Reverse KL-Divergence Training of Prior Networks: Improved Uncertainty and Adversarial Robustness

Andrey Malinin *
Yandex Research
am969@yandex-team.ru

Mark Gales
Department of Engineering
University of Cambridge
mjfg@eng.cam.ac.uk

Abstract

Ensemble approaches for uncertainty estimation have recently been applied to the tasks of misclassification detection, out-of-distribution input detection and adversarial attack detection. Prior Networks have been proposed as an approach to efficiently *emulate* an ensemble of models for classification by parameterising a Dirichlet prior distribution over output distributions. These models have been shown to outperform alternative ensemble approaches, such as Monte-Carlo Dropout, on the task of out-of-distribution input detection. However, scaling Prior Networks to complex datasets with many classes is difficult using the training criteria originally proposed. This paper makes two contributions. First, we show that the appropriate training criterion for Prior Networks is the *reverse* KL-divergence between Dirichlet distributions. This addresses issues in the nature of the training data target distributions, enabling prior networks to be successfully trained on classification tasks with arbitrarily many classes, as well as improving out-of-distribution detection performance. Second, taking advantage of this new training criterion, this paper investigates using Prior Networks to detect adversarial attacks and proposes a generalized form of adversarial training. It is shown that the construction of successful *adaptive* whitebox attacks, which affect the prediction and evade detection, against Prior Networks trained on CIFAR-10 and CIFAR-100 using the proposed approach requires a greater amount of computational effort than against networks defended using standard adversarial training or MC-dropout.

1 Introduction

Neural Networks (NNs) have become the dominant approach to addressing computer vision (CV) [1, 2, 3], natural language processing (NLP) [4, 5, 6], speech recognition (ASR) [7, 8] and bioinformatics [9, 10] tasks. One important challenge is for NNs to make reliable estimates of confidence in their predictions. Notable progress has recently been made on predictive uncertainty estimation for Deep Learning through the definition of baselines, tasks and metrics [11], and the development of practical methods for estimating uncertainty using ensemble methods, such as Monte-Carlo Dropout [12] and Deep Ensembles [13]. Uncertainty estimates derived from ensemble approaches have been successfully applied to the tasks of detecting misclassifications and out-of-distribution inputs, and have also been investigated for adversarial attack detection [14, 15]. However, ensembles can be computationally expensive and it is hard to control their behaviour. Recently, [16] proposed *Prior Networks* - a new approach to modelling uncertainty which has been shown to outperform Monte-Carlo dropout on a range of tasks. Prior Networks parameterize a Dirichlet prior over output distributions, which allows them to *emulate* an ensemble of models using a single network, whose behaviour can be *explicitly* controlled via choice of training data.

*Work done while at Cambridge University Department of Engineering

33rd Conference on Neural Information Processing Systems (NeurIPS 2019), Vancouver, Canada.

In [16], Prior Networks are trained using the *forward* KL-divergence between the model and a target Dirichlet distribution. It is, however, necessary to use auxiliary losses, such as cross-entropy, to yield competitive classification performance. Furthermore, it is also difficult to train Prior Networks using this criterion on complex datasets with many classes. In this work we show that the *forward* KL-divergence (KL) is an inappropriate optimization criterion and instead propose to train Prior Networks with the *reverse* KL-divergence (RKL) between the model and a target Dirichlet distribution. In sections 3 and 4 of this paper it is shown, both theoretically and empirically on synthetic data, that this loss yields the desired behaviours of a Prior Network and does not require auxiliary losses. In section 5 Prior Networks are successfully trained on a range of image classification tasks using the proposed criterion without loss of classification performance. It is also shown that these models yield better out-of-distribution detection performance on the CIFAR-10 and CIFAR-100 datasets than Prior Networks trained using *forward* KL-divergence.

An interesting application of uncertainty estimation is the detection of *adversarial attacks*, which are small perturbations to the input that are almost imperceptible to humans, yet which drastically affect the predictions of the neural network [17]. Adversarial attacks are a serious security concern, as there exists a plethora of adversarial attacks which are quite easy to construct [18, 19, 20, 21, 22, 23, 24, 25]. At the same time, while it is possible to improve the robustness of a network to adversarial attacks using adversarial training [17] and adversarial distillation [26], it is still possible to craft successful adversarial attacks against these networks [21]. Instead of considering *robustness* to adversarial attacks, [14] investigates *detection* of adversarial attacks and shows that adversarial attacks can be detectable using a range of approaches. While, *adaptive* attacks can be crafted to successfully attack the proposed detection schemes, [14] singles out detection of adversarial attacks using uncertainty measures derived from Monte-Carlo dropout as being more challenging to successfully overcome using adaptive attacks. Thus, in this work we investigate the detection of adversarial attacks using Prior Networks, which have previously outperformed Monte-Carlo dropout on other tasks.

Using the greater degree of control over the behaviour of Prior Networks which the *reverse* KL-divergence loss affords, Prior Networks are trained to predict the correct class on adversarial inputs, but yield a higher measure of uncertainty than on natural inputs. Effectively, this is a direct generalization of adversarial training [17] which improves *both* the robustness of the model to adversarial attacks *and* also allows them to be detected. As Prior Networks yield measures of uncertainty derived from distributions over output distributions, rather than simple confidences, adversarial attacks need to satisfy far more constraints in order to both successfully attack the Prior Network and evade detection. Results in section 6 show that on the CIFAR-10 and CIFAR-100 datasets it is more computationally challenging to construct *adaptive* adversarial attacks against Prior Networks than against standard DNNs, adversarially trained DNNs and Monte-Carlo dropout defended networks.

Thus, the two main contributions of this paper are the following. Firstly, a new *reverse* KL-divergence training criterion which yields the desired behaviour of Prior Networks and allows them to be trained on more complex datasets. Secondly, a generalized form of adversarial training, enabled using the proposed training criterion, which makes successful *adaptive* whitebox attacks, which aim to both attack the network and evade detection, far more computationally expensive to construct for Prior Networks than for models defended using standard adversarial training or Monte-Carlo dropout.

2 Prior Networks

An ensemble of models can be interpreted as a set of output distributions drawn from an *implicit* conditional distribution over output distributions. A Prior Network $\mathrm{p}(\boldsymbol{\pi}|\boldsymbol{x}^*;\hat{\boldsymbol{\theta}})$ [2], is a neural network which *explicitly* parametrizes a prior distribution over output distributions. This effectively allows a Prior Network to *emulate* an ensemble and yield the same measures of uncertainty [27, 28], but in closed form and without sampling.

$$\mathrm{p}(\boldsymbol{\pi}|\boldsymbol{x}^*;\hat{\boldsymbol{\theta}}) = \mathrm{p}(\boldsymbol{\pi}|\hat{\boldsymbol{\alpha}}), \quad \hat{\boldsymbol{\alpha}} = \boldsymbol{f}(\boldsymbol{x}^*;\hat{\boldsymbol{\theta}}) \tag{1}$$

A Prior Network typically parameterizes the Dirichlet distribution[3] (eq. 2), which is the conjugate prior to the categorical, due to its tractable analytic properties. The Dirichlet distribution is defined

[2] Here $\boldsymbol{\pi} = [\mathrm{P}(y=\omega_1), \cdots, \mathrm{P}(y=\omega_K)]^{\mathrm{T}}$ - the parameters of a categorical distribution.

[3] Alternate choices of distribution, such as a mixture of Dirichlets or the Logistic-Normal, are possible.

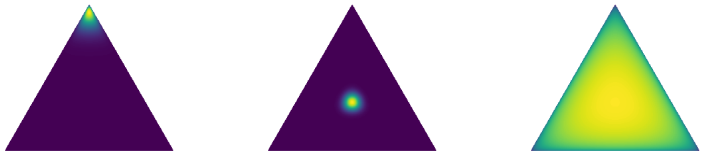
as:

$$\mathrm{p}(\boldsymbol{\pi};\boldsymbol{\alpha}) = \frac{\Gamma(\alpha_0)}{\prod_{c=1}^{K}\Gamma(\alpha_c)}\prod_{c=1}^{K}\pi_c^{\alpha_c-1}, \quad \alpha_c > 0,\ \alpha_0 = \sum_{c=1}^{K}\alpha_c \tag{2}$$

where $\Gamma(\cdot)$ is the gamma function. The Dirichlet distribution is parameterized by its concentration parameters $\boldsymbol{\alpha}$, where α_0, the sum of all α_c, is called the *precision* of the Dirichlet distribution. Higher values of α_0 lead to sharper, more confident distributions. The predictive distribution of a Prior Network is given by the expected categorical distribution under the conditional Dirichlet prior:

$$\mathrm{P}(y=\omega_c|\boldsymbol{x}^*;\hat{\boldsymbol{\theta}}) = \mathbb{E}_{\mathrm{p}(\boldsymbol{\pi}|\boldsymbol{x}^*;\hat{\boldsymbol{\theta}})}[\mathrm{P}(y=\omega_c|\boldsymbol{\pi})] = \hat{\pi}_c = \frac{\hat{\alpha}_c}{\sum_{k=1}^{K}\alpha_k} = \frac{e^{\hat{z}_c}}{\sum_{k=1}^{K}\hat{e}^{\hat{z}_k}} \tag{3}$$

where $\hat{z}_c$ are the logits predicted by the model. The desired behaviors of a Prior Network, as described in [16], can be visualized on a simplex in figure 1. Here, figure 1:a describes confident behavior (low-entropy prior focused on low-entropy output distributions), figure 1:b describes uncertainty due to severe class overlap (*data uncertainty*) and figure 1:c describes the behaviour for an out-of-distribution input (*knowledge uncertainty*).



(a) Low uncertainty (b) High data uncertainty (c) Out-of-distribution

Figure 1: Desired Behaviors of a Dirichlet distribution over categorical distributions.

Given a Prior Network which yields the desired behaviours, it is possible to derive measures of uncertainty in the prediction by considering the mutual information between y and $\boldsymbol{\pi}$:

$$\underbrace{\mathcal{MI}[y,\boldsymbol{\pi}|\boldsymbol{x}^*;\hat{\boldsymbol{\theta}}]}_{Knowledge\ Uncertainty} = \underbrace{\mathcal{H}\big[\mathbb{E}_{\mathrm{p}(\boldsymbol{\pi}|\boldsymbol{x}^*;\hat{\boldsymbol{\theta}})}[\mathrm{P}(y|\boldsymbol{\pi})]\big]}_{Total\ Uncertainty} - \underbrace{\mathbb{E}_{\mathrm{p}(\boldsymbol{\pi}|\boldsymbol{x}^*;\hat{\boldsymbol{\theta}})}\big[\mathcal{H}[\mathrm{P}(y|\boldsymbol{\pi})]\big]}_{Expected\ Data\ Uncertainty} \tag{4}$$

The given expression allows *total uncertainty*, given by the entropy of the predictive distribution, to be decomposed into *data uncertainty* and *knowledge uncertainty*. *Data uncertainty* arises due to class-overlap in the data, which is the equivalent of noise for classification problems. *Knowledge Uncertainty*, also know as *epistemic uncertainty* [12] or *distributional uncertainty* [16], arises due to the model's lack of understanding or *knowledge* about the input. In other word, *knowledge uncertainty* arises due to a mismatch between the training and test data.

3 Forward and Reverse KL-Divergence Losses

As Prior Networks parameterize the Dirichlet distribution, ideally we would like to have a dataset $\mathcal{D}_{trn} = \{\boldsymbol{x}^{(i)}, \boldsymbol{\beta}^{(i)}\}_{i=1}^{N}$, where $\boldsymbol{\beta}^{(i)}$ are the parameters of a target Dirichlet distribution $\mathrm{p}(\boldsymbol{\pi}|\boldsymbol{\beta})$. In this scenario, we could simply minimize the (forward) KL-divergence between the model and the target for every training sample $\boldsymbol{x}^{(i)}$. Alternatively, if we had a set of *samples* of categorical distributions from the target Dirichlet distribution for *every input*, then we could maximize the likelihood their under the predicted Dirichlet [29], which, in expectation, is equivalent to minimizing the KL-divergence. In practice, however, we only have access to the target class label $y^{(i)} \in \{\omega_1, \cdots, \omega_K\}$ for every input $\boldsymbol{x}^{(i)}$. When training standard DNNs with cross-entropy loss this isn't

a problem, as the correct target distribution $\hat{\mathtt{P}}_{\mathtt{tr}}(y|\boldsymbol{x})$ is *induced in expectation*, as shown below:

$$\begin{aligned}\mathcal{L}(\boldsymbol{\theta}, \mathcal{D}_{trn}) &= \mathbb{E}_{\hat{\mathtt{p}}_{\mathtt{tr}}(\boldsymbol{x})}\big[-\sum_{c=1}^{K}\mathbb{E}_{\hat{\mathtt{P}}_{\mathtt{tr}}(y|\boldsymbol{x})}[\mathcal{I}(y=\omega_c)]\ln \mathtt{P}(\hat{y}=\omega_c|\boldsymbol{x};\boldsymbol{\theta})\big] \\ &= \mathbb{E}_{\hat{\mathtt{p}}_{\mathtt{tr}}(\boldsymbol{x})}\Big[-\sum_{c=1}^{K}\hat{\mathtt{P}}_{\mathtt{tr}}(y=\omega_c|\boldsymbol{x})\ln \mathtt{P}(\hat{y}=\omega_c|\boldsymbol{x};\boldsymbol{\theta})\Big] \\ &= \mathbb{E}_{\hat{\mathtt{p}}_{\mathtt{tr}}(\boldsymbol{x})}\Big[\mathtt{KL}\big[\hat{\mathtt{P}}_{\mathtt{tr}}(y|\boldsymbol{x})||\mathtt{P}(y|\boldsymbol{x};\boldsymbol{\theta})\big]\Big] + const \\ &= \mathbb{E}_{\hat{\mathtt{p}}_{\mathtt{tr}}(\boldsymbol{x})}\Big[\mathtt{KL}\big[\boldsymbol{\pi}_{tr}||\hat{\boldsymbol{\pi}}\big]\Big] + const\end{aligned} \tag{5}$$

Unfortunately, training models which are a (higher-order) *distribution over predictive distributions* based on samples from the (lower-order) predictive distribution is more challenging. The solution to this problem proposed in the original work on Prior Networks [16] was to minimize the *forward* KL-divergence between the model and a target Dirichlet distribution $\mathtt{p}(\boldsymbol{\pi}|\boldsymbol{\beta}^{(c)})$:

$$\mathcal{L}^{KL}(y, \boldsymbol{x}, \boldsymbol{\theta}; \beta) = \sum_{c=1}^{K}\mathcal{I}(y=\omega_c)\cdot \mathtt{KL}[\mathtt{p}(\boldsymbol{\pi}|\boldsymbol{\beta}^{(c)})||\mathtt{p}(\boldsymbol{\pi}|\boldsymbol{x};\boldsymbol{\theta})] \tag{6}$$

The target concentration parameters $\boldsymbol{\beta}^{(c)}$ depend on the class c and are set manually as follows:

$$\beta_k^{(c)} = \begin{cases} \beta+1 & if\ c=k \\ 1 & if\ c\neq k \end{cases} \tag{7}$$

where β is a *hyper-parameter* which is set by hand, rather than learned from the data. This criterion is jointly optimized on in-domain and out-of-domain data $\mathcal{D}_{trn}$ and $\mathcal{D}_{out}$ as follows:

$$\mathcal{L}(\boldsymbol{\theta}, \mathcal{D}; \beta_{in}, \beta_{out}, \gamma) = \mathcal{L}_{in}^{KL}(\boldsymbol{\theta}, \mathcal{D}_{trn}; \beta_{in}) + \gamma\cdot\mathcal{L}_{out}^{KL}(\boldsymbol{\theta}, \mathcal{D}_{out}; \beta_{out}) \tag{8}$$

where γ is the out-of-distribution loss weight. In-domain β_{in} should take on a large value, for example $1e2$, so that the concentration is high only in the corner corresponding to the target class, and low elsewhere. Note, the concentration parameters have to be strictly positive, so it is not possible to set the rest of the concentration parameters to 0. Instead, they are set to one, which also provides a small degree smoothing. Out-of-domain $\beta_{out}=0$, which results in a flat Dirichlet distribution.

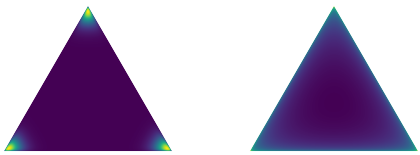
However, there is a significant issue with this criterion. Consider taking the expectation of equation 6 with respect to the empirical distribution $\hat{\mathtt{p}}_{\mathtt{tr}}(\boldsymbol{x}, y) = \{\boldsymbol{x}^{(i)}, y^{(i)}\}_{i=1}^{N} = \mathcal{D}_{trn}$:

$$\begin{aligned}\mathcal{L}^{KL}(\boldsymbol{\theta}, \mathcal{D}_{trn}; \beta) &= \mathbb{E}_{\hat{\mathtt{p}}_{\mathtt{tr}}(\boldsymbol{x},y)}\Big[\sum_{c=1}^{K}\mathcal{I}(y=\omega_c)\cdot\mathtt{KL}[\mathtt{p}(\boldsymbol{\pi}|\boldsymbol{\beta}^{(c)})||\mathtt{p}(\boldsymbol{\pi}|\boldsymbol{x};\boldsymbol{\theta})]\Big] \\ &= \mathbb{E}_{\hat{\mathtt{p}}_{\mathtt{tr}}(\boldsymbol{x})}\Big[\mathtt{KL}\big[\sum_{c=1}^{K}\hat{\mathtt{P}}_{\mathtt{tr}}(y=\omega_c|\boldsymbol{x})\mathtt{p}(\boldsymbol{\pi}|\boldsymbol{\beta}^{(c)})||\mathtt{p}(\boldsymbol{\pi}|\boldsymbol{x};\boldsymbol{\theta})\big]\Big] + \text{const}\end{aligned} \tag{9}$$

In expectation this loss induces a target distribution which is a *mixture of Dirichlet distributions* that has a mode in each corner of the simplex, as shown in figure 2:a. When the level of *data uncertainty* is low, this is not a problem, as there will only be a single significant mode. However, the target distribution will be multi-modal when there is a significant amount of *data uncertainty*. As the *forward* KL-divergence is *zero-avoiding*, it will drive the model to spread itself over each mode, effectively 'inverting' the Dirichlet distribution and forcing the precision $\hat{\alpha}_0$ to a low value, as depicted in figure 2:b. This is an undesirable behaviour and can compromise predictive performance. Rather, as previously stated, in regions of significant *data uncertainty* the model should yield a distribution with a single high-precision mode at the center of the simplex, as shown in figure 1:b.

The main issue with the *forward* KL-divergence loss is that it induces an *arithmetic mixture* of target distributions $\mathtt{p}(\boldsymbol{\pi}|\boldsymbol{\beta}^{(c)})$ in expectation. This can be avoided by instead considering the *reverse* KL-divergence between the target distribution $\mathtt{p}(\boldsymbol{\pi}|\boldsymbol{\beta}^{(c)})$ and the model:

$$\mathcal{L}^{RKL}(y, \boldsymbol{x}, \boldsymbol{\theta}; \beta) = \sum_{c=1}^{K}\mathcal{I}(y=\omega_c)\cdot\mathtt{KL}[\mathtt{p}(\boldsymbol{\pi}|\boldsymbol{x};\boldsymbol{\theta})||\mathtt{p}(\boldsymbol{\pi}|\boldsymbol{\beta}^{(c)})] \tag{10}$$



(a) Induced Target (b) Actual behaviour

Figure 2: Induced target and predicted Dirichlet distribution when trained with equation 6

The expectation of this criterion with respect to the empirical distribution induces a *geometric mixture* of target Dirichlet distributions:

$$\begin{aligned}
\mathcal{L}^{RKL}(\boldsymbol{\theta}, \mathcal{D}_{trn}; \beta) &= \mathbb{E}_{\hat{\mathtt{p}}_{\mathtt{tr}}(\boldsymbol{x})}\Big[\sum_{c=1}^{K} \hat{\mathtt{P}}_{\mathtt{tr}}(y=\omega_c|\boldsymbol{x})\mathtt{KL}\big[\mathtt{p}(\boldsymbol{\pi}|\boldsymbol{x};\boldsymbol{\theta})||\mathtt{p}(\boldsymbol{\pi}|\boldsymbol{\beta}^{(c)})\big]\Big] \\
&= \mathbb{E}_{\hat{\mathtt{p}}_{\mathtt{tr}}(\boldsymbol{x})}\Big[\mathbb{E}_{\mathtt{p}(\boldsymbol{\pi}|\boldsymbol{x};\boldsymbol{\theta})}\big[\ln \mathtt{p}(\boldsymbol{\pi}|\boldsymbol{x};\boldsymbol{\theta}) - \ln \prod_{c=1}^{K} \mathtt{p}(\boldsymbol{\pi}|\boldsymbol{\beta}^{(c)})^{\hat{\mathtt{P}}_{\mathtt{tr}}(y=\omega_c|\boldsymbol{x})}\big]\Big] \\
&= \mathbb{E}_{\hat{\mathtt{p}}_{\mathtt{tr}}(\boldsymbol{x})}\Big[\mathtt{KL}\big[\mathtt{p}(\boldsymbol{\pi}|\boldsymbol{x};\boldsymbol{\theta})||\mathtt{p}(\boldsymbol{\pi}|\bar{\boldsymbol{\beta}})\big]\Big] + \text{const} \\
\bar{\boldsymbol{\beta}} &= \sum_{c=1}^{K} \hat{\mathtt{P}}_{\mathtt{tr}}(y=\omega_c|\boldsymbol{x}) \cdot \boldsymbol{\beta}^{(c)}
\end{aligned} \tag{11}$$

A geometric mixture of Dirichlet distributions results in a standard Dirichlet distribution whose concentration parameters $\bar{\boldsymbol{\beta}}$ are an arithmetic mixture of the target concentration parameters for each class. Thus, this loss induces a target distribution which is *always* a standard uni-modal Dirichlet with a mode at the point on the simplex which reflects the correct level of *data uncertainty* (figure 1a-b). Furthermore, as a consequence using of this loss in equation 8 instead of the *forward* KL-divergence, the concentration parameters are appropriately interpolated on the boundary of the in-domain and out-of-distribution regions, where the degree of interpolation depends on the OOD loss weight γ. Further analysis of the properties of the *reverse* KL-divergence loss is provided in appendix A.

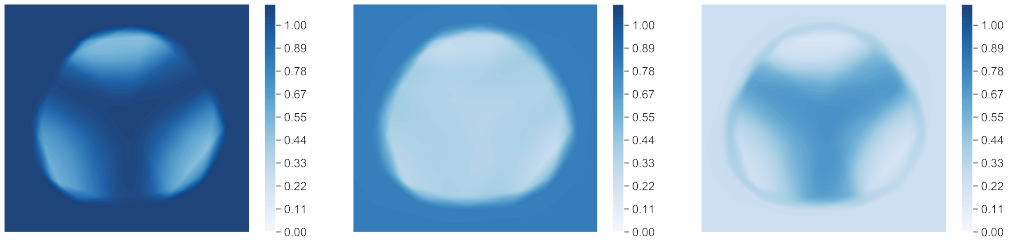
Finally, it is important to emphasize that this discussion is about *what target distribution is induced in expectation* when training models which parameterize a *distribution over output distributions* using samples from *the output distribution*. It is necessary to stress that if either the parameters of, or samples from, the correct target distribution over output distributions are available, for every input, then *forward KL-divergence* **is** a sensible training criterion.

4 Experiments on Synthetic Data

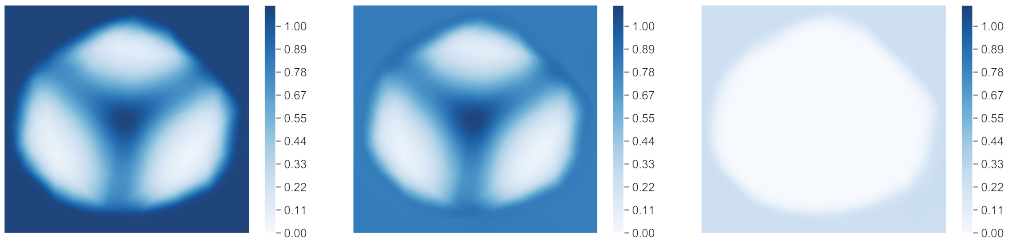
The previous section investigated the theoretical properties of forward and reverse KL-divergence training criteria for Prior Networks. In this section these criteria are assessed empirically by using them to train Prior Networks on the artificial high-uncertainty 3-class dataset[4] introduced in [16]. In these experiments, the out-of-distribution training data $\mathcal{D}_{out}$ was sampled such that it forms a thin shell around the training data. The target concentration parameters $\boldsymbol{\beta}^{(c)}$ were constructed as described in equation 7, with $\beta_{in} = 1e2$ and $\beta_{out} = 0$. The in-domain loss and out-of-distribution losses were equally weighted ($\gamma = 1$).

Figure 3 depicts the *total uncertainty*, *expected data uncertainty* and mutual information, which is a measure of *knowledge uncertainty*, derived using equation 4 from Prior Networks trained using both criteria. By comparing figures 3a and 3d it is clear that a Prior Network trained using *forward* KL-divergence over-estimates *total uncertainty* in domain, as the *total uncertainty* is *equally* high along the decision boundaries, in the region of class overlap and out-of-domain. The Prior Network trained using the *reverse* KL-divergence, on the other hand, yields an estimate of *total uncertainty* which better reflects the structure of the dataset. Figure 3b shows that the *expected data uncertainty* is altogether incorrectly estimated by the Prior Network trained via *forward* KL-divergence, as

[4]Described in appendix B.



(a) Total Uncertainty - KL (b) Data Uncertainty - KL (c) Mutual Information - KL



(d) Total Uncertainty - RKL (e) Data Uncertainty - RKL (f) Mutual Information - RKL

Figure 3: Comparison of measures of uncertainty derived from Prior Networks trained with *forward* and *reverse* KL-divergence. Measures of uncertainty are derived via equation 4.

it is uniform over the entire in-domain region. As a result, the mutual information is higher in-domain along the decision boundaries than out-of-domain. In contrast, figures 3c and 3f show that the measures of uncertainty provided by a Prior Network trained using the *reverse* KL-divergence decompose correctly - *data uncertainty* is highest in regions of class overlap while mutual information is low in-domain and high out-of-domain. Thus, these experiments support the analysis in the previous section, and illustrate how the *reverse* KL-divergence is the more suitable optimization criterion.

5 Image Classification Experiments

Having evaluated the *forward* and *reverse* KL-divergence losses on a synthetic dataset in the previous section, we now evaluate these losses on a range of image classification datasets. The training configurations are described in appendix C. Table 1 presents the classification error rates of standard DNNs, an ensemble of 5 DNNs [13], and Prior Networks trained using both the *forward* and *reverse* KL-divergence losses. From table 1 it is clear that Prior Networks trained using *forward* KL-divergence (PN-KL) achieve increasingly worse classification performance as the datasets become more complex and have a larger number of classes. At the same time, Prior Networks trained using the *reverse* KL-divergence loss (PN-RKL) have similar error rates as ensembles and standard DNNs. Note that in these experiments no auxiliary losses were used.[5]

Table 1: Mean classification performance (% Error) $\pm 2\sigma$ across 5 random initializations.

Dataset	DNN	PN-KL	PN-RKL	ENSM
MNIST	**0.5** ± 0.1	0.6 ± 0.1	**0.5** ± 0.1	**0.5** $\pm$ NA
SVHN	4.3 ± 0.3	5.7 ± 0.2	4.2 ± 0.2	**3.3** $\pm$ NA
CIFAR-10	8.0 ± 0.4	14.7 ± 0.4	7.5 ± 0.3	**6.6** $\pm$ NA
CIFAR-100	30.4 ± 0.6	-	28.1 ± 0.2	**26.9** $\pm$ NA
TinyImageNet	41.7 ± 0.4	-	40.3 ± 0.4	**36.9** $\pm$ NA

[5] An on-going PyTorch re-implementation of this paper, along updated results, is available at `https://github.com/KaosEngineer/PriorNetworks`

Table 2 presents the out-of-distribution detection performance of Prior Networks trained on CIFAR-10 and CIFAR-100 [30] using the *forward* and *reverse* KL-divergences. Prior Networks trained on CIFAR-10 use CIFAR-100 are OOD training data, while Prior Networks trained on CIFAR-100 use TinyImageNet [31] as OOD training data. Performance is assessed using area under an ROC curve (AUROC) in the same fashion as in [16, 11]. The results on CIFAR-10 show that PN-RKL consistently yields better performance than PN-KL and the ensemble on all OOD test datasets (SVHN, LSUN and TinyImagenet). The results using model trained on CIFAR-100 show that Prior Networks are capable of out-performing the ensembles when evaluated against the LSUN and SVHN datasets. However, Prior Networks have difficulty distinguishing between the CIFAR-10 and CIFAR-100 test sets. However, this represents a limitation of the both the classification model and the OOD training data, rather than the training criterion. Improving classification performance of Prior Networks on CIFAR-100, which improves understanding of what is 'in-domain', and using a more appropriate OOD training dataset, which provides a better contrast, is likely improve OOD detection performance.

Table 2: Out-of-domain detection results (mean % AUROC $\pm 2\sigma$ across 5 rand. inits) using mutual information (eqn. 4) derived from models trained on CIFAR-10 and CIFAR-100.

Model	CIFAR-10			CIFAR-100		
	SVHN	LSUN	TinyImageNet	SVHN	LSUN	CIFAR-10
ENSM	89.5 $\pm$ NA	93.2 $\pm$ NA	90.3 $\pm$ NA	78.9 $\pm$ NA	85.6 $\pm$ NA	**76.5** $\pm$ NA
PN-KL	97.8 $\pm$1.1.	91.6 $\pm$1.7	92.4 $\pm$0.9	-	-	-
PN-RKL	**98.2** $\pm$1.1	**95.7** $\pm$0.9	**95.7** $\pm$0.7	**84.8** $\pm$0.8	**100.0** $\pm$0.0	57.8 $\pm$0.4

6 Adversarial Attack Detection

The previous section has discussed the use of the reverse KL-divergence training criterion for training Prior Networks. Here, we show that the proposed loss also offers a generalization of adversarial training [17, 25] which allows Prior Networks to be *both* more robust to adversarial attacks *and* detect them as OOD samples. The use of measures of uncertainty for adversarial attack detection was previously studied in [14], where it was shown that Monte-Carlo dropout ensembles yield measures of uncertainty which are more challenging to attack than other considered methods. In a similar fashion to Monte-Carlo dropout, Prior Networks yield rich measures of uncertainty derived from distributions over distributions. For Prior Networks this means that for an adversarial attack to both affect the prediction and evade detection, it must satisfy several criteria. Firstly, the adversarial input must be located in a region of input space classified as the desired class. Secondly, the adversarial input must be in a region of input space where both the *relative* and *absolute* magnitudes of the model's logits $\hat{\boldsymbol{z}}$, and therefore all the measures of uncertainty derivable from the predicted distribution over distribution, are the same as for the natural input, making it challenging to distinguish between the natural and adversarial input. Clearly, this places more constraints on the space of solutions for successful adversarial attacks than detection based on the confidence of the prediction, which places a constraint only on the *relative* value of just a single logit.

(a) Natural Target (b) Adversarial Target

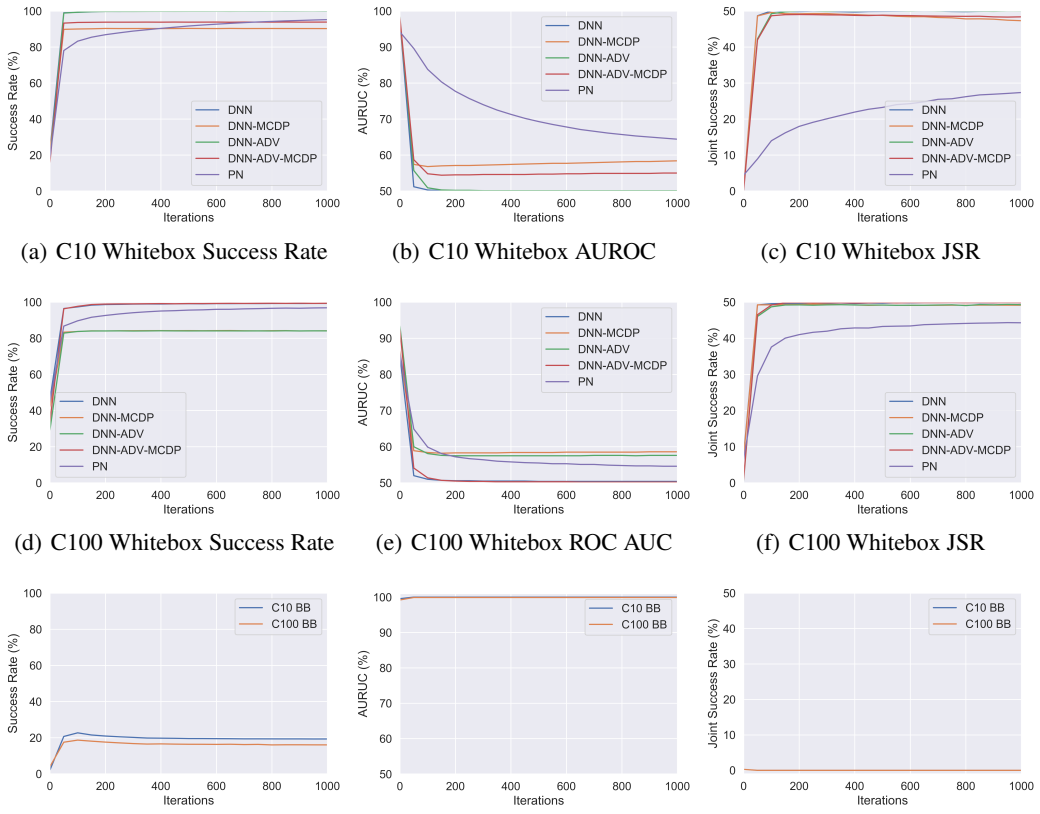
Figure 4: Target Dirichlet distributions for natural and adversarial inputs.

Using the greater degree of control over the behaviour of Prior Networks which the *reverse* KL-divergence loss affords, Prior Networks can be *explicitly* trained to yield high uncertainty for example adversarial attacks, further constraining the space of successful solutions. Here, adversarially perturbed inputs are used as the out-of-distribution training data for which the Prior Network is

trained to *both* yield the correct prediction *and* high measures of uncertainty. Thus, the Prior Network is jointly trained to yield either a *sharp* or *wide* Dirichlet distribution at the appropriate corner of the simplex for natural or adversarial data, respectively, as described in figure 4.

$$\mathcal{L}(\boldsymbol{\theta}, \mathcal{D}; \beta_{in}, \beta_{adv}, \gamma) = \mathcal{L}_{in}^{RKL}(\boldsymbol{\theta}, \mathcal{D}_{trn}; \beta_{in}) + \gamma \cdot \mathcal{L}_{adv}^{RKL}(\boldsymbol{\theta}, \mathcal{D}_{adv}; \beta_{adv}) \tag{12}$$

The target concentration parameters are set using equation 7, where $\beta_{in} = 1e2$ for natural and $\beta_{adv} = 1$ for adversarial data, for example. This approach can be seen as a generalization of *adversarial training* [17, 25]. The difference is that here we are training the model to yield a particular behaviour of an entire *distribution over output distributions*, rather than simply making sure that the decision boundaries are correct in regions of input space which correspond to adversarial attacks. Furthermore, it is important to highlight that this generalized form of adversarial training is a *drop-in replacement* for standard adversarial training which only requires changing the loss function.



(a) C10 Whitebox Success Rate (b) C10 Whitebox AUROC (c) C10 Whitebox JSR

(d) C100 Whitebox Success Rate (e) C100 Whitebox ROC AUC (f) C100 Whitebox JSR

(g) PN Blackbox Success Rate (h) PN Blackbox AUROC (i) PN Blacbox JSR

Figure 5: Adaptive Attack detection performance in terms of mean Success Rate, % AUROC and Joint Success Rate (JSR) across 5 random inits. L_∞ bound on adversarial perturbation is 30 pixels.

As discussed in [14, 32], approaches to detecting adversarial attacks need to be evaluated against the strongest possible attacks - *adaptive* whitebox attacks which have full knowledge of the detection scheme and actively seek to bypass it. Here, targeted iterative PGD-MIM [20, 25] attacks are used for evaluation and simple targeted FGSM [17] are used during training. The goal is to switch the prediction to a target class but leave measures of uncertainty derived from the model unchanged.

Two forms of criteria, expressed in equation 13, are used to generate the adversarial sample, $\tilde{\boldsymbol{x}}$. For both criteria the target for the attacks is set to the second most likely class, as this should yield the least 'unnatural' perturbation of the outputs. The first approach involves permuting the model's predictive distribution over class labels $\hat{\boldsymbol{\pi}}$ and minimizing the *forward* KL-divergence between $\hat{\boldsymbol{\pi}}$ and the target permuted distribution $\boldsymbol{\pi}_{adv}$. This ensures that the target class is predicted, but places constraints only the *relative* values of the logits, and therefore only on measures of uncertainty derived from the predictive posterior. The second approach involves permuting the concentration parameters $\hat{\boldsymbol{\alpha}}$ and

minimizing the *forward* KL divergence to the permuted target Dirichlet distribution $\mathtt{p}_{\mathtt{adv}}(\boldsymbol{\pi})$. This places constraints on both the *relative* and *absolute* values of the logits, and therefore on measures of uncertainty derived from the entire distribution over distributions.

$$\mathcal{L}^{KL}_{PMF}\big(\mathtt{P}(y|\tilde{\boldsymbol{x}};\hat{\boldsymbol{\theta}}),t\big) = \mathtt{KL}[\boldsymbol{\pi}_{adv}||\hat{\boldsymbol{\pi}}],\ \mathcal{L}^{KL}_{DIR}\big(\mathtt{p}(\boldsymbol{\pi}|\tilde{\boldsymbol{x}};\hat{\boldsymbol{\theta}}),t\big) = \mathtt{KL}[\mathtt{p}_{\mathtt{adv}}(\boldsymbol{\pi})||\mathtt{p}(\boldsymbol{\pi}|\tilde{\boldsymbol{x}};\hat{\boldsymbol{\theta}})] \tag{13}$$

Though $\mathcal{L}^{KL}_{DIR}$ has more explicit constraints, it was found to be more challenging to optimize and yield less aggressive attacks than $\mathcal{L}^{KL}_{PMF}$. [6] Thus, only attacks generated via $\mathcal{L}^{KL}_{PMF}$ are considered.

In the following set of experiments Prior Networks are trained on either the CIFAR-10 or CIFAR-100 datasets [30] using the procedure discussed above and detailed in appendixC. The baseline models are an undefended DNN and a DNN trained using standard adversarial training (DNN-ADV). For these models uncertainty is estimated via the entropy of the predictive posterior. Additionally, estimates of mutual information (*knowledge uncertainty*) are derived via a Monte-Carlo dropout ensemble generated from each of these models. Similarly, Prior Networks also use the mutual information (eqn. 4) for adversarial attack detection. Performance is assessed via the Success Rate, AUROC and Joint Success Rate (JSR). For the ROC curves considered here the true positive rate is computed using natural examples, while the false-positive rate is computed using only *successful* adversarial attacks[7]. The JSR, described in greater detail in appendix D, is the equal error rate where false positive rate equals false negative rate, and allows *joint* assessment of adversarial robustness and detection.

The results presented in figure 5 show that on both the CIFAR-10 and CIFAR-100 datasets whitebox attacks successfully change the prediction of DNN and DNN-ADV models to the second most likely class and evade detection (AUROC goes to 50). Monte-Carlo dropout ensembles are marginally harder to adversarially overcome, due to the random noise. At the same time, it takes far more iterations of gradient descent to successfully attack Prior Networks such that they fail to detect the attack. On CIFAR-10 the Joint Success Rate is only 0.25 at 1000 iterations, while the JSR for the other models is 0.5 (the maximum). Results on the more challenging CIFAR-100 dataset show that adversarially trained Prior Networks yield a more modest increase in robustness over baseline approaches, but it still takes significantly more computational effort to attack the model. Thus, these results support the assertion that adversarially trained Prior Networks constrain the solution space for adaptive adversarial attack, making them computationally more difficult to successfully construct. At the same time, *blackbox attacks*, computed on identical networks trained on the same data from a different random initialization, fail entirely against Prior Networks trained on CIFAR-10 and CIFAR-100. This shows that the adaptive attacks considered here are non-transferable.

7 Conclusion

Prior Networks have been shown to be an interesting approach to deriving rich and interpretable measures of uncertainty from neural networks. This work consists of two contributions which aim to improve these models. Firstly, a new training criterion for Prior Networks, the reverse KL-divergence between Dirichlet distributions, is proposed. It is shown, both theoretically and empirically, that this criterion yields the desired set of behaviours of a Prior Network and allows these models to be trained on more complex datasets with arbitrary numbers of classes. Furthermore, it is shown that this loss improves out-of-distribution detection performance on the CIFAR-10 and CIFAR-100 datasets relative to the *forward* KL-divergence loss used in [16]. However, it is necessary to investigate proper choice of out-of-distribution training data, as an inappropriate choice can limit OOD detection performance on complex datasets. Secondly, this improved training criterion enables Prior Networks to be applied to the task of detecting whitebox adaptive adversarial attacks. Specifically, adversarial training of Prior Networks can be seen as both a generalization of, and a drop in replacement for, standard adversarial training which improves robustness to adversarial attacks and the ability to detect them by placing more constraints on the space of solutions to the optimization problem which yields adversarial attacks. It is shown that it is significantly more computationally challenging to construct successfully adaptive whitebox PGD attacks against Prior Network than against baseline models. It is necessary to point out that the evaluation of adversarial attack detection using Prior Networks is limited to only strong L_∞ attacks. It is of interest to assess how well Prior Networks are able to detect adaptive C&W L_2 attacks [21] and EAD L_1 attacks [33].

[6] Results are described in appendix E.

[7] The may result in minimum AUROC performance being a little greater than 50 is the success rate is not 100 %, as is the case with MCDP AUROC in figure 5.

Acknowledgments

This paper reports on research partly supported by Cambridge Assessment, University of Cambridge. This work is also partly funded by a DTA EPSRC award.

References

[1] Ross Girshick, "Fast R-CNN," in *Proc. 2015 IEEE International Conference on Computer Vision (ICCV)*, 2015, pp. 1440–1448.

[2] Karen Simonyan and Andrew Zisserman, "Very Deep Convolutional Networks for Large-Scale Image Recognition," in *Proc. International Conference on Learning Representations (ICLR)*, 2015.

[3] Ruben Villegas, Jimei Yang, Yuliang Zou, Sungryull Sohn, Xunyu Lin, and Honglak Lee, "Learning to Generate Long-term Future via Hierarchical Prediction," in *Proc. International Conference on Machine Learning (ICML)*, 2017.

[4] Tomas Mikolov et al., "Linguistic Regularities in Continuous Space Word Representations," in *Proc. NAACL-HLT*, 2013.

[5] Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean, "Efficient Estimation of Word Representations in Vector Space," 2013, arXiv:1301.3781.

[6] Tomas Mikolov, Martin Karafiát, Lukás Burget, Jan Cernocký, and Sanjeev Khudanpur, "Recurrent Neural Network Based Language Model," in *Proc. INTERSPEECH*, 2010.

[7] Geoffrey Hinton, Li Deng, Dong Yu, George Dahl, Abdel rahman Mohamed, Navdeep Jaitly, Andrew Senior, Vincent Vanhoucke, Patrick Nguyen, Tara Sainath, and Brian Kingsbury, "Deep neural networks for acoustic modeling in speech recognition," *Signal Processing Magazine*, 2012.

[8] Awni Y. Hannun, Carl Case, Jared Casper, Bryan Catanzaro, Greg Diamos, Erich Elsen, Ryan Prenger, Sanjeev Satheesh, Shubho Sengupta, Adam Coates, and Andrew Y. Ng, "Deep speech: Scaling up end-to-end speech recognition," 2014, arXiv:1412.5567.

[9] Rich Caruana, Yin Lou, Johannes Gehrke, Paul Koch, Marc Sturm, and Noemie Elhadad, "Intelligible models for healthcare: Predicting pneumonia risk and hospital 30-day readmission," in *Proc. 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, New York, NY, USA, 2015, KDD '15, pp. 1721–1730, ACM.

[10] Babak Alipanahi, Andrew Delong, Matthew T. Weirauch, and Brendan J. Frey, "Predicting the sequence specificities of DNA- and RNA-binding proteins by deep learning," *Nature Biotechnology*, vol. 33, no. 8, pp. 831–838, July 2015.

[11] Dan Hendrycks and Kevin Gimpel, "A Baseline for Detecting Misclassified and Out-of-Distribution Examples in Neural Networks," `http://arxiv.org/abs/1610.02136`, 2016, arXiv:1610.02136.

[12] Yarin Gal and Zoubin Ghahramani, "Dropout as a Bayesian Approximation: Representing Model Uncertainty in Deep Learning," in *Proc. 33rd International Conference on Machine Learning (ICML-16)*, 2016.

[13] B. Lakshminarayanan, A. Pritzel, and C. Blundell, "Simple and Scalable Predictive Uncertainty Estimation using Deep Ensembles," in *Proc. Conference on Neural Information Processing Systems (NIPS)*, 2017.

[14] Nicholas Carlini and David A. Wagner, "Adversarial examples are not easily detected: Bypassing ten detection methods," *CoRR*, 2017.

[15] L. Smith and Y. Gal, "Understanding Measures of Uncertainty for Adversarial Example Detection," in *UAI*, 2018.

[16] Andrey Malinin and Mark Gales, "Predictive uncertainty estimation via prior networks," in *Advances in Neural Information Processing Systems*, 2018, pp. 7047–7058.

[17] Christian Szegedy, Alexander Toshev, and Dumitru Erhan, "Deep neural networks for object detection," in *Advances in Neural Information Processing Systems*, 2013.

[18] Ian Goodfellow, Jonathon Shlens, and Christian Szegedy, "Explaining and harnessing adversarial examples," in *International Conference on Learning Representations*, 2015.

[19] Alexey Kurakin, Ian J. Goodfellow, and Samy Bengio, "Adversarial examples in the physical world," 2016, vol. abs/1607.02533.

[20] Yinpeng Dong, Fangzhou Liao, Tianyu Pang, Hang Su, Jun Zhu, Xiaolin Hu, and Jianguo Li, "Boosting adversarial attacks with momentum," in *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018.

[21] Nicholas Carlini and David A. Wagner, "Towards evaluating the robustness of neural networks," *CoRR*, 2016.

[22] Nicolas Papernot, Patrick D. McDaniel, Ian J. Goodfellow, Somesh Jha, Z. Berkay Celik, and Ananthram Swami, "Practical black-box attacks against deep learning systems using adversarial examples," *CoRR*, vol. abs/1602.02697, 2016.

[23] Nicolas Papernot, Patrick D. McDaniel, Somesh Jha, Matt Fredrikson, Z. Berkay Celik, and Ananthram Swami, "The limitations of deep learning in adversarial settings," in *IEEE European Symposium on Security and Privacy, EuroS&P 2016, Saarbrücken, Germany, March 21-24, 2016*, 2016, pp. 372–387.

[24] Yanpei Liu, Xinyun Chen, Chang Liu, and Dawn Song, "Delving into transferable adversarial examples and black-box attacks," *CoRR*, vol. abs/1611.02770, 2016.

[25] Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu, "Towards deep learning models resistant to adversarial attacks," *arXiv preprint arXiv:1706.06083*, 2017.

[26] Nicolas Papernot, Patrick D. McDaniel, Xi Wu, Somesh Jha, and Ananthram Swami, "Distillation as a defense to adversarial perturbations against deep neural networks," in *IEEE Symposium on Security and Privacy, SP 2016, San Jose, CA, USA, May 22-26, 2016*, 2016, pp. 582–597.

[27] Yarin Gal, *Uncertainty in Deep Learning*, Ph.D. thesis, University of Cambridge, 2016.

[28] Stefan Depeweg, José Miguel Hernández-Lobato, Finale Doshi-Velez, and Steffen Udluft, "Decomposition of uncertainty for active learning and reliable reinforcement learning in stochastic systems," *stat*, vol. 1050, pp. 11, 2017.

[29] Anonymous, "Ensemble distribution distillation," in *Submitted to International Conference on Learning Representations*, 2020, under review.

[30] Alex Krizhevsky, "Learning multiple layers of features from tiny images," 2009.

[31] Stanford CS231N, "Tiny ImageNet," `https://tiny-imagenet.herokuapp.com/`, 2017.

[32] Nicholas Carlini, Anish Athalye, Nicolas Papernot, Wieland Brendel, Jonas Rauber, Dimitris Tsipras, Ian Goodfellow, and Aleksander Madry, "On evaluating adversarial robustness," *arXiv preprint arXiv:1902.06705*, 2019.

[33] Pin-Yu Chen, Yash Sharma, Huan Zhang, Jinfeng Yi, and Cho-Jui Hsieh, "Ead: elastic-net attacks to deep neural networks via adversarial examples," in *Thirty-second AAAI conference on artificial intelligence*, 2018.

[34] Murat Sensoy, Lance Kaplan, and Melih Kandemir, "Evidential deep learning to quantify classification uncertainty," in *Advances in Neural Information Processing Systems 31*, S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett, Eds., pp. 3179–3189. Curran Associates, Inc., 2018.

[35] Martín Abadi et al., "TensorFlow: Large-Scale Machine Learning on Heterogeneous Systems," 2015, Software available from tensorflow.org.

[36] Diederik P. Kingma and Jimmy Ba, "Adam: A Method for Stochastic Optimization," in *Proc. 3rd International Conference on Learning Representations (ICLR)*, 2015.

[37] Zhitao Gong, Wenlu Wang, and Wei-Shinn Ku, "Adversarial and clean data are not twins," *CoRR*, vol. abs/1704.04960, 2017.

[38] Kathrin Grosse, Praveen Manoharan, Nicolas Papernot, Michael Backes, and Patrick D. McDaniel, "On the (statistical) detection of adversarial examples," *CoRR*, vol. abs/1702.06280, 2017.

[39] Jan Hendrik Metzen, Tim Genewein, Volker Fischer, and Bastian Bischoff, "On detecting adversarial perturbations," in *Proceedings of 5th International Conference on Learning Representations (ICLR)*, 2017.

Appendix A Further Analysis of reverse KL-divergence Loss

It is interesting to further analyze the properties of the *reverse* KL-divergence loss by decomposing it into the reverse *cross-entropy* and the negative differential entropy:

$$\mathcal{L}^{RKL}(\boldsymbol{\theta};\beta) = \mathbb{E}_{\hat{\mathtt{p}}_{\mathtt{tr}}(\boldsymbol{x})}\Big[\underbrace{\mathbb{E}_{\mathtt{p}(\boldsymbol{\pi}|\boldsymbol{x};\boldsymbol{\theta})}\big[-\ln \mathtt{Dir}(\boldsymbol{\pi}|\bar{\boldsymbol{\beta}})\big]}_{Reverse\ Cross-Entropy} - \underbrace{\mathcal{H}\big[\mathtt{p}(\boldsymbol{\pi}|\boldsymbol{x};\boldsymbol{\theta})\big]}_{Differential\ Entropy}\Big] \tag{14}$$

Lets consider the reverse-cross entropy term in more detail (and dropping additive constants):

$$\begin{aligned}\mathcal{L}^{RCE}(\boldsymbol{\theta};\beta) &= \mathbb{E}_{\hat{\mathtt{p}}_{\mathtt{tr}}(\boldsymbol{x})}\Big[\mathbb{E}_{\mathtt{p}(\boldsymbol{\pi}|\boldsymbol{x};\boldsymbol{\theta})}\Big[-\sum_{c=1}^{K}\sum_{k=1}^{K}\hat{\mathtt{P}}_{\mathtt{tr}}(y=\omega_c|\boldsymbol{x})\big(\beta_k^{(c)}-1\big)\ln\pi_k\Big]\Big] \\ &= \mathbb{E}_{\hat{\mathtt{p}}_{\mathtt{tr}}(\boldsymbol{x})}\Big[-\sum_{c=1}^{K}\sum_{k=1}^{K}\hat{\mathtt{P}}_{\mathtt{tr}}(y=\omega_c|\boldsymbol{x})\big(\beta_k^{(c)}-1\big)\big(\psi(\hat{\alpha}_k)-\psi(\hat{\alpha}_0)\big)\Big]\end{aligned} \tag{15}$$

When the target concentration parameters $\boldsymbol{\beta}^{(c)}$are defined as in equation 7, the form of the reverse cross-entropy will be:

$$\mathcal{L}^{RCE}(\boldsymbol{\theta};\beta) = \mathbb{E}_{\hat{\mathtt{p}}_{\mathtt{tr}}(\boldsymbol{x})}\Big[-\beta\sum_{c=1}^{K}\hat{\mathtt{P}}_{\mathtt{tr}}(y=\omega_c|\boldsymbol{x})\big(\psi(\hat{\alpha}_c)-\psi(\hat{\alpha}_0)\big)\Big] \tag{16}$$

This expression for the reverse cross entropy is a scaled version of an upper-bound to the cross entropy between *discrete* distributions, obtained via Jensen's inequality, which was proposed in a parallel work [34] that investigated a model similar to Dirichlet Prior networks:

$$\mathcal{L}^{NLL-UB}(\boldsymbol{\theta}) = \mathbb{E}_{\hat{\mathtt{p}}_{\mathtt{tr}}(\boldsymbol{x})}\Big[-\sum_{c=1}^{K}\hat{\mathtt{P}}_{\mathtt{tr}}(y=\omega_c|\boldsymbol{x})\big(\psi(\hat{\alpha}_c)-\psi(\hat{\alpha}_0)\big)\Big] \tag{17}$$

This form of this upper bound loss is identical to standard negative log-likelihood loss, except with digamma functions instead of natural logarithms. This loss can be analyzed further by considering the following asymptotic series approximation to the digamma function:

$$\psi(x) = \ln x - \frac{1}{2x} + \mathcal{O}(x^2) \approx \ln x - \frac{1}{2x} \tag{18}$$

Given this approximation, it is easy to show that this upper-bound loss is equal to the negative log-likelihood plus an extra term which drives the concentration parameter $\hat{\alpha}_c$ to be as large as possible:

$$\begin{aligned}\mathcal{L}^{NLL-UB}(\boldsymbol{\theta}) &= \mathbb{E}_{\hat{\mathtt{p}}_{\mathtt{tr}}(\boldsymbol{x})}\Big[-\sum_{c=1}^{K}\hat{\mathtt{P}}_{\mathtt{tr}}(y=\omega_c|\boldsymbol{x})\big(\psi(\hat{\alpha}_c)-\psi(\hat{\alpha}_0)\big)\Big] \\ &\approx \mathbb{E}_{\hat{\mathtt{p}}_{\mathtt{tr}}(\boldsymbol{x})}\Big[-\sum_{c=1}^{K}\hat{\mathtt{P}}_{\mathtt{tr}}(y=\omega_c|\boldsymbol{x})\big(\ln(\hat{\pi}_c)-\frac{1-\hat{\pi}}{2\hat{\alpha}_c}\big)\Big] \\ &= \mathcal{L}^{NLL}(\boldsymbol{\theta}) + \mathbb{E}_{\hat{\mathtt{p}}_{\mathtt{tr}}(\boldsymbol{x})}\Big[\sum_{c=1}^{K}\hat{\mathtt{P}}_{\mathtt{tr}}(y=\omega_c|\boldsymbol{x})\big(\frac{1-\hat{\pi}}{2\hat{\alpha}_c}\big)\Big]\end{aligned} \tag{19}$$

Thus, the reverse KL-divergence between Dirichlet distributions, given setting of target concentration parameters via equation 7, yields the following expression:

$$\mathcal{L}^{RKL}(\boldsymbol{\theta};\beta) \approx \beta\cdot\mathcal{L}^{NLL}(\boldsymbol{\theta}) + \mathbb{E}_{\hat{\mathtt{p}}_{\mathtt{tr}}(\boldsymbol{x})}\Big[\beta\sum_{c=1}^{K}\hat{\mathtt{P}}_{\mathtt{tr}}(y=\omega_c|\boldsymbol{x})\big(\frac{1-\hat{\pi}}{2\hat{\alpha}_c}\big) - \mathcal{H}\big[\mathtt{p}(\boldsymbol{\pi}|\boldsymbol{x};\boldsymbol{\theta})\big]\Big] \tag{20}$$

Clearly, this expression is equal to the standard negative log-likelihood loss for discrete distributions, weighted by β, plus a term which drives the precision $\hat{\alpha}_0$ of the Dirichlet to be $\beta + K$, where K is the number of classes.

Appendix B Synthetic Experiments

The current appendix describes the high data uncertainty artificial dataset used in section 4 of this paper. This dataset is sampled from a distribution $\mathtt{p}_{\mathtt{tr}}(\boldsymbol{x}, y)$ which consists of three normally distributed clusters with tied isotropic covariances with equidistant means, where each cluster corresponds to a separate class. The marginal distribution over $\boldsymbol{x}$ is given as a mixture of Gaussian distributions:

$$\mathtt{p}_{\mathtt{tr}}(\boldsymbol{x}) = \sum_{c=1}^{3} \mathtt{p}_{\mathtt{tr}}(\boldsymbol{x}|y=\omega_c) \cdot \mathtt{P}_{\mathtt{tr}}(y=\omega_c) = \frac{1}{3}\sum_{c=1}^{3} \mathcal{N}(\boldsymbol{x}; \boldsymbol{\mu}_c, \sigma^2 \cdot \boldsymbol{I}) \tag{21}$$

The conditional distribution over the classes y can be obtained via Bayes' rule:

$$\mathtt{P}_{\mathtt{tr}}(y=\omega_c|\boldsymbol{x}) = \frac{\mathtt{p}_{\mathtt{tr}}(\boldsymbol{x}|y=\omega_c) \cdot \mathtt{P}_{\mathtt{tr}}(y=\omega_c)}{\sum_{k=1}^{3} \mathtt{p}_{\mathtt{tr}}(\boldsymbol{x}|y=\omega_k) \cdot \mathtt{P}_{\mathtt{tr}}(y=\omega_k)} = \frac{\mathcal{N}(\boldsymbol{x}; \boldsymbol{\mu}_c, \sigma^2 \cdot \boldsymbol{I})}{\sum_{k=1}^{3} \mathcal{N}(\boldsymbol{x}; \boldsymbol{\mu}_k, \sigma^2 \cdot \boldsymbol{I})} \tag{22}$$

This dataset is depicted for $\sigma = 4$ below. The green points represent the 'out-of-distribution' training data, which is sampled close to the in-domain region. The Prior Networks considered in section 4 are trained on this dataset. Figure 7 depicts the behaviour of the differential entropy of Prior Networks

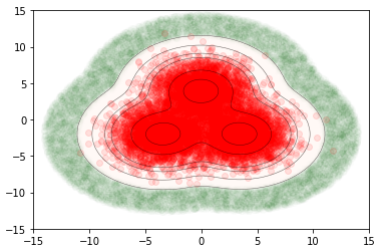
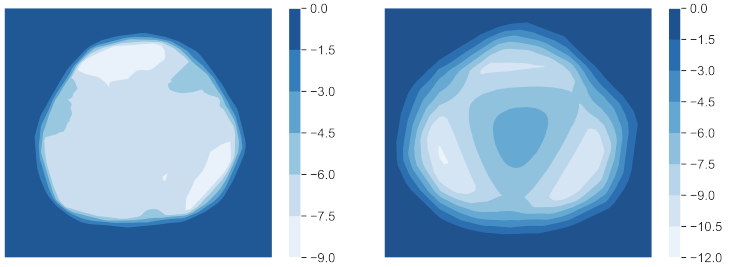


Figure 6: High Data Uncertainty artificial dataset.

trained on the high data uncertainty artificial dataset using both KL-divergence losses. Unlike the *total uncertainty*, *expected data uncertainty* and mutual information, it is less clear what is the desired behaviour of the differential entropy. Figure 7 shows that both losses yield low differential entropy in-domain and high differential entropy out-of-distribution. However, the reverse KL-divergence seems to capture more of the structure of the dataset, which is especially evident in figure 7b, than the forward KL-divergence. This suggests that the differential entropy of Prior Networks trained via *reverse* KL-divergence is a measures of *total uncertainty*, while the differential entropy of Prior Networks trained using *forward* KL-divergence is a measure of *knowledge uncertainty*. The latter is consistent with results in [16].



(a) Differential Entropy PN-KL (b) Differential Entropy PN-RKL

Figure 7: Differential Entropy derived from Prior Networks trained with *forward* and *reverse* KL-divergence loss.

Appendix C Experimental Setup

The current appendix describes the experimental setup and datasets used for experiments considered in this paper. Table 3 describes the datasets used in terms of their size and numbers of classes.

Table 3: Description of datasets in terms of number of images and classes.

Dataset	Train	Valid	Test	Classes
MNIST	55000	5000	10000	10
SVHN	73257	-	26032	10
CIFAR-10	50000	-	10000	10
LSUN	-	-	10000	10
CIFAR-100	50000	-	10000	100
TinyImagenet	100000	10000	10000	200

All models considered in this paper were implemented in Tensorflow [35] using the VGG-16 [2] architecture for image classification, but with the dimensionality of the fully-connected layer reduced down to 2048 units. DNN models were trained using the negative log-likelihood loss. Prior Networks were trained using both the forward KL-divergence (PN-KL) and reverse KL-divergence (PN-RKL) losses to compare their behaviour on more challenging datasets. Identical target concentration parameters $\boldsymbol{\beta}^{(c)}$ were used for both the forward and reverse KL-divergence losses. All models were trained using the Adam [36] optimizer, with a 1-cycle learning rate policy and dropout regularization. In additional, data augmentation was done when training models on the CIFAR-10, CIFAR-100 and TinyImageNet datasets via random left-right flips, random shifts up to ± 4 pixels and random rotations by up to $\pm$ 15 degrees. The details of the training configurations for all models and each dataset can be found in table 4. 5 models of each type were trained starting from different random seeds. The 5 DNN models were evaluated both individually (DNN) and as an explicit ensemble of models (ENS).

C.1 Adversarial Attack Generation

An adversarial input $\boldsymbol{x}_{\texttt{adv}}$ will be defined as the output of a constrained optimization process $\mathcal{A}_{\texttt{adv}}$ applied to a *natural* input $\boldsymbol{x}$:

$$\mathcal{A}_{\texttt{adv}}(\boldsymbol{x}, \omega_t) = \arg\min_{\tilde{\boldsymbol{x}} \in \mathcal{R}^D} \Big\{ \mathcal{L}\big(y = \omega_t, \tilde{\boldsymbol{x}}, \hat{\boldsymbol{\theta}}\big) \Big\} : \ \delta(\boldsymbol{x}, \tilde{\boldsymbol{x}}) < \epsilon \tag{23}$$

The loss $\mathcal{L}$ is typically the negative log-likelihood of a particular target class $y = \omega_t$:

$$\mathcal{L}\big(y = \omega_t, \tilde{\boldsymbol{x}}, \hat{\boldsymbol{\theta}}\big) = \ - \ln \texttt{P}(y = \omega_t | \tilde{\boldsymbol{x}}; \hat{\boldsymbol{\theta}}) \tag{24}$$

The distance $\delta(\cdot, \cdot)$ represents a proxy for the *perceptual distance* between the natural sample $\boldsymbol{x}$ and the adversarial sample $\tilde{\boldsymbol{x}}$. In the case of adversarial images $\delta(\cdot, \cdot)$ is typically the L_1, L_2 or L_∞ norm. The distance $\delta(\cdot, \cdot)$ is constrained to be within the set of allowed perturbations such that the adversarial attack is still *perceived* to be a natural input to a human observer. First-order optimization under a L_p constraint is called Projected Gradient Descent [25], where the solution is projected back onto the L_p-norm ball whenever it exceeds the constraint.

There are multiple ways in which the PGD optimization problem 23 can be solved [17, 18, 19, 20, 25]. The simplest way to generate an adversarial example is via the *Fast Gradient Sign Method* or FGSM [18], where the sign of the gradient of the loss with respect to the input is added to the input:

$$\boldsymbol{x}_{adv} = \boldsymbol{x} - \epsilon \cdot \texttt{sign}(\nabla_{\boldsymbol{x}} \mathcal{L}\big(\omega_t, \boldsymbol{x}, \hat{\boldsymbol{\theta}}\big)) \tag{25}$$

Epsilon controls the magnitude of the perturbation under a particular distance $\delta(\boldsymbol{x}, \boldsymbol{x}_{adv})$, the L_∞ norm in this case. A generalization of this approach to other L_p norms, called *Fast Gradient Methods* (FGM), is provided below:

$$\boldsymbol{x}_{adv} = \boldsymbol{x} - \epsilon \cdot \frac{\nabla_{\boldsymbol{x}} \mathcal{L}\big(\omega_t, \boldsymbol{x}, \hat{\boldsymbol{\theta}}\big)}{||\nabla_{\boldsymbol{x}} \mathcal{L}\big(y_t, \boldsymbol{x}, \hat{\boldsymbol{\theta}}\big)||_p} \tag{26}$$

FGM attacks are simple adversarial attacks which are not always successful. A more challenging class of attacks are iterative FGM attacks, such as the Basic Iterative Method (BIM) [19] and Momentum

Table 4: Training Configurations. η_0 is the initial learning rate, γ is the out-of-distribution loss weight and β is the concentration of the target class. The batch size for all models was 128. Dropout rate is quoted in terms of probability of *not* dropping out a unit.

Training Dataset	Model	η_0	Epochs	Cycle Length	Dropout	γ	β_{in}	OOD data
MNIST	DNN	1e-3	20	10	0.5		-	
	PN-KL PN-RKL					0.0	1e3	-
SVHN	DNN	1e-3	40	30	0.5		-	
	PN-KL PN-RKL	5e-4 5e-6			0.7 0.7	1.0 10.0	1e3	CIFAR-10
CIFAR-10	DNN DNN-ADV	1e-3	45	30	0.5	-	-	- FGSM-ADV
	PN-KL PN-RKL	5e-4 5e-6	45	30	0.7	1.0 10.0	1e2	CIFAR-100
	PN	5e-6	45	30	0.7	30.0	1e2	FGSM-ADV
CIFAR-100	DNN DNN-ADV	1e-3	100	70	0.5	-	-	- FGSM-ADV
	PN-KL PN-RKL	5e-4 5e-6	100	70	0.7	1.0 10.0	1e2	TinyImageNet
	PN	5e-4	100	70	0.7	30.0	1e2	FGSM-ADV
TinyImageNet	DNN	1e-3	120	80	0.5		-	
	PN-KL PN-RKL	5e-4 5e-6				0.0	1e2	-

Iterative Method (MIM) [20], and others [21, 33]. However, as pointed out by Madry et. al [25], all of these attacks, whether one-step or iterative, are generated using variants of *Projected Gradient Descent* to solve the constrained optimization problem in equation 23. Madry [25] argues that all attacks generated using various forms of PGD share similar properties, even if certain attacks use more sophisticated forms of PGD than others.

In this work MIM L_∞ attacks, which are considered to be strong L_∞ attacks, are used to attack all models considered in section 6. However, standard targeted attacks which minimize the negative log-likelihood of a target class are not *adaptive* to the detection scheme. Thus, in this work *adaptive* targeted attacks are generated by minimizing the losses proposed in section 6, in equation 13.

The optimization problem in equation 23 contains a *hard constraint*, which essentially projects the solutions of gradient descent optimization to the allowed L_p-norm ball whenever $\delta(\cdot,\cdot)$ is larger than the constraint. This may be both disruptive to iterative momentum-based optimization methods. An alternative *soft-constraint* formulation of the optimization problem is to simultaneously minimize the loss as well as the perturbation $\delta(\cdot,\cdot)$ directly:

$$\mathcal{A}_{\texttt{adv}}(\boldsymbol{x}, t) = \arg\min_{\tilde{\boldsymbol{x}} \in \mathcal{R}^K} \left\{ \mathcal{L}\big(\omega_t, \tilde{\boldsymbol{x}}, \hat{\boldsymbol{\theta}}\big) + c \cdot \delta(\boldsymbol{x}, \tilde{\boldsymbol{x}}) \right\} \tag{27}$$

In this formulation c is a hyper-parameter which trades of minimization of the loss $\mathcal{L}\big(\omega_t, \tilde{\boldsymbol{x}}, \hat{\boldsymbol{\theta}}\big)$ and the perturbation $\delta(\cdot,\cdot)$. Approaches which minimize this expression are the Carlini and Wagner L_2 (C&W) attack [21] and the "Elastic-net Attacks to DNNs" (EAD) attack [33]. While the optimization expression is different, these methods are also a form of PGD and therefore are expected to have similar properties as other PGD-based attacks [25]. The C&W and EAD are considered to be particularly strong L_2 and L_1 attacks, and Prior Networks need to be assessed on their ability to be robust to and detect them. However, adaptation of these attacks to Prior Networks is non-trivial and left to future work.

C.2 Adversarial Training of DNNs and Prior Networks

Prior Networks and DNNs considered in section 6 are trained on a combination of natural and adversarially perturbed data, which is known as adversarial training. DNNs are trained on L_∞ targeted FGSM attacks which are generated dynamically during training from the current training minibatch. The target class ω_t is selected from a uniform categorical distribution, but such that it is **not** the true class of the image. The magnitude of perturbation ϵ is randomly sampled for each image in the minibatch from a truncated normal distribution, which only yields positive values, with a standard deviation of 30 pixels:

$$\epsilon \sim \mathcal{N}_{pos}(0, \frac{30}{128}) \tag{28}$$

The perturbation strength is sampled such that the model learns to be robust to adversarial attacks across a range of perturbations. The DNN is then trained via maximum likelihood on both the natural and adversarially perturbed version of the minibatch.

Adversarial training of the Prior Network is a little more involved. During training, an adversarially perturbed version of the minibatch is generated using the targeted FGSM method. However, the loss is not the negative log-likelihood of a target class, but the reverse KL-divergence (eqn. 11) between the model and a targeted Dirichlet which is focused on a target class which is chosen from a uniform categorical distribution (but not the true class of the image). For this loss the target concentration is the same as for natural data ($\beta_{in} = 1e2$). The Prior Network is then jointly trained on the natural and adversarially perturbed version of the minibatch using the following loss:

$$\mathcal{L}(\boldsymbol{\theta}, \mathcal{D}) = \mathcal{L}_{in}^{RKL}(\boldsymbol{\theta}, \mathcal{D}_{train}; \beta_{in}) + \gamma \cdot \mathcal{L}_{adv}^{RKL}(\boldsymbol{\theta}, \mathcal{D}_{adv}; \beta_{adv}) \tag{29}$$

Here, the concentration of the target class for natural data is $\beta_{in} = 1e2$ and for adversarially perturbed data $\beta_{adv} = 1$, where the concentration parameters are set via 7. Setting $\beta_{adv} = 1$ results in a very wide Dirichlet distribution whose mode and mean are closest to the target class. This ensures that the prediction yields the correct class and that all measure of uncertainty, such as entropy of the predictive posterior or the mutual information, are high. Note, that due to the nature of the reverse KL-divergence loss, adversarial inputs which have a very small perturbation ϵ and lie close to their natural counterparts will naturally have a target concentration which is an interpolation between the concentration for natural data and for adversarial data. The degree of interpolation is determined by the OOD loss weight γ, as discussed in section 3.

It is necessary to point out that FGSM attack are used because they are computationally cheap to compute during training. However, iterative adversarial attacks can also be considered during training, although this will make training much slower.

Appendix D Jointly Assessing Adversarial Attack Robustness and Detection

In order to investigate detection of adversarial attacks, it is necessary to discuss how to assess the effectiveness of an adversarial attack in the scenario where detection of the attack is possible. Previous work on detection of adversarial examples [37, 38, 39, 14, 15] assesses the performance of detection methods separately from whether an adversarial attack was successful, and use the standard measures of adversarial success and detection performance. However, in a real deployment scenario, an attack can only be considered successful if it *both* affects the predictions *and* evades detection. Here, we develop a measure of performance to assess this.

For the purposes of this discussion the adversarial generation process $\mathcal{A}_{\texttt{adv}}$ will be defined to either yield a successful adversarial attack $\boldsymbol{x}_{\texttt{adv}}$ or an empty set $\emptyset$. In a standard scenario, where there is no detection, the efficacy of an adversarial attack on a model[8] can be summarized via the *success rate* $\mathcal{S}$ of the attack:

$$\mathcal{S} = \frac{1}{N}\sum_{i=1}^{N} \mathcal{I}(\mathcal{A}_{\texttt{adv}}(\boldsymbol{x}_i, \omega_t)), \quad \mathcal{I}(\boldsymbol{x}) = \begin{cases} 1, & \boldsymbol{x} \neq \emptyset \\ 0, & \boldsymbol{x} = \emptyset \end{cases} \tag{30}$$

Typically $\mathcal{S}$ is plotted against the total maximum perturbation ϵ from the original image, measured as either the L_1, L_2 or L_∞ distance from the original image.

Consider using a threshold-based detection scheme where a sample is labelled 'positive' if some measure of uncertainty $\mathcal{H}(\boldsymbol{x})$, such as entropy or mutual information, is less than a threshold T and 'negative' if it is higher than a threshold:

$$\mathcal{I}_T(\boldsymbol{x}) = \begin{cases} 1, & T > \mathcal{H}(\boldsymbol{x}) \\ 0, & T \leq \mathcal{H}(\boldsymbol{x}) \end{cases} \tag{31}$$

The performance of such a scheme can be evaluated at every threshold value using the *true positive rate* $t_p(T)$ and the *false positive rate* $f_p(T)$:

$$t_p(T) = \frac{1}{N}\sum_{i=1}^{N} \mathcal{I}_T(\boldsymbol{x}_i), \quad f_p(T) = \frac{1}{N}\sum_{i=1}^{N} \mathcal{I}_T(\mathcal{A}_{\texttt{adv}}(\boldsymbol{x}_i, \omega_t)) \tag{32}$$

The whole range of such trade offs can be visualized using a Receiver-Operating-Characteristic (ROC) and the quality of the trade-off can be summarized using area under the ROC curve. However, a standard ROC curve does account for situations where the process $\mathcal{A}_{\texttt{adv}}(\cdot)$ fails to produce a successful attack. In fact, if an adversarial attack is made against a system which has a detection scheme, it can only be considered successful if it *both* affects the predictions *and* evades detection. This condition can be summarized in the following indicator function:

$$\hat{\mathcal{I}}_T(\boldsymbol{x}) = \begin{cases} 1, & T > \mathcal{H}(\boldsymbol{x}) \\ 0, & T \leq \mathcal{H}(\boldsymbol{x}) \\ 0, & \boldsymbol{x} = \emptyset \end{cases} \tag{33}$$

Given this indicator function, a new false positive rate $\hat{f}_P(T)$ can be defined as:

$$\hat{f}_p(T) = \frac{1}{N}\sum_{i=1}^{N} \hat{\mathcal{I}}_T(\mathcal{A}_{\texttt{adv}}(\boldsymbol{x}_i, \omega_t)) \tag{34}$$

This false positive rate can now be seen as a new *Joint Success Rate* which measures how many attacks were both successfully generated and evaded detection, given the threshold of the detection scheme. The *Joint Success Rate* can be plotted against the standard true positive rate on an ROC curve to visualize the possible trade-offs. One possible operating point is where the false positive rate is equal to the false negative rate, also known as the *Equal Error-Rate* point:

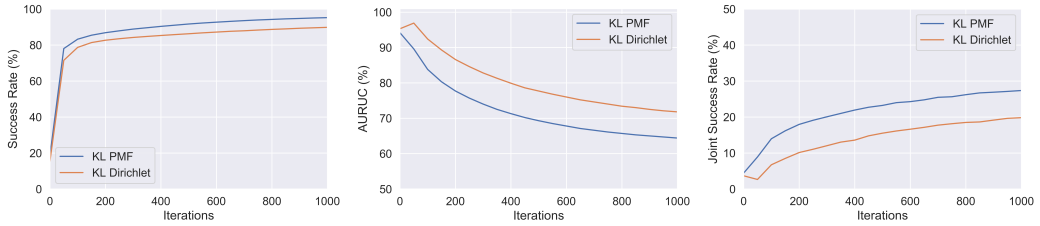
$$\hat{f}_P(T_{\texttt{EER}}) = 1 - t_P(T_{\texttt{EER}}) \tag{35}$$

Throughout this work the EER false positive rate will be quoted as the *Joint Success Rate*.

[8] Given an evaluation dataset $\mathcal{D}_{\texttt{test}} = \{\boldsymbol{x}_i, y_i\}_{i=1}^{N}$

Appendix E Additional Adversarial Attack Detection Experiments

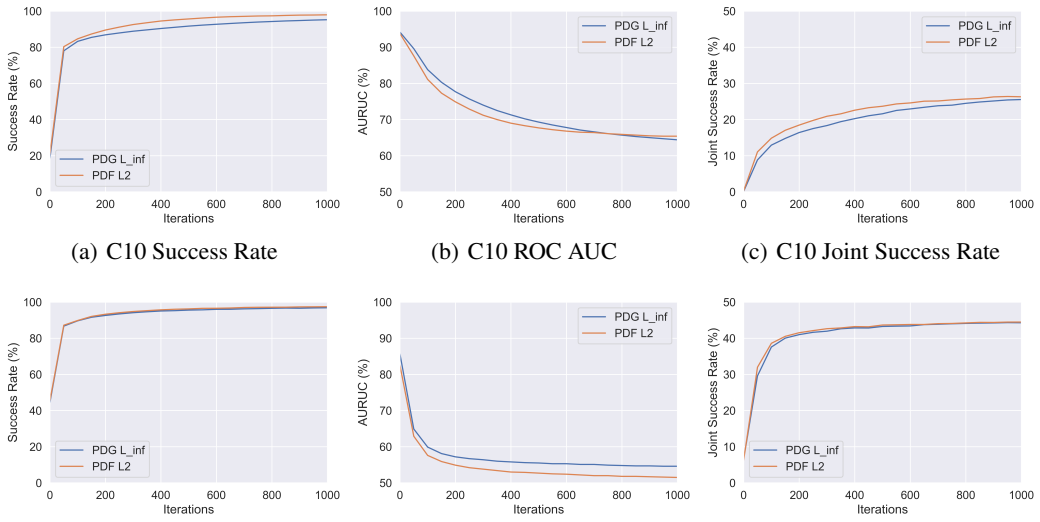
In this appendix additional experiments on adversarial attack detection are presented. In figure 8 adaptive whitebox adversarial attacks generated by iteratively minimizing KL divergence between the original and target (permuted) categorical distributions $\mathcal{L}^{KL}_{PMF}$ are compared to attacks generated by minimzing the KL-divergence between the predicted and permuted Dirichlet distributions $\mathcal{L}^{KL}_{DIR}$. Performance is assessed only against Prior Network models. The results show that KL PMF attacks are more successful at switching the prediction to the desired class and at evading detection. The could be due to the fact that Dirichlet distributions which are sharp at different corners have limited common support, making the optimization of the KL-divergence between them more difficult than the KL-divergence between categorical distributions.



(a) C10 Success Rate (b) C10 ROC AUC (c) C10 Joint Success Rate

Figure 8: Comparison of performance of whitebox adaptive PGD MIM L_∞ attacks which minimize the KL-divergence between PMFs (KL PMF) and Dirichlet distributions (KL DIR) on CIFAR-10.

Results in figure 9 show that L_2 PGD Momentum Iterative attacks which minimize the $\mathcal{L}^{KL}_{PMF}$ loss are marginally more successful than the L_∞ version of these attacks. However, it is necessary to consider appropriate adaptation of the C&W L_2 attacks to the loss functions considered in this work for a more aggressive set of L_2 attacks.



(a) C10 Success Rate (b) C10 ROC AUC (c) C10 Joint Success Rate

(d) C100 Success Rate (e) C100 ROC AUC (f) C100 Joint Success Rate

Figure 9: Comparison of performance of whitebox adaptive L_∞ and L_2 PGD MIM attacks against Prior Networks trained on CIFAR-10 (C10) and CIFAR-100 (C100) datasets.