Aircraft [38], Stanford Cars [26] and smooth noise as test out-distributions. Since the computation of adversarial AUCs (next paragraph) requires computationally expensive adversarial attacks, we restrict the evaluation on the out-distribution to a fixed subset of 1000 images (300 in the case of LSUN_CR) for the CIFAR experiments and 400 for the R.ImgNet models. We still use the entire test set for the in-distribution. We also show the results on additional test out-distributions in App. G.

Guaranteed and Adversarial AUC We use the confidence of the classifier as the feature to discriminate between in- and out-distribution samples. While in standard OOD detection one uses the area under the receiver-operator characteristic (AUC) to measure discrimination of in- from out-distribution, several prior works also study the worst-case AUC (WCAUC) [41, 4, 8, 9, 5], which is defined as the minimal AUC one can achieve if each out-distribution sample is allowed to be perturbed to reach maximal confidence within a certain threat model, which in our case is an l_∞-ball of radius ϵ. Note that an alternative formulation of a worst-case AUC as the worst-case across all samples from the out-distribution would turn out to be uninteresting, since it would necessarily be close to zero even if only a single sample gets assigned high-confidence, so we do not consider this notion here. Formally, the AUC and WCAUC of a feature $h : \mathbb{R}^d \to \mathbb{R}$ are defined as:

$$\mathrm{AUC}_h(p_1, p_2) = \mathop{\mathbb{E}}_{\substack{x\sim p_1\\ z\sim p_2}}\left[\mathbb{1}_{h(x)>h(z)}\right], \quad \mathrm{WCAUC}_h(p_1, p_2) = \mathop{\mathbb{E}}_{\substack{x\sim p_1\\ z\sim p_2}}\left[\mathbb{1}_{h(x)>\max\limits_{\|z'-z\|_\infty\leq\epsilon} h(z')}\right], \quad (12)$$

where p_1, p_2 are in-resp. out-distribution and the indicator function $\mathbb{1}$ returns 1 if the expression in its argument is true and 0 otherwise.

For all but one of our baselines, the OOD detecting feature h is the confidence of the classifier. Since the exact evaluation of the WCAUC is computationally infeasible, we compute an upper bound and

Table 2: **OOD performance:** For all models we report accuracy on the test set of the in-distribution and AUCs, guaranteed AUCs (GAUC), adversarial AUCs (AAUC) for different test out-distributions. The radius of the l_∞-ball for the adversarial manipulations of the OOD data is $\epsilon = 0.01$ for all datasets. The bias shift Δ that was used for ProoD is shown for each in-distribution. The AAUCs and GAUCs for ProoD tend to be very close, indicating remarkably tight certification bounds. Models with accuracy drop of $> 3\%$ relative to the model with highest accuracy are grayed out. Of the remaining models, we highlight the best OOD detection performance.

In: CIFAR10		CIFAR100			SVHN			LSUN_CR			Smooth		
	Acc	AUC	GAUC	AAUC	AUC	GAUC	AAUC	AUC	GAUC	AAUC	AUC	GAUC	AAUC
Plain	**95.01**	90.0	0.0	0.7	93.8	0.0	0.3	93.1	0.0	0.5	98.0	0.0	0.7
OE	94.91	**91.1**	0.0	0.9	97.3	0.0	0.0	**100.0**	0.0	2.7	**99.9**	0.0	1.5
ATOM	93.63	78.3	0.0	21.7	94.4	0.0	24.1	79.8	0.0	20.1	99.5	0.0	**73.2**
ACET	93.43	86.0	0.0	4.0	99.3	0.0	4.6	89.2	0.0	3.7	99.9	0.0	40.2
GOOD_{80}*	87.39	76.7	47.1	57.1	90.8	43.4	76.8	97.4	70.6	93.6	96.2	72.9	89.9
GOOD_{100}*	86.96	67.8	48.1	49.7	62.6	34.9	36.3	84.9	74.6	75.6	87.0	76.1	78.1
ProoD-Disc	-	62.9	57.1	57.8	72.6	65.6	66.4	78.1	71.5	72.3	59.2	49.7	50.4
ProoD $\Delta = 3$	94.99	89.8	**46.1**	**46.8**	**98.3**	**53.3**	**54.1**	**100.0**	**58.3**	**59.7**	**99.9**	**38.2**	38.8
In: CIFAR100		**CIFAR10**			**SVHN**			**LSUN_CR**			**Smooth**		
	Acc	AUC	GAUC	AAUC	AUC	GAUC	AAUC	AUC	GAUC	AAUC	AUC	GAUC	AAUC
Plain	**77.38**	**77.7**	0.0	0.4	81.9	0.0	0.2	76.4	0.0	0.3	86.6	0.0	0.4
OE	77.25	77.4	0.0	0.2	**92.3**	0.0	0.0	**100.0**	0.0	0.7	**99.5**	0.0	0.5
ATOM	68.32	78.3	0.0	50.3	91.1	0.0	67.0	95.9	0.0	75.6	98.2	0.0	80.7
ACET	73.02	73.0	0.0	1.4	97.8	0.0	0.7	75.8	0.0	2.6	99.9	0.0	12.8
ProoD-Disc	-	56.1	52.1	52.3	61.0	58.2	58.4	70.4	66.9	67.1	29.6	26.4	26.5
ProoD $\Delta = 5$	77.16	76.6	**17.3**	**17.4**	91.5	**19.7**	**19.8**	**100.0**	**22.5**	**23.1**	98.9	**9.0**	**9.0**
In: R.ImgNet		**Flowers**			**FGVC**			**Cars**			**Smooth**		
	Acc	AUC	GAUC	AAUC	AUC	GAUC	AAUC	AUC	GAUC	AAUC	AUC	GAUC	AAUC
Plain	96.34	92.3	0.0	0.5	92.6	0.0	0.0	92.7	0.0	0.1	**98.9**	0.0	8.6
OE	97.10	**96.9**	0.0	0.2	99.7	0.0	0.4	**99.9**	0.0	1.8	98.0	0.0	1.9
ProoD-Disc	-	81.5	76.8	77.3	92.8	89.3	89.6	90.7	86.9	87.3	81.0	74.0	74.8
ProoD $\Delta = 4$	**97.25**	**96.9**	**57.5**	**58.0**	**99.8**	**67.4**	**67.9**	**99.9**	**65.7**	**66.2**	98.6	**52.7**	**53.5**

*Uses different architecture of classifier, see "Baselines" in Section 4.2.

a lower bound on the WCAUC by finding $\underline{h}(z) \leq \max_{\|z'-z\|_\infty \leq \epsilon} h(z') \leq \bar{h}(z)$. We find the upper bound on the WCAUC - the adversarial AUC (AAUC) - by maximizing the confidence using an adversarial attack inside the l_∞-ball (i.e. finding an $\underline{h}$). We compute a lower bound on the WCAUC - the guaranteed AUC (GAUC) - by computing upper bounds on the confidence inside the l_∞-ball (i.e. $\bar{h}$) via IBP. For non-provable methods, no non-trivial upper bound $\bar{h} < \infty$ is available so their GAUCs are always 0. Note that our threat model is different from adversarial robustness on the in-distribution which neither our method nor the baselines pursue. Since practical OOD detection scenarios require the selection of a threshold, we also evaluate the false positive rate (FPR) at 95% true positive rate and show the results in App. F.

Vanishing gradients [47, 3] are a significant challenge for the evaluation of AAUCs [8] even more than in the evaluation of adversarial robustness on the in-distribution as the models are trained to be "flat" on the out-distribution. Thus we use an ensemble of variants of projected gradient descent (PGD) [37] as well as the black-box SquareAttack [1] with 5000 queries. We use APGD [13] (except on RImgNet, due to a memory leak) with 500 iterations and 5 random restarts. We also use a 200-step PGD attack with momentum of 0.9 and backtracking that starts with a step size of 0.1 which is halved every time a gradient step does not increase the confidence and gets multiplied by 1.1 otherwise. As stated above, the models are trained to be flat and thus the gradients can be exactly zero which renders gradient-based optimization impossible. Therefore it is important to use a variety of different starting points. For PGD we start from: i) a decontrasted version of the image, i.e. the point that minimizes the l_∞-distance to the grey image $0.5 \cdot \vec{1}$ within the threat model, ii) 3 uniformly drawn

samples from the threat model, and iii) 3 versions of the original image perturbed by Gaussian noise with $\sigma = 10^{-4}$ and then clipped to the threat model. We always clip to $[0, 1]^d$ at each step of the attack. For all attacks and models we directly optimize the final score used for OOD detection.

Baselines We compare to a normally trained baseline (Plain) and outlier exposure (OE), both trained using the same architecture and hyperparameters as the classifier in ProoD. For both ATOM and ACET we found the models' OOD detection to be much less adversarially robust than claimed in [9] (see App. E) so we retrained their models using the our architecture, threat model and training out-distribution with their original code (for CIFAR10/100). Running these adversarial training procedures on ImageNet resolution is infeasibly expensive. For GOOD we also retrain using OpenImages as training OOD dataset with the code from [8] (comparisons with their pre-trained models can be found in App. E). Since they are only available on CIFAR10, we tried to train models on CIFAR100 using their code and same hyperparameters and schedules as they used for CIFAR10. This only lead to models with accuracy below 25%, so we do not include these models in our evaluation. Since CCU was already shown to not provide benefits over OE on OOD data that is not very far from the in-distribution (e.g. uniform noise) [41, 8] we do not include it as baseline. We also evaluate the OOD-performance of the provable binary discriminator (ProoD-Disc) that we trained for ProoD. Note that this is not a classifier and is included only for reference. All results are in Table 2.

Results ProoD achieves non-trivial GAUCs on all datasets. As was also observed in [8], this shows that the IBP guarantees not only generalize to unseen samples but even to unseen distributions. In App. I we show that they even generalize to the larger threat model $\epsilon = 8/255$. In general, the gap between our GAUCs and AAUCs is extremely small. This shows that the seemingly simple IBP bounds can be remarkably tight, as has been observed in other works [17, 23]. It also shows that there would be very little benefit in applying stronger verification techniques like [10, 24, 15] in ProoD. Similarly, it demonstrates the strengths of our attack as there provably does not exist an attack that could lower the AAUCs on our ProoD model by more than 1.4% on any of the out-distributions. The bounds are also much tighter than for GOOD, which is likely due to the fact that for GOOD the confidence is much harder to optimize during an attack because it involves maximizing the confidence in an essentially random class.

For CIFAR10, on 3 out of 4 out-distributions ProoD's GAUCs are higher than ATOM's and ACET's AAUCs, i.e. our model's *provable* adversarial robustness exceeds the SOTA methods' *empirical* adversarial robustness in these cases. Note that this is *not* due to our retraining, because the authors' pre-trained models perform even more poorly (as shown in App. E). On CIFAR100, ProoD's guarantees are weaker and ATOM produces strong AAUCs. However, we observe that training both ACET and ATOM can produce inconsistent results, i.e. sometimes almost no robustness is achieved. For the successfully trained robust ATOM model on CIFAR100 we observe drastically reduced accuracy. Due to the difficulty in attacking these models, it is not unlikely that a more sophisticated attack could produce even lower AAUCs. Combined with the fact that both ACET and ATOM rely on expensive adversarial training procedures we argue that using ProoD is preferable in practice.

On CIFAR10, we see that ProoD's GAUCs are comparable to, if slightly worse than the ones of both GOOD_{80} and GOOD_{100}. Note that although the presented GOOD models are retrained, the same observations hold true when comparing to the pre-trained models (see App. E). However, we want to point out that ProoD achieves this while retaining both high accuracy and OOD performance, both of which are lacking for GOOD. It is also noteworthy that the GOOD models' memory footprints are over twice as large as ProoD's. Generally, for ProoD the accuracy is comparable to OE and the OOD performance is similar or marginally worse. Thus ProoD shows that it is possible to achieve certifiable adversarial robustness on the out-distribution while keeping very good prediction and OOD detection performance. Note that all methods struggle on separating CIFAR10 and CIFAR100 when using OpenImages as training OOD (as compared to 80M Tiny Images in App. E).

To the best of our knowledge with R.ImgNet we provide the first worst case OOD guarantees on high-resolution images. The GAUCs are higher than on CIFAR, indicating that meaningful certificates on higher resolution are more achievable on this task than one might expect. FGVC and Cars may seem simple to separate from the animals in R.ImgNet but this cannot be said for Flowers which are difficult to provably distinguish from images of insects on flowers.

In summary, ProoD achieves our goal of maintaining high accuracy and clean OOD detection performance while providing provably adversarially robust OOD detection. In fact, out of all the

methods that do not significantly impair the in-distribution accuracy, ProoD is the only method providing such guarantees as well while simultaneously having the highest empirical robustness. Also note that for applications where adversarial robustness on the in-distribution is desired despite the induced reduction in accuracy, one can combine our ProoD model with a robustly trained classifier. In App. K, we demonstrate that ProoD in fact improves the clean as well as the robust OOD detection performance in this setting.

Limitations & Impact Like our baselines, ProoD depends on a suitable training out-distribution being available. Also, as noted throughout the paper, our method only focuses on adversarial robustness around OOD samples and does not by itself achieve adversarial robustness around the in-distribution. Furthermore, our considered threat model is l_∞ and while the general architecture could also be applied to other threat models, this would require moving beyond IBP. We also show in App. F that non-trivial bounds on the adversarial FPR are beyond all current methods, including ProoD.

Since our work aims to make ML models safer and more reliable, we do not anticipate any misuse of the technology. Additionally, since our method does not rely on any adversarial training, the environmental impact of training is lower than for most of our baselines. However, there have been concerns about any use of the 80M Tiny Images dataset, which is why we include these results only in App. E for back-compatibility.

5 Conclusion

We have demonstrated how to combine a provably adversarially robust binary discriminator between in- and out-distribution with a standard classifier in order to simultaneously achieve high accuracy, high clean OOD detection performance as well as certified adversarially robust OOD detection. Thus, we have combined the best properties of previous work with only a small increase in total model size and only a single hyperparameter. This suggests that certifiable adversarial robustness on the out-distribution (as opposed to the in-distribution) is indeed possible without losing accuracy. We further showed how in our model simply enforcing negativity in the final weights of the discriminator fixes the problem of asymptotic overconfidence in ReLU classifiers. Training ProoD models is simple and stable and thus ProoD provides OOD guarantees that come (almost) for free.

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 (EXCnumber 2064/1, Project number 390727645), as well as from the DFG TRR 248 (Project number389792660). The authors thank the International Max Planck Research School for Intelligent Systems (IMPRS-IS) for supporting Alexander Meinke. We also thank Maximilian Augustin for helpful advice.

References

[1] M. Andriushchenko, F. Croce, N. Flammarion, and M. Hein. Square attack: a query-efficient black-box adversarial attack via random search. In *ECCV*, 2020.

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

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

[4] M. Augustin, A. Meinke, and M. Hein. Adversarial robustness on in-and out-distribution improves explainability. In *ECCV*, 2020.

[5] L. Berrada, S. Dathathri, K. Dvijotham, R. Stanforth, R. Bunel, J. Uesato, S. Gowal, M. P. Kumar, et al. Make sure you're unsure: A framework for verifying probabilistic specifications. In *NeurIPS*, 2021.

[6] A. Birhane and V. U. Prabhu. Large image datasets: A pyrrhic win for computer vision? In *WACV*, 2021.

[7] J. Bitterwolf, A. Meinke, M. Augustin, and M. Hein. Breaking down out-of-distribution detection: Many methods based on OOD training data estimate a combination of the same core quantities. In *ICML*, 2022.

[8] J. Bitterwolf, A. Meinke, and M. Hein. Certifiably adversarially robust detection of out-of-distribution data. *NeurIPS*, 2020.

[9] J. Chen, Y. Li, X. Wu, Y. Liang, and S. Jha. Atom: Robustifying out-of-distribution detection using outlier mining. In *ECML*, 2021.

[10] C.-H. Cheng, G. Nührenberg, and H. Ruess. Maximum resilience of artificial neural networks. In *International Symposium on Automated Technology for Verification and Analysis*, 2017.

[11] M. Cimpoi, S. Maji, I. Kokkinos, S. Mohamed, , and A. Vedaldi. Describing textures in the wild. In *Proceedings of the IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2014.

[12] F. Croce, M. Andriushchenko, V. Sehwag, E. Debenedetti, N. Flammarion, M. Chiang, P. Mittal, and M. Hein. Robustbench: a standardized adversarial robustness benchmark. In *Thirty-fifth Conference on Neural Information Processing Systems Datasets and Benchmarks Track*, 2021.

[13] F. Croce and M. Hein. Reliable evaluation of adversarial robustness with an ensemble of diverse parameter-free attacks. In *ICML*, 2020.

[14] E. D. Cubuk, B. Zoph, D. Mane, V. Vasudevan, and Q. V. Le. Autoaugment: Learning augmentation strategies from data. In *CVPR*, 2019.

[15] S. Dathathri, K. Dvijotham, A. Kurakin, A. Raghunathan, J. Uesato, R. Bunel, S. Shankar, J. Steinhardt, I. Goodfellow, P. Liang, et al. Enabling certification of verification-agnostic networks via memory-efficient semidefinite programming. In *NeurIPS*, 2020.

[16] 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.

[17] 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.

[18] S. Gowal, S.-A. Rebuffi, O. Wiles, F. Stimberg, D. A. Calian, and T. A. Mann. Improving robustness using generated data. *NeurIPS*, 2021.

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

[20] 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.

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

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

[23] N. Jovanović, M. Balunović, M. Baader, and M. Vechev. Certified defenses: Why tighter relaxations may hurt training? *arXiv:2102.06700*, 2021.

[24] G. Katz, C. Barrett, D. Dill, K. Julian, and M. Kochenderfer. Reluplex: An efficient smt solver for verifying deep neural networks. In *CAV*, 2017.

[25] A.-K. Kopetzki, B. Charpentier, D. Zügner, S. Giri, and S. Günnemann. Evaluating robustness of predictive uncertainty estimation: Are dirichlet-based models reliable? *arXiv:2010.14986*, 2020.

[26] J. Krause, M. Stark, J. Deng, and L. Fei-Fei. 3d object representations for fine-grained categorization. In *ICCV vision workshop*, 2013.

[27] A. Kristiadi, M. Hein, and P. Hennig. Being Bayesian, Even Just a Bit, Fixes Overconfidence in ReLU Networks. In *ICML*, 2020.

[28] A. Kristiadi, M. Hein, and P. Hennig. Fixing asymptotic uncertainty of bayesian neural networks with infinite relu features. *arXiv:2010.02709*, 2020.

[29] A. Krizhevsky and G. Hinton. Learning multiple layers of features from tiny images. Technical report, Citeseer, 2009.

[30] A. Kuznetsova, H. Rom, N. Alldrin, J. Uijlings, I. Krasin, J. Pont-Tuset, S. Kamali, S. Popov, M. Malloci, A. Kolesnikov, et al. The open images dataset v4. *International Journal of Computer Vision*, 2020.

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

[32] 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.

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

[34] W. Liu, X. Wang, J. Owens, and Y. Li. Energy-based out-of-distribution detection. *Advances in Neural Information Processing Systems*, 2020.

[35] D. Macêdo and T. Ludermir. Enhanced isotropy maximization loss: Seamless and high-performance out-of-distribution detection simply replacing the softmax loss. *arXiv:2105.14399*, 2021.

[36] D. Macêdo, T. I. Ren, C. Zanchettin, A. L. Oliveira, and T. Ludermir. Entropic out-of-distribution detection: Seamless detection of unknown examples. *IEEE Transactions on Neural Networks and Learning Systems*, 2021.

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

[38] S. Maji, E. Rahtu, J. Kannala, M. Blaschko, and A. Vedaldi. Fine-grained visual classification of aircraft. *arXiv:1306.5151*, 2013.

[39] A. Malinin and M. Gales. Predictive uncertainty estimation via prior networks. In *NeurIPS*, 2018.

[40] A. Malinin and M. Gales. Reverse kl-divergence training of prior networks: Improved uncertainty and adversarial robustness. In *NeurIPS*, 2019.

[41] A. Meinke and M. Hein. Towards neural networks that provably know when they don't know. In *ICLR*, 2020.

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

[43] 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.

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

[45] M.-E. Nilsback and A. Zisserman. Automated flower classification over a large number of classes. In *2008 Sixth Indian Conference on Computer Vision, Graphics & Image Processing*, 2008.

[46] A.-A. Papadopoulos, M. R. Rajati, N. Shaikh, and J. Wang. Outlier exposure with confidence control for out-of-distribution detection. *Neurocomputing*, 2021.

[47] N. Papernot, P. McDaniel, I. Goodfellow, S. Jha, Z. B. Celik, and A. Swami. Practical black-box attacks against machine learning. In *ACM ASIACCS*, 2017.

[48] J. Ren, P. J. Liu, E. Fertig, J. Snoek, R. Poplin, M. A. DePristo, J. V. Dillon, and B. Lakshminarayanan. Likelihood ratios for out-of-distribution detection. In *NeurIPS*, 2019.

[49] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, A. C. Berg, and L. Fei-Fei. ImageNet Large Scale Visual Recognition Challenge. *International Journal of Computer Vision (IJCV)*, 2015.

[50] 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.

[51] M. Sensoy, L. Kaplan, and M. Kandemir. Evidential deep learning to quantify classification uncertainty. In *NeurIPS*, 2018.

[52] 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.

[53] D. Tsipras, S. Santurkar, L. Engstrom, A. Turner, and A. Madry. Robustness may be at odds with accuracy. In *ICLR*, 2018.

[54] P. Xu, K. A. Ehinger, Y. Zhang, A. Finkelstein, S. R. Kulkarni, and J. Xiao. Rich feature hierarchies for accurate object detection and semantic segmentation. *arXiv:1504.06755*, 2015.

[55] 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.

[56] 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.

[57] B. Zhou, A. Lapedriza, A. Khosla, A. Oliva, and A. Torralba. Places: A 10 million image database for scene recognition. *IEEE transactions on pattern analysis and machine intelligence*, 40(6):1452–1464, 2017.

Checklist

1. For all authors...
   (a) Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope? [Yes]
   (b) Did you describe the limitations of your work? [Yes]
   (c) Did you discuss any potential negative societal impacts of your work? [Yes]
   (d) Have you read the ethics review guidelines and ensured that your paper conforms to them? [Yes]
2. If you are including theoretical results...
   (a) Did you state the full set of assumptions of all theoretical results? [Yes]
   (b) Did you include complete proofs of all theoretical results? [Yes]
3. If you ran experiments...
   (a) Did you include the code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL)? [Yes]
   (b) Did you specify all the training details (e.g., data splits, hyperparameters, how they were chosen)? [Yes]
   (c) Did you report error bars (e.g., with respect to the random seed after running experiments multiple times)? [Yes] In order to be mindful of our resource consumption, we limited the measurement of error bars to a single in-distribution, see App. J.
   (d) Did you include the total amount of compute and the type of resources used (e.g., type of GPUs, internal cluster, or cloud provider)? [Yes]
4. If you are using existing assets (e.g., code, data, models) or curating/releasing new assets...
   (a) If your work uses existing assets, did you cite the creators? [Yes]
   (b) Did you mention the license of the assets? [Yes] See App. D.
   (c) Did you include any new assets either in the supplemental material or as a URL? [N/A]
   (d) Did you discuss whether and how consent was obtained from people whose data you're using/curating? [N/A]
   (e) Did you discuss whether the data you are using/curating contains personally identifiable information or offensive content? [Yes]
5. If you used crowdsourcing or conducted research with human subjects...
   (a) Did you include the full text of instructions given to participants and screenshots, if applicable? [N/A]
   (b) Did you describe any potential participant risks, with links to Institutional Review Board (IRB) approvals, if applicable? [N/A]
   (c) Did you include the estimated hourly wage paid to participants and the total amount spent on participant compensation? [N/A]

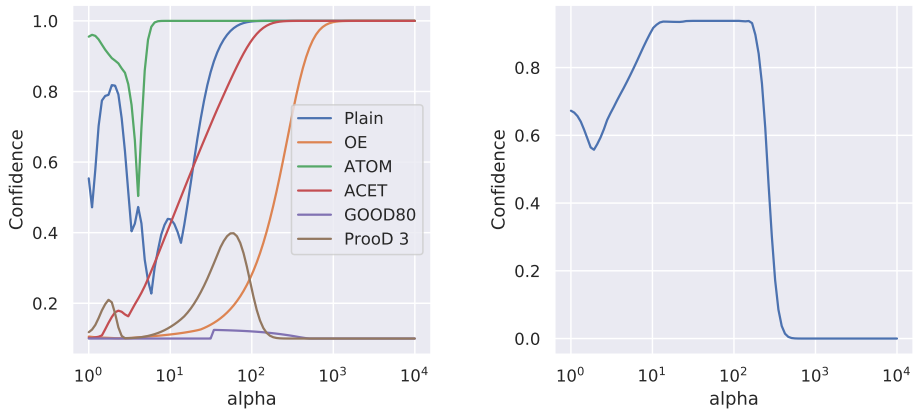

Figure 2: **Left, Asymptotic confidence:** We plot the mean confidence in the predicted in-distribution class for different models as one moves away from CIFAR100 samples along the trajectories $x + \alpha n$, where $n \in [-0.5, 0.5]^d$ and $\alpha \geq 0$. Only GOOD and ProoD converge to uniform confidence. **Right, Adversarial asymptotic confidence:** We try to find adversarial directions in which ProoD remains at a constant high confidence, as opposed to converging to low confidence. We plot the *maximum* of $\hat{p}(i|x)$ across 100 adversarially chosen directions as one moves further in these directions by factors of α. Note that $\hat{p}(i|x) \to 0$ implies $\hat{p}(y|x) \to \frac{1}{K}$.

A Adversarial Asymptotic Overconfidence

According to the authors of [20], under mild conditions, we should expect to find asymptotic overconfidence in all ReLU networks and almost all directions. In order to empirically evaluate this, we take different models that were trained on CIFAR10 and evaluate their confidence on different CIFAR100 samples. For each sample x we track the confidence, $\max_k \hat{p}(k|x)$, along a trajectory in a uniform noise direction $x + \alpha n$, where $n \in [-0.5, 0.5]^d$ and $\alpha \geq 0$. The mean confidence across 100 such trajectories is shown on the left side of Figure 2. Even the models that produce low confidences on the original OOD sample asympotically converge to maximal confidence far away. The only exceptions here are GOOD and ProoD and only ProoD can guarantee that the confidence cannot converge to 1.

However, even though the architecture provably prevents arbitrarily overconfident predictions and Theorem 1 ensures that most directions will indeed converge to uniform, it is, in principle, possible to find directions where the confidence $\hat{p}(i|x)$ remains constant if the condition $Ux \neq 0$ in Theorem 1 is not satisfied. We attempted to find such directions by running the following type of attack. We start from a random point $x \in [-0.5, 0.5]^d$ that we project onto a sphere of radius 100. We now run gradient descent (for 20000 steps), maximizing $g(x)$ while projecting onto the sphere at each step (unnormalized gradients with step size 0.1 for the first 10000 steps and 0.01 for the last 10000 steps). We then increase the radius to 1000 and run an additional 20000 steps with step size 0.1. We rescale the resulting direction vector down to an l_∞-ball of norm 1 and compute the confidence $\hat{p}(i|x)$ as a function of the scaling in the adversarial directions. We show the resulting scale-wise *maximum* over 100 adversarial directions in Figure 2. Note that even the worst-case over 100 adversarially found directions decays to 0 asymptotically, thus empirically confirming the practical utility of Theorem 1. Note that the value of $\hat{p}(i|x)$ converging to 0 implies that the confidence of the ProoD model $\hat{p}(y|x)$ converges to 10%.

In Figure 2 GOOD also stands out as having low confidence in all directions that we studied. This is because in all the asymptotic regions that we looked at, the pre-activations of the penultimate layer are all negative. If one moves outward and these pre-activations only get more negative in all directions far away from the data, the confidence does, in fact, remain low. Unfortunately, it also leads to gradients that are precisely zero, which is why the same attack can not be applied here. However, there is no guarantee that GOOD does not also get in some direction asymptotically overconfident.

Table 3: **Architecture:** The architectures that are used for the binary discriminators. Each convolutional layer is directly followed by a ReLU.

CIFAR	R.ImgNet
Conv2d(3, 128)	Conv2d(3, 128)
Conv2d(128, 256)$_{s=2}$	AvgPool(2)
Conv2d(256, 256)	Conv2d(128, 256)$_{s=2}$
AvgPool(2)	AvgPool(2)
FC(16384, 128)	Conv2d(256, 256)
FC(128, 1)	AvgPool(2)
	FC(50176, 128)
	FC(128, 1)

B Separate Training for ProoD

In Section 2.1 we describe semi-joint training of $\hat{p}(y|x)$. However, as pointed out in that section, it is possible to separately train a certifiable binary discriminator $\hat{p}(i|x)$ and an OOD aware classifier $\hat{p}(y|x, i)$ and to then simply combine them via Eq. (2). We call this method separate training ProoD-S and evaluate it by using an OE trained model for $\hat{p}(y|x, i)$. The two versions of the methods ProoD and ProoD-S are also described in Algorithm 1 and Algorithm 2. We show the results in Table 4, where we repeat the results for OE and ProoD for the reader's convenience. Note that OE and ProoD-S must always have the same accuracy on the in-distribution since they use the same model for classification (note that (2) preserves the ranking of $\hat{p}(y|x, i)$).

We see that the AUCs of ProoD-S are almost identical to those of OE. Without almost any loss in performance ProoD-S manages to provide non-trivial GAUCs. However, as one would expect, the semi-jointly trained ProoD provides stronger guarantees at similar clean performance. Nonetheless, this post-hoc method of adding some amount of certifiability to an existing system may be interesting in applications where retraining a deployed model from scratch is infeasible.

Table 4: **Separate training:** Addendum to Table 2 showing the AUCs, GAUCs and AAUCs of ProoD-S on all datasets. The accuracy must always be identical to that of OE and the clean AUCs are also very similar to those of OE. The guarantees are almost always strictly weaker than those provided by the semi-jointly trained ProoD.

In: CIFAR10		CIFAR100			SVHN			LSUN_CR			Smooth		
	Acc	AUC	GAUC	AAUC	AUC	GAUC	AAUC	AUC	GAUC	AAUC	AUC	GAUC	AAUC
OE	94.91	**91.1**	0.0	0.9	97.3	0.0	0.0	**100.0**	0.0	2.7	**99.9**	0.0	1.5
ProoD-S $\Delta=3$	94.91	89.3	44.7	45.3	97.3	51.8	52.6	**100.0**	56.7	57.7	**99.9**	36.7	37.6
ProoD $\Delta=3$	**94.99**	89.8	**46.1**	**46.8**	**98.3**	**53.3**	**54.1**	**100.0**	**58.3**	**59.7**	**99.9**	**38.2**	**38.8**
In: CIFAR100		**CIFAR10**			**SVHN**			**LSUN_CR**			**Smooth**		
	Acc	AUC	GAUC	AAUC	AUC	GAUC	AAUC	AUC	GAUC	AAUC	AUC	GAUC	AAUC
OE	**77.25**	**77.4**	0.0	0.2	**92.3**	0.0	0.0	**100.0**	0.0	0.7	**99.5**	0.0	0.5
ProoD-S $\Delta=5$	**77.25**	**77.4**	17.2	17.3	**92.3**	19.5	19.6	**100.0**	22.4	22.6	**99.5**	**9.0**	**9.1**
ProoD $\Delta=5$	77.16	76.6	**17.3**	**17.4**	91.5	**19.7**	**19.8**	**100.0**	**22.5**	**23.1**	98.9	**9.0**	9.0
In: R.ImgNet		**Flowers**			**FGVC**			**Cars**			**Smooth**		
	Acc	AUC	GAUC	AAUC	AUC	GAUC	AAUC	AUC	GAUC	AAUC	AUC	GAUC	AAUC
OE	97.10	**96.9**	0.0	0.2	99.7	0.0	0.4	**99.9**	0.0	1.8	98.0	0.0	1.9
ProoD-S $\Delta=4$	97.10	**96.9**	50.1	50.7	99.7	59.7	60.6	**99.9**	57.9	58.9	98.0	40.8	42.3
ProoD $\Delta=4$	**97.25**	**96.9**	**57.5**	**58.0**	**99.8**	**67.4**	**67.9**	**99.9**	**65.7**	**66.2**	**98.6**	**52.7**	**53.5**

C Proof of Theorem 1

The following result of [20] basically says that as one moves to infinity by upscaling a vector one eventually ends up in a polytope which extends to infinity. We use this in the proof of our Theorem.

Lemma 1 ([20]). *Let $\{Q_r\}_{r=1}^R$ be the set of convex polytopes on which a ReLU-network $f : \mathbb{R}^d \to \mathbb{R}^K$ is an affine function, that is for every $k \in \{1, \dots, R\}$ and $x \in Q_k$ there exists $V^k \in \mathbb{R}^{K \times d}$ and $c^k \in \mathbb{R}^K$ such that $f(x) = V^k x + c^k$. For any $x \in \mathbb{R}^d$ with $x \neq 0$ there exists $\alpha \in \mathbb{R}$ and $t \in \{1, \dots, R\}$ such that $\beta x \in Q_t$ for all $\beta \geq \alpha$.*

Theorem 1. *Let $x \in \mathbb{R}^d$ with $x \neq 0$ and let $g : \mathbb{R}^d \to \mathbb{R}$ be the ReLU-network of the binary discriminator (with the last activation being a non-leaky ReLU). Denote by $\{Q_r\}_{r=1}^R$ the finite set of polytopes on which g is affine (exists by Lemma 1 in App. C). Denote by Q_t the polytope such that $\beta x \in Q_t$ for all $\beta \geq \alpha$ and let $x^{(L-1)}(z) = Uz + d$ with $U \in \mathbb{R}^{n_{L-1} \times d}$ and $d \in \mathbb{R}^{n_{L-1}}$ be the output of the pre-logit layer of g for $z \in Q_t$. If $Ux \neq 0$, then $\lim_{\beta \to \infty} \hat{p}(y|\beta x) = \frac{1}{K}$.*

Proof. We note that with a similar argument as in the derivation of (4) it holds

$$\hat{p}(y|\beta x) \leq \hat{p}(i|\beta x) + \frac{1}{K}\big(1 - \hat{p}(i|\beta x)\big) = \frac{K-1}{K}\hat{p}(i|\beta x) + \frac{1}{K}. \tag{13}$$

Using Lemma 1 we know that there exist a polytope Q_t such that $\beta x \in Q_t$ for all $\beta \geq \alpha$. Thus for all $\beta \geq \alpha$ it holds that $\beta x \in Q_t$ so that

$$\hat{p}(i|\beta x) = \frac{1}{1 + e^{-g(\beta x)}} = \frac{1}{1 + e^{\left\langle W_g^{(L_g)}, U\beta x + d \right\rangle + b_g^{(L_g)}}}.$$

As $x_i^{(L-1)}(x) \geq 0$ for all $x \in \mathbb{R}^d$ it has to hold $(\beta U x + d)_i \geq 0$ for all $\beta \geq \alpha$ and $i = 1, \dots, n_{L-1}$. This implies that $(Ux)_i \geq 0$ for all $i = 1, \dots, n_{L-1}$ and since $Ux \neq 0$ there has to exist at least one component i^* such that $(Ux)_{i^*} > 0$. Moreover, $W_g^{(L_g)}$ has strictly negative components and thus for all $\beta \geq \alpha$ it holds

$$g(\beta x) = \left\langle W_g^{(L_g)}, U\beta x + d \right\rangle + b_g^{(L_g)} = \beta \left\langle W_g^{(L_g)}, Ux \right\rangle + \left\langle W_g^{(L_g)}, d \right\rangle + b_g^{(L_g)}.$$

As $\left\langle W_g^{(L_g)}, Ux \right\rangle < 0$ we get $\lim_{\beta \to \infty} g(x) = -\infty$ and thus

$$\lim_{\beta \to \infty} \hat{p}(i|\beta x) = 0.$$

Plugging this into (13) yields the result. □

D Experimental Details

Datasets We use CIFAR10 and CIFAR100 [29] (MIT license), SVHN [43] (free for non-commercial use), LSUN [55] (no license), the ILSVRC2012 split of ImageNet [16, 49] (free for non-commercial use), FGVC-Aircraft [38] (free for non-commercial use), Stanford Cars [26] (free for non-commercial use), OpenImages v4 [30] (images have a CC BY 2.0 license), Oxford 102 Flower [45] (no license) as well as 80M Tiny Images [52] (no license given, see also App. E). For the train/test splits we use the standard splits, except on 80M Tiny Images where we treat a random but fixed subset of 1000 images in the first 1,000,000 as our test set. For all datasets that get used as a test out-distribution we use a random but fixed subset of 1000 images.

Following [8], the smooth noise that is used is generated as follows. Uniform noise is generated and then smoothed using a Gaussian filter with a width that is drawn uniformly at random in $[1, 2.5]$. Each datapoint is then shifted and scaled linearly to ensure full range in $[0, 1]$, i.e. $x' = \frac{x - \min(x)}{\max(x) - \min(x)}$.

Binary Training The architecture that we use for the binary discriminator is relatively shallow (5 linear layers). The architecture is shown in Table 3. Our results are fairly robust to the exact choice of architecture and significantly larger models do not necessarily lead to better results as we show in App. H. Similarly to [56, 8], we use long training schedules, running Adam for 1000 epochs, with an initial learning rate of $1e-4$ that we decrease by a factor of 5 on epochs $500, 750$ and 850 and with a

Algorithm 1 Training of ProoD

Require: Training data on in-distribution $(X, Y)_{n=1}^{N}$ and out-distribution $(Z)_{m=1}^{M}$, untrained classifier f_θ, untrained binary discriminator g_η, adversarial radius ϵ, batch size b, number of classes K, bias shift Δ

for t in discriminator training steps **do** ▷ Train the binary discriminator
  $x \leftarrow$ sample minibatch from X
  $z \leftarrow$ sample minibatch from Z
  loss $\leftarrow 0$
  for r in $1..b$ **do** ▷ Parallelized in practice
    $\underline{z}_r \leftarrow \mathrm{Clip}(\mathrm{z_r} - \epsilon, 0, 1)$ ▷ Keeps perturbed values in box
    $\overline{z}_r \leftarrow \mathrm{Clip}(\mathrm{z_r} + \epsilon, 0, 1)$
    $\overline{g}_r \leftarrow \mathrm{IntervalBoundPropagation}(\mathrm{g}_\eta, \underline{\mathrm{z}}_\mathrm{r}, \overline{\mathrm{z}}_\mathrm{r})$ ▷ An algorithm that provides upper bounds on the output of g_η
    loss $\leftarrow$ loss $+ \frac{1}{\mathrm{b}} \log\big(1 + \mathrm{e}^{-\mathrm{g}_\eta(\mathrm{x_r})}\big) + \frac{1}{\mathrm{b}} \log\left(1 + \mathrm{e}^{\overline{\mathrm{g}}_\mathrm{r}}\right)$
  end for
  $\eta \leftarrow$ SGD_Update(η, loss, t)
end for
$g_\eta \leftarrow g_\eta + \Delta$ ▷ Apply the bias shift
for t in classifier training steps **do** ▷ Train the classifier using semi-joint training
  $x, y \leftarrow$ sample minibatch from X, Y
  $z \leftarrow$ sample minibatch from Z
  loss $\leftarrow 0$
  for r in $1..b$ **do**
    $p_i \leftarrow \mathrm{SoftMax}(\mathrm{f}_\theta(\mathrm{x_r}), \mathrm{y_r}) \cdot \mathrm{Sigmoid}(\mathrm{g}_\eta(\mathrm{x_r})) + \frac{1}{\mathrm{K}}\big(1 - \mathrm{Sigmoid}(\mathrm{g}_\eta(\mathrm{x_r}))\big)$
    $p_o \leftarrow \frac{1}{K}\big(1 - \mathrm{Sigmoid}(\mathrm{g}_\eta(\mathrm{z_r}))\big) + \frac{1}{\mathrm{K}} \sum_{\mathrm{c}=1}^{\mathrm{K}} \mathrm{SoftMax}(\mathrm{f}_\theta(\mathrm{z_r}), \mathrm{c}) \cdot \mathrm{Sigmoid}(\mathrm{g}_\eta(\mathrm{z_r}))$
    loss $\leftarrow$ loss $+ \frac{1}{\mathrm{b}}\mathrm{p_i} + \frac{1}{\mathrm{b}}\mathrm{p_o}$
  end for
  $\theta \leftarrow$ SGD_Update(θ, loss, t) ▷ Only θ gets updated - not η.
end for

batch size of 128 from the in-distribution and 128 from the out-distribution (for R.ImgNet: 50 epochs with drops at 25, 35, 45, batch sizes 32). In order to avoid large losses we also use a simple ramp up schedule for the ϵ used in IBP and we downweight the out-distribution loss during the initial phase of training by a scalar κ. Both ϵ and κ are increased linearly from 0 to their final values (0.01 and 1, respectively) over the first 300 epochs (for R.ImgNet over the first 25 epochs). Compared to the training of [8] which sometimes fails, we found that training of the binary discriminator is very stable and even 100 epochs on CIFAR would be sufficient, but we found that longer training lead to slightly better results. Weight decay is set to $5 \cdot 10^{-4}$, but is disabled for the weights in the final layer. As data augmentation we use AutoAugment [14] for CIFAR and simple 4 pixel crops and reflections on R.ImgNet. The strict negativity of the weights leads to a negative bias of g which can cause problems at an early stage if the $b_g^{(L_g)}$ is initialized at 0 and thus we choose 3 as initialization. All binary discriminators were trained on single 2080Ti GPUs, managed on a SLURM cluster. Overall, the training of a provable discriminator takes around 16h on CIFAR and 44h on R.ImgNet (wall clock time including evaluations and logging on each epoch).

Semi-Joint Training On CIFAR we train for 100 epochs using SGD with momentum of 0.9 and a learning rate of 0.1 that drops by a factor of 10 on epochs $50, 75$ and 90 (on R.ImgNet 75 epochs with drops at 30 and 60). For all datasets we train using a batch size of 128 (plus 128 out-distribution samples in the case of OE). The CIFAR experiments were run on single 2080Ti GPUs. This takes about 4h20min in wall clock time. In order to fit batches of 128 in-distribution samples and 128 out-distribution samples on R.ImgNet we had to train using 4 V100 GPUs in parallel. Because of batch normalization in multi-GPU training it is important to not simply stack the batches but to interlace in- and out-distribution samples. The wall clock time was around 15h for the semi-joint training on R.ImgNet. For selecting the bias we use the procedure described in Section 4. The trade-off curves for the AUC and GAUC on CIFAR100 and R.ImgNet are given in Figure 3.

Algorithm 2 Training of ProoD-S

Require: Training data on in-distribution $(X, Y)_{n=1}^{N}$ and out-distribution $(Z)_{m=1}^{M}$, untrained classifier f_θ, untrained binary discriminator g_η, adversarial radius ϵ, batch size b, number of classes K, bias shift Δ

for t in discriminator training steps **do** ▷ Train the binary discriminator
  $x \leftarrow$ sample minibatch from X
  $z \leftarrow$ sample minibatch from Z
  loss $\leftarrow 0$
  for r in $1..b$ **do** ▷ Parallelized in practice
    $\underline{z}_r \leftarrow \mathrm{Clip}(\mathrm{z_r} - \epsilon, 0, 1)$ ▷ Keeps perturbed values in box
    $\overline{z}_r \leftarrow \mathrm{Clip}(\mathrm{z_r} + \epsilon, 0, 1)$
    $\overline{g}_r \leftarrow \mathrm{IntervalBoundPropagation}(\mathrm{g}_\eta, \underline{\mathrm{z}}_\mathrm{r}, \overline{\mathrm{z}}_\mathrm{r})$ ▷ An algorithm that provides upper bounds on the output of g_η
    loss $\leftarrow$ loss $+ \frac{1}{\mathrm{b}} \log\left(1 + \mathrm{e}^{-\mathrm{g}_\eta(\mathrm{x_r})}\right) + \frac{1}{\mathrm{b}} \log\left(1 + \mathrm{e}^{\overline{\mathrm{g}}_\mathrm{r}}\right)$
  end for
  $\eta \leftarrow$ SGD_Update(η, loss, t)
end for
$g_\eta \leftarrow g_\eta + \Delta$ ▷ Apply the bias shift
for t in classifier training steps **do** ▷ Train the classifier completely separately
  $x, y \leftarrow$ sample minibatch from X, Y
  $z \leftarrow$ sample minibatch from Z
  loss $\leftarrow 0$
  for r in $1..b$ **do**
    $p_i \leftarrow \mathrm{SoftMax}(\mathrm{f}_\theta(\mathrm{x_r}), \mathrm{y_r})$ ▷ The difference to Alg. 1 is here.
    $p_o \leftarrow \frac{1}{K} \sum_{c=1}^{K} \mathrm{SoftMax}(\mathrm{f}_\theta(\mathrm{z_r}), \mathrm{c})$ ▷ Note that g_η does not appear in this loss.
    loss $\leftarrow$ loss $+ \frac{1}{\mathrm{b}}\mathrm{p_i} + \frac{1}{\mathrm{b}}\mathrm{p_o}$
  end for
  $\theta \leftarrow$ SGD_Update(θ, loss, t)
end for

E 80M Tiny Images as Training Out-Distribution

The 80M Tiny Images dataset has been retracted by the authors due to concerns over offensive class labels [6]. We support the decision of the community to move away from the use of 80M Tiny Images, so we choose to train our CIFAR models using a downscaled version of OpenImages v4 [30] as a training out-distribution. However, since all prior work used this dataset, we present results on 80M Tiny Images here in order to compare ProoD's performance to prior baselines. We encourage the community to use the results in Table 2 for future comparisons.

Our results on 80M Tiny Images are shown in Table 5. Apart from the models shown in Table 2 we add here the pre-trained energy-based OOD detector EB from [34] as an additional baseline for clean OOD detection. As EB is not trained robustly, as expected EB has low AAUCs. In terms of clean OOD detection it performs similarly to OE but with worse results on the more difficult OOD detection task CIFAR10 vs CIFAR100 and vice versa. For GOOD we use the pre-trained models from [8]. For ATOM and ACET we use the pre-trained models from [9]. Note that these use the densenet architecture and were actually trained to withstand attacks in the much stronger threat model of $\epsilon = \frac{8}{255}$. In the original paper, the authors claim near perfect AAUCs on this task but we can show that this is a case of gross overestimation of robustness. Even on the easier threat model $\epsilon = 0.01$ that we test in Table 5, their robustness is almost non-existent for both ATOM models and the CIFAR100 ACET model. The fact that their adversarial attacks were unable to find these samples clearly demonstrates that evaluating models adversarially is very difficult and potentially unreliable. Because of this we believe that our guarantees are a valuable contribution to the community.

F False Positive Rates

Since in a practical setting a threshold for OOD detection ultimately has to be chosen, it can be interesting to study the false positive rate at a fixed threshold. It is relatively standard to pick the

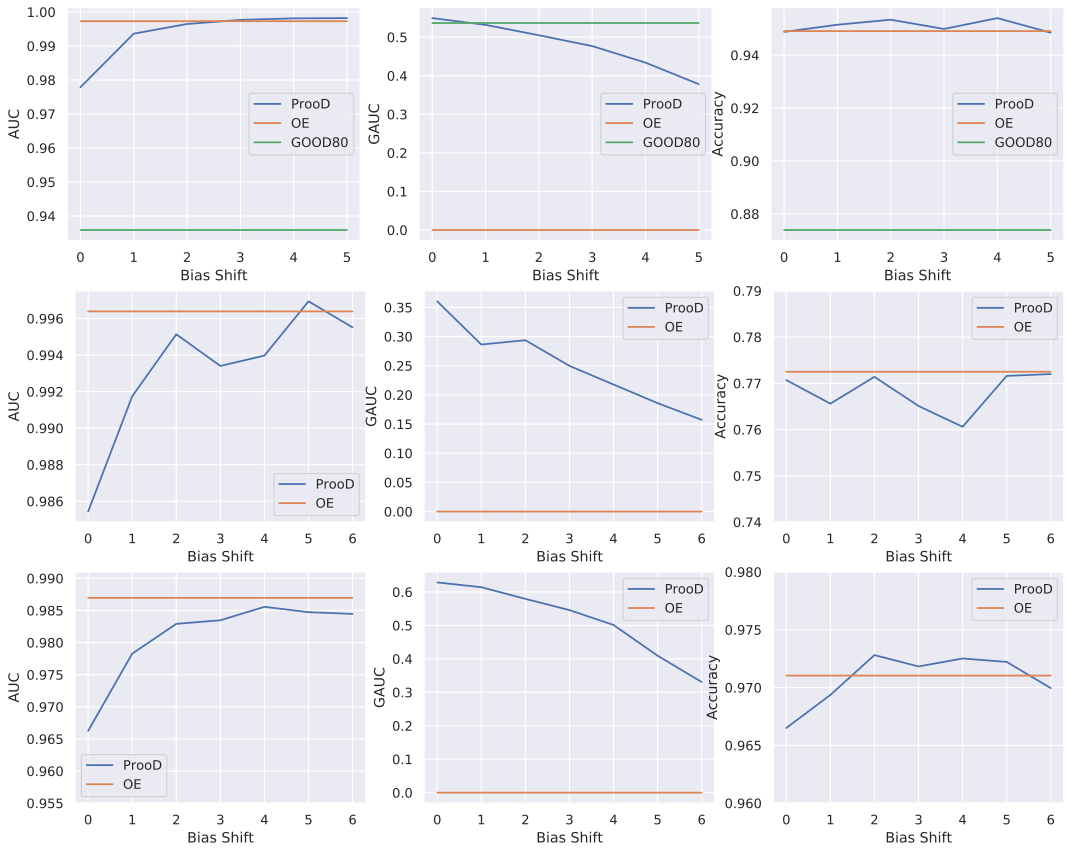

Figure 3: **Bias selection for CIFAR100 and RImgNet:** Using CIFAR10 (top), CIFAR100 (middle) and R.ImgNet (bottom) as the in-distribution and the test set of OpenImages (or NotR.ImgNet respectively) as OOD we plot the test accuracy, AUC and GAUC as a function of the bias shift Δ (see Eq. (9)).

false positive rate at 95% true positive rate (called FPR in Table 6), where low values are desirable. We show the results for all methods and datasets in Table 6. Although ProoD has similarly good performance as OE on this task, it fails to give non-trivial guarantees. Therefore achieving stronger bounds on the worst-case FPR is an interesting task for future work.

G Additional Datasets

In addition to the results shown in Table 2, it is interesting to study how ProoD performs on additional datasets as well as the test set of the out-distribution it was trained on. For the CIFAR datasets, we report LSUN crops, LSUN_resize, Places365 [57], iSUN [54], Textures [11], 80M Tiny Images and uniform noise in Table 7. On RImgNet we report uniform noise and the training out-distribution in Table 8.

As in Table 2 the clean performance of ProoD is comparable to that of OE, but it achieves non-trivial GAUC. On CIFAR10, GOOD_{100} achieves almost perfect GAUC against uniform noise, which comes at the price of significantly worse clean AUCs on all other out-distributions, see Table 2. Almost all methods achieve very high AAUCs on uniform noise, but it is not clear if a sufficiently powerful attack could lower those scores significantly. The unusually large gap between the GAUC and AAUC of ProoD would seem to indicate that that might be the case. Surprisingly, the robustness characteristics of both ATOM and ACET vary wildly between the different datasets, sometimes appearing to be perfectly robust and on other datasets displaying no robustness at all. Especially surprising are the relatively low AAUCs on the test set of the training out-distribution.

Table 5: **Training with 80M Tiny Images:** We repeat the evaluation from Table 2 for models that were trained using 80M Tiny Images as out-distribution instead of OpenImages. Plain is identical to before and is just repeated for the reader's convenience. Note that the conclusions from the main paper still hold, which indicates that our method is robust to changes in the choice of training out-distribution. For ATOM and ACET we compare to pre-trained models from [9]. Note that these models show almost no robustness on CIFAR100 - despite the far stronger claims in [9]. Models with accuracy drop of $> 3\%$ relative to the model with highest accuracy are grayed out. Of the remaining models, we highlight the best OOD detection performance.

In: CIFAR10		CIFAR100			SVHN			LSUN_CR			Smooth		
	Acc	AUC	GAUC	AAUC	AUC	GAUC	AAUC	AUC	GAUC	AAUC	AUC	GAUC	AAUC
Plain	95.01	90.0	0.0	0.6	93.8	0.0	0.1	93.1	0.0	0.5	98.2	0.0	0.6
OE	**95.53**	**96.1**	0.0	6.0	99.2	0.0	0.4	99.5	0.0	15.2	99.0	0.0	11.3
EB*	95.22	93.8	0.0	2.8	99.3	0.0	0.0	99.5	0.0	6.0	99.4	0.0	3.5
ATOM†	95.20	93.7	0.0	14.4	**99.6**	0.0	8.6	**99.7**	0.0	40.0	99.6	0.0	18.8
ACET†	91.48	91.2	0.0	80.5	95.3	0.0	87.6	98.9	0.0	95.0	99.9	0.0	98.3
GOOD$_{80}$*	90.13	87.2	42.5	63.9	94.2	37.5	67.4	93.3	55.2	83.6	95.3	57.3	88.5
GOOD$_{100}$*	90.14	70.7	54.5	55.0	74.9	56.3	56.6	75.2	61.0	61.6	81.4	66.6	67.5
ProoD-Disc	-	67.4	61.0	61.7	73.2	65.5	66.4	78.0	72.2	72.7	82.3	71.5	72.9
ProoD $\Delta = 3$	95.47	96.0	**41.9**	**43.9**	99.5	**48.8**	**49.4**	99.6	**47.6**	**53.1**	**99.7**	**55.8**	**57.0**
In: CIFAR100		**CIFAR10**			**SVHN**			**LSUN_CR**			**Smooth**		
	Acc	AUC	GAUC	AAUC	AUC	GAUC	AAUC	AUC	GAUC	AAUC	AUC	GAUC	AAUC
Plain	**77.38**	77.7	0.0	0.3	81.9	0.0	0.2	76.4	0.0	0.3	88.8	0.0	0.5
OE	77.28	**83.9**	0.0	0.8	92.8	0.0	0.1	**97.4**	0.0	4.6	97.6	0.0	0.9
EB*	75.70	77.4	0.0	0.8	**96.5**	0.0	0.0	96.7	0.0	5.9	**98.9**	0.0	4.3
ATOM†	75.06	64.3	0.0	0.2	93.6	0.0	0.2	97.5	0.0	9.3	98.5	0.0	15.0
ACET†	74.43	79.8	0.0	0.2	90.2	0.0	0.0	96.0	0.0	2.1	92.9	0.0	0.3
ProoD-Disc	-	53.8	50.3	50.4	73.1	69.8	69.9	68.1	63.8	64.0	67.2	63.8	63.9
ProoD $\Delta = 1$	76.79	80.5	**23.1**	**23.2**	93.7	**33.9**	**34.0**	97.2	**29.6**	**30.4**	**98.9**	**29.7**	**31.3**

* Pre-trained WideResnet from [34].
†Densenet architecture, using models from [9] pre-trained with $\epsilon = \frac{8}{255}$.
*CNN architecture using pre-trained models from [8].

H Size Ablation for Binary Discriminator

Since larger models should typically lead to better performance, we investigated the impact that model size has on the performance of our binary discriminator. We retrained ProoD-Disc models with CIFAR10 as in-distribution and 80M Tiny images as the out-distribution (since our dataloader is faster than for OpenImages which speeds up training). Since longer schedules only slightly improve results we also used shorter schedules with only 300 epochs where ϵ and κ linearly increase from 0 to 0.01 and 1.0 respectively within the first 100 epochs and the learning rate drops occur at 150, 200 and 250 epochs. As architectures we use 10 different CNNs with different widths and depths ranging from 5 to 8 layers (for the precise architectures please refer to sizes {S, XL_b, XS, SR, SR2, C1, C3s, C3, C2, C4} in the file `provable_classifiers.py` of the code provided at `https://github.com/AlexMeinke/Provable-OOD-Detection`). We present scatter plots of the models' their performance on CIFAR100 and 80M Tiny Images in Figure 4 against their size. Clearly, there is no correlation between model size and performance and most differences are rather small, justifying our choice of a fairly small architecture in the main paper.

I Generalization to Larger Threat Model

Since $\epsilon = 0.01$ is a relatively weak threat model we evaluate if ProoD's guarantees actually generalize to the much stronger $\epsilon = \frac{8}{255} \approx 0.031$ that is standard in much of the literature on adversarial robustness. We use the exact same CIFAR models from Table 2 and show the results of our evaluation at $\epsilon = \frac{8}{255}$ in Table 9.

Table 6: **False positive rates:** For all models we report accuracy on the test of the in-distribution and the false positive rate at 95% true positive rate (FPR) (smaller is better). We also show the adversarial FPR (AFPR) and the guaranteed FPR (GFPR) for different test out-distributions. The radius of the l_∞-ball for the adversarial manipulations of the OOD data is $\epsilon = 0.01$ for all datasets. The bias shift Δ that was used for ProoD is shown for each in-distribution. ProoD struggles to give non-trivial guarantees for the FPR@95% on most datasets. However, different from GOOD or ProoD-Disc, the clean performance is generally as good as that of OE. Models with accuracy drop of $> 3\%$ relative to the model with highest accuracy are grayed out. Of the remaining models, we highlight the best OOD detection performance.

In: CIFAR10		CIFAR100			SVHN			LSUN_CR			Smooth		
	Acc	FPR	GFPR	AFPR	FPR	GFPR	AFPR	FPR	GFPR	AFPR	FPR	GFPR	AFPR
Plain	**95.01**	56.3	100.0	100.0	40.7	100.0	100.0	46.7	100.0	100.0	10.6	100.0	100.0
OE	94.91	52.2	100.0	99.9	15.4	100.0	100.0	**0.0**	100.0	99.0	**0.0**	100.0	99.0
ATOM	93.63	73.4	100.0	**98.8**	33.9	100.0	100.0	85.3	100.0	100.0	**0.0**	100.0	**86.1**
ACET	93.43	65.4	100.0	99.5	**3.0**	100.0	**99.8**	62.7	100.0	100.0	**0.0**	100.0	89.0
GOOD_{80}	87.39	65.1	100.0	84.8	26.8	100.0	48.8	6.0	100.0	24.7	19.6	100.0	52.8
GOOD_{100}	86.96	84.8	100.0	99.3	87.9	100.0	99.7	66.0	100.0	99.0	68.6	100.0	98.2
ProoD-Disc	-	83.9	87.5	87.2	76.9	84.9	84.2	76.7	85.3	84.7	96.6	98.4	98.4
ProoD $\Delta = 3$	94.99	**48.0**	**99.9**	99.7	9.1	100.0	100.0	**0.0**	100.0	**98.7**	**0.0**	100.0	100.0
In: CIFAR100		**CIFAR10**			**SVHN**			**LSUN_CR**			**Smooth**		
	Acc	FPR	GFPR	AFPR	FPR	GFPR	AFPR	FPR	GFPR	AFPR	FPR	GFPR	AFPR
Plain	**77.38**	80.1	100.0	100.0	77.3	100.0	100.0	79.0	100.0	100.0	70.0	100.0	100.0
OE	77.25	81.8	100.0	100.0	37.7	100.0	100.0	0.0	100.0	99.7	**0.0**	100.0	100.0
ATOM	68.32	81.3	100.0	99.6	51.0	100.0	97.4	26.3	100.0	94.7	10.0	100.0	86.0
ACET	73.02	87.9	100.0	100.0	8.2	100.0	100.0	87.0	100.0	100.0	0.0	100.0	98.2
ProoD-Disc	-	95.9	97.5	97.5	91.3	92.8	92.7	91.7	95.3	95.0	100.0	100.0	100.0
ProoD $\Delta = 5$	77.16	**82.2**	100.0	100.0	**37.6**	100.0	100.0	**0.0**	100.0	**99.3**	3.4	100.0	100.0
In: R.ImgNet		**Flowers**			**FGVC**			**Cars**			**Smooth**		
	Acc	FPR	GFPR	AFPR	FPR	GFPR	AFPR	FPR	GFPR	AFPR	FPR	GFPR	AFPR
Plain	96.34	55.2	100.0	100.0	48.2	100.0	100.0	75.2	100.0	100.0	**0.0**	100.0	100.0
OE	97.10	**18.2**	100.0	100.0	**0.2**	100.0	100.0	**0.0**	100.0	100.0	**0.0**	100.0	100.0
ProoD-Disc	-	59.2	65.2	65.0	51.0	67.8	66.5	51.7	63.7	62.3	100.0	100.0	100.0
ProoD $\Delta = 4$	**97.25**	18.5	100.0	100.0	0.5	100.0	100.0	**0.0**	100.0	100.0	**0.0**	100.0	100.0

Perhaps surprisingly, ProoD's guarantees generalize remarkably well to the much larger radius on CIFAR10. The same holds for GOOD, as was already observed in [8]. ATOM and ACET are not robust at this radius, despite having lower accuracy, lower clean OOD performance and more expensive training. On the other hand ProoD's guarantees on CIFAR100 are quite weak at this radius, but given ATOM's and ACET's low AAUCs here and the fact that GOOD cannot be trained at all on CIFAR100, the results are not worse than for the competitors.

J Error Bars

In order to be mindful of our resource consumption we restrict the computation of error bars to our experiments on CIFAR10. Additionally, because the dataloader was much faster we ran these experiments using 80M Tiny Images as an out-distribution as opposed to OpenImages. We reran our experiments using the same hyperparameters 5 times. We computed the mean and the standard deviations for our models for all metrics shown in Table 5. The results are shown in Table 10. We see that the fluctuations across different runs are indeed rather small. Furthermore, the clean performance of OE and ProoD show no significant discrepancies.

Table 7: **Additional datasets:** We show the AUC, AAUC and GAUC for all models on uniform noise and on the test set of the train out-distribution. Models with accuracy drop of $> 3\%$ relative to the model with highest accuracy are grayed out. Of the remaining models, we highlight the best OOD detection performance.

In: CIFAR10		LSUN			LSUN_resize			Places365			iSUN		
	Acc	AUC	GAUC	AAUC	AUC	GAUC	AAUC	AUC	GAUC	AAUC	AUC	GAUC	AAUC
Plain	**95.01**	95.9	0.0	1.3	95.0	0.0	8.7	89.5	0.0	0.3	94.5	0.0	10.4
OE	94.91	98.7	0.0	0.9	97.4	0.0	6.1	**99.9**	0.0	2.5	97.6	0.0	9.4
ATOM	93.63	77.3	0.0	12.1	**100.0**	0.0	**98.4**	82.6	0.0	22.0	**100.0**	0.0	**98.8**
ACET	93.43	89.2	0.0	2.5	**100.0**	0.0	91.3	88.0	0.0	3.4	**100.0**	0.0	92.1
GOOD80	87.39	96.5	68.4	89.7	87.4	61.7	69.5	96.8	58.8	90.2	87.1	58.9	70.3
GOOD100	86.96	95.0	86.0	86.7	81.1	67.6	67.9	74.4	59.7	60.9	77.5	63.3	65.2
ProoD-Disc	-	95.8	94.1	94.2	76.4	70.3	71.5	76.6	71.1	71.5	74.9	69.0	70.3
ProoD $\Delta = 3$	94.99	**99.2**	**82.2**	**82.3**	97.1	**57.7**	59.2	**99.9**	**58.7**	**59.6**	97.1	**56.4**	58.1

In: CIFAR10		Uniform			Textures			80M Tiny Images			OpenImages		
	Acc	AUC	GAUC	AAUC	AUC	GAUC	AAUC	AUC	GAUC	AAUC	AUC	GAUC	AAUC
Plain	**95.01**	97.9	0.0	83.1	92.0	0.0	8.6	91.1	0.0	0.8	84.0	0.0	0.4
OE	94.91	99.9	0.0	98.2	**99.9**	0.0	13.8	93.6	0.0	0.6	99.7	0.0	2.5
ATOM	93.63	**100.0**	0.0	**100.0**	97.5	0.0	**74.7**	82.8	0.0	26.7	73.4	0.0	22.3
ACET	93.43	**100.0**	0.0	99.9	98.0	0.0	41.6	89.0	0.0	7.8	81.7	0.0	5.6
GOOD80	87.39	99.1	83.9	98.1	96.1	54.7	89.9	84.7	50.5	66.8	93.6	53.7	85.6
GOOD100	86.96	94.6	86.1	89.3	71.0	49.9	57.1	74.5	56.7	58.2	69.5	54.2	55.6
ProoD-Disc	-	53.8	46.7	46.7	75.3	69.9	70.5	75.7	69.8	70.9	64.8	59.0	59.6
ProoD $\Delta = 3$	94.99	99.9	**35.0**	94.7	**99.9**	**57.6**	63.1	**93.8**	**57.5**	**59.0**	**99.8**	**47.7**	**49.7**

In: CIFAR100		LSUN			LSUN_resize			Places365			iSUN		
	Acc	AUC	GAUC	AAUC	AUC	GAUC	AAUC	AUC	GAUC	AAUC	AUC	GAUC	AAUC
Plain	**77.38**	84.5	0.0	0.9	78.9	0.0	3.9	75.7	0.0	0.6	79.4	0.0	5.6
OE	77.25	94.5	0.0	1.3	**88.2**	0.0	2.1	**99.7**	0.0	1.9	**88.3**	0.0	3.1
ATOM	68.32	83.4	0.0	66.7	92.1	0.0	71.3	87.0	0.0	62.1	91.7	0.0	73.7
ACET	73.02	81.3	0.0	4.3	100.0	0.0	77.0	74.8	0.0	4.3	100.0	0.0	78.2
ProoD-Disc	-	81.4	78.8	79.3	70.1	66.7	67.1	71.7	68.2	68.5	69.3	65.6	66.0
ProoD $\Delta = 5$	77.16	**94.8**	**28.5**	**28.7**	86.3	**22.6**	**22.8**	**99.7**	**23.6**	**23.8**	87.2	**22.1**	**22.4**

In: CIFAR100		Uniform			Textures			80M Tiny Images			OpenImages		
	Acc	AUC	GAUC	AAUC	AUC	GAUC	AAUC	AUC	GAUC	AAUC	AUC	GAUC	AAUC
Plain	**77.38**	82.7	0.0	53.0	77.5	0.0	5.3	79.7	0.0	1.2	75.5	0.0	0.7
OE	77.25	99.3	0.0	**91.4**	99.0	0.0	7.5	80.2	0.0	1.5	99.6	0.0	1.0
ATOM	68.32	100.0	0.0	100.0	90.0	0.0	66.6	90.1	0.0	70.0	83.7	0.0	58.9
ACET	73.02	100.0	0.0	99.9	91.5	0.0	21.0	77.0	0.0	6.8	74.5	0.0	3.0
ProoD-Disc	-	40.9	37.0	37.4	53.6	50.2	50.3	59.7	56.7	56.9	58.3	54.6	54.8
ProoD $\Delta = 5$	77.16	**99.7**	**12.1**	87.0	**99.1**	**16.8**	**22.6**	**80.5**	**19.5**	**19.7**	**99.7**	**18.6**	**20.1**

K Combining ProoD with a Robust Classifier

In this work we have provided guarantees on adversarially robust out-of-distribution detection that do not come at the cost of accuracy. This is the main reason why we did not consider adversarial robustness on the in-distribution since this is known to come at the cost of clean accuracy [53]. However, it is an interesting question if ProoD can nonetheless be applied to models that are also adversarially robust on the in-distribution. In order to illustrate that it in fact can, we combine an adversarially robust Resnet-18 [18] from RobustBench [12] with a robust accuracy of 58.5% at l_∞ $\epsilon = 8/255$ on CIFAR10 with our binary discriminator as described in App. B. We use the same bias shift of $\Delta = 3$ as for our ProoD model on CIFAR10. We call the robust model "Robust" and the combined model "Robust-ProoD". Note that because no retraining is necessary, both clean accuracy and robust accuracy of both models are guaranteed to stay the same (ProoD does not change the

Table 8: **Additional datasets:** For RImgNet, we show the AUC, AAUC and GAUC for all models on uniform noise and on the test set of the train out-distribution, i.e. NotRImgNet.

In: R.ImgNet		Uniform			NotR.ImgNet		
	Acc	AUC	GAUC	AAUC	AUC	GAUC	AAUC
Plain	96.34	99.3	0.0	74.9	91.7	0.0	0.2
OE	97.10	99.6	0.0	84.6	**98.7**	0.0	1.2
ProoD-Disc	-	99.7	99.2	99.3	73.6	69.9	69.9
ProoD $\Delta = 4$	**97.25**	**99.8**	**79.7**	**95.2**	98.6	**50.1**	**51.3**

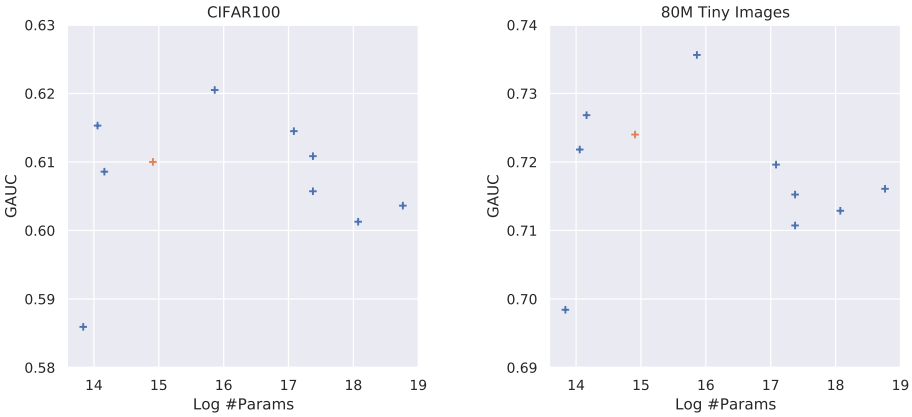

Figure 4: **Bigger models do not yield better guarantees:** We show scatter plots of the GAUC of different architectures against the log of the number of trainable parameters in the model. The orange cross indicates the architecture that is used in the main paper. There is no clear dependence of performance on model size, so it is preferable to use fairly small models.

predictions). We evaluate the adversarial robustness on the out-distribution and report the results in Table 11.

We can see that while the robust model already has remarkably strong empirical robustness on the out-distribution, ProoD does not harm the OOD detection performance of the model - neither clean nor adversarial. In fact, in all but a single case ProoD strictly improves the results (except for a 0.1% drop in clean AUC on LSUN_CR). In addition to this, ProoD provides non-zero GAUCs as well as the guarantees on asymptotically low confidence. The fact that ProoD operates so well in a regime for which it was not designed highlights its versatility.

Table 9: **Generalization to Larger ϵ:** We evaluate all CIFAR models in Table 2 using an $\epsilon = \frac{8}{255}$, and thus an unseen threat model. The provable methods GOOD and ProoD generalize surprisingly well, while neither ATOM nor ACET display any generalization to the larger threat model. 1

In: CIFAR10		CIFAR100			SVHN			LSUN_CR			Smooth		
	Acc	AUC	GAUC	AAUC	AUC	GAUC	AAUC	AUC	GAUC	AAUC	AUC	GAUC	AAUC
Plain	**95.01**	90.0	0.0	0.0	93.8	0.0	0.0	93.1	0.0	0.0	98.0	0.0	0.0
OE	94.91	**91.1**	0.0	0.1	97.3	0.0	0.0	**100.0**	0.0	0.1	**99.9**	0.0	0.0
ATOM	93.63	78.3	0.0	1.3	94.4	0.0	1.5	79.8	0.0	0.2	99.5	0.0	9.6
ACET	93.43	86.0	0.0	1.1	**99.3**	0.0	1.1	89.2	0.0	0.8	**99.9**	0.0	3.8
GOOD80*	87.39	76.7	37.5	51.6	90.8	38.6	74.3	97.4	57.6	90.2	96.2	61.1	87.8
GOOD100*	86.96	67.8	39.4	43.5	62.6	29.0	30.9	84.9	67.6	70.7	87.0	63.3	69.2
ProoD-Disc	-	62.9	44.1	46.1	72.6	52.5	57.1	78.1	56.3	58.9	59.2	34.9	37.2
ProoD $\Delta = 3$	94.99	89.8	**39.2**	**41.0**	98.3	**46.9**	**50.8**	**100.0**	**50.2**	**52.7**	**99.9**	**30.4**	**30.6**
In: CIFAR100		CIFAR10			SVHN			LSUN_CR			Smooth		
	Acc	AUC	GAUC	AAUC	AUC	GAUC	AAUC	AUC	GAUC	AAUC	AUC	GAUC	AAUC
Plain	**77.38**	**77.7**	0.0	0.4	81.9	0.0	0.2	76.4	0.0	0.3	86.6	0.0	0.3
OE	77.25	77.4	0.0	0.2	**92.3**	0.0	0.0	**100.0**	0.0	0.7	**99.5**	0.0	0.5
ATOM	68.32	78.3	0.0	10.4	91.1	0.0	15.2	95.9	0.0	23.0	98.2	0.0	23.5
ACET	73.02	73.0	0.0	1.4	97.8	0.0	0.7	75.8	0.0	2.6	99.9	0.0	3.8
ProoD-Disc	-	56.1	41.1	43.1	61.0	50.5	51.8	70.4	57.5	58.8	29.6	20.9	20.8
ProoD $\Delta = 5$	76.51	76.6	**13.7**	**14.1**	91.5	**16.9**	**16.9**	**100.0**	**18.1**	**18.2**	98.9	**8.1**	**8.1**
In: R.ImgNet		Flowers			FGVC			Cars			Smooth		
	Acc	AUC	GAUC	AAUC	AUC	GAUC	AAUC	AUC	GAUC	AAUC	AUC	GAUC	AAUC
Plain	96.34	92.3	0.0	0.0	92.6	0.0	0.0	92.7	0.0	0.0	**98.9**	0.0	0.0
OE	97.10	**96.9**	0.0	0.2	99.7	0.0	0.0	**99.9**	0.0	0.0	98.0	0.0	0.0
ProoD-Disc	-	81.5	60.4	61.4	92.8	78.0	80.8	90.7	76.3	79.2	81.0	47.3	53.7
ProoD $\Delta = 4$	**97.25**	**96.9**	**42.8**	**45.0**	**99.8**	**57.0**	**59.4**	**99.9**	**56.0**	**58.7**	98.6	**31.6**	**36.3**

*Uses different architecture of classifier, see "Baselines" in Section 4.2.

Table 10: **Error bars:** We show the mean and standard deviation σ of all metrics for our CIFAR10 models across 5 runs. The tolerances for ProoD's clean performance are very small and yet the differences in clean performance between OE ProoD are not significant.

In: CIFAR10		CIFAR100			SVHN			LSUN_CR			Smooth		
	Acc	AUC	GAUC	AAUC	AUC	GAUC	AAUC	AUC	GAUC	AAUC	AUC	GAUC	AAUC
Plain	94.91	90.0	0.0	0.6	93.9	0.0	0.1	93.4	0.0	0.7	96.7	0.0	1.2
Plain σ	0.16	0.1	0.0	0.1	1.2	0.0	0.0	0.3	0.0	0.2	2.1	0.0	0.5
OE	95.56	96.1	0.0	7.6	99.4	0.0	0.4	99.6	0.0	16.7	99.6	0.0	4.3
OE σ	0.04	0.1	0.0	1.5	0.1	0.0	0.2	0.1	0.0	3.5	0.3	0.0	3.7
ProoD-Disc	-	67.7	61.6	62.2	75.5	68.6	69.3	76.5	70.4	70.9	87.2	77.7	78.8
ProoD-Disc σ	-	0.7	0.7	0.7	1.4	1.7	1.5	1.4	1.7	1.7	3.6	4.3	4.3
ProoD $\Delta = 3$	95.60	96.0	42.2	44.1	99.4	48.6	49.2	99.6	47.1	52.0	99.8	55.2	57.0
ProoD $\Delta = 3$ σ	0.11	0.1	0.8	0.8	0.1	0.6	0.6	0.1	1.5	1.9	0.1	2.9	3.4

Table 11: **Robust models:** We report the OOD detection performance (AUCs, AAUCs and GAUCs) of a model that is adversarially robust on the in-distribution for different test out-distributions. The radius of the l_∞-ball for the adversarial manipulations of the OOD data is $\epsilon = 0.01$ for all datasets.

In: CIFAR10		CIFAR100			SVHN			LSUN_CR			Smooth		
	Acc	AUC	GAUC	AAUC	AUC	GAUC	AAUC	AUC	GAUC	AAUC	AUC	GAUC	AAUC
Robust	87.35	82.7	0.0	75.3	90.0	0.0	84.0	89.7	0.0	82.9	**94.5**	0.0	85.7
Robust-ProoD	87.35	**82.9**	**14.3**	**75.6**	**90.4**	**17.9**	**84.5**	**90.1**	**18.8**	**83.4**	94.4	**10.3**	**85.8**