Are Generative Classifiers More Robust to Adversarial Attacks?

Yingzhen Li 1  John Bradshaw 2 3  Yash Sharma 4

Abstract

There is a rising interest in studying the robustness of deep neural network classifiers against adversaries, with both advanced attack and defence techniques being actively developed. However, most recent work focuses on discriminative classifiers, which only model the conditional distribution of the labels given the inputs. In this paper, we propose and investigate the deep Bayes classifier, which improves classical naive Bayes with conditional deep generative models. We further develop detection methods for adversarial examples, which reject inputs with low likelihood under the generative model. Experimental results suggest that deep Bayes classifiers are more robust than deep discriminative classifiers, and that the proposed detection methods are effective against many recently proposed attacks.

1. Introduction

Deep neural networks have been shown to be vulnerable to adversarial examples (Szegedy et al., 2014; Goodfellow et al., 2015). The latest attack techniques can easily fool a deep net with imperceptible perturbations (Goodfellow et al., 2015; Papernot et al., 2016b; Carlini & Wagner, 2017a; Kurakin et al., 2016; Madry et al., 2018; Chen et al., 2018), even in the black-box case, where the attacker does not have access to the network’s weights (Papernot et al., 2017b; Chen et al., 2017; Brendel et al., 2018; Athalye et al., 2018; Alzantot et al., 2018a; Uesato et al., 2018). Adversarial attacks are serious security threats to machine learning systems, threatening applications beyond image classification (Carlini & Wagner, 2018; Alzantot et al., 2018b).

To address this outstanding security issue, researchers have proposed defence mechanisms against adversarial attacks.

Adversarial training, which augments the training data with adversarially perturbed inputs, has shown moderate success at defending against recently proposed attack techniques (Szegedy et al., 2014; Goodfellow et al., 2015; Tramèr et al., 2018; Madry et al., 2018). In addition, recent advances in Bayesian neural networks have demonstrated that uncertainty estimates can be used to detect adversarial attacks (Li & Gal, 2017; Feinman et al., 2017; Louizos & Welling, 2017; Smith & Gal, 2018). Another notable category of defence techniques involves the usage of generative models. For example, Gu & Rigazio (2014) used an auto-encoder to denoise the inputs before feeding them to the classifier. This denoising approach has been extensively investigated, and the “denoisers” in usage include generative adversarial networks (Samangouei et al., 2018), PixelCNNs (Song et al., 2018) and denoising auto-encoders (Kurakin et al., 2018). These developments rely on the “off-manifold” conjecture – adversarial examples are far away from the data manifold, although Gilmer et al. (2018) has challenged this idea with a synthetic “sphere classification” example.

Surprisingly, much less recent work has investigated the robustness of generative classifiers (Ng & Jordan, 2002) against adversarial attacks, where such classifiers explicitly model the conditional distribution of the inputs given labels. Typically, a generative classifier produces predictions by comparing between the likelihood of the labels for a given input, which is closely related to the “distance” of the input to the data manifold associated with a class. Therefore, generative classifiers should be robust to many recently proposed adversarial attacks if the “off-manifold” conjecture holds for many real-world applications. Unfortunately, many generative classifiers in popular use, including naive Bayes and linear discriminant analysis (Fisher, 1936), perform poorly on natural image classification tasks, making it difficult to verify the “off-manifold” conjecture and the robustness of generative classifiers with these tools.

Are generative classifiers more robust to recently proposed adversarial attack techniques? To answer this, we improve the naive Bayes algorithm by using deep generative models, and evaluate the conjecture on the proposed generative classifier. In summary, our contributions include:

- We propose deep Bayes which models the (conditional) distribution of an input by a deep latent variable model (LVM). We learn the LVM with the variational auto-
encoder algorithm (Kingma & Welling, 2014; Rezende et al., 2014), and for classification we approximate Bayes’ rule using importance sampling.

- We propose three detection methods for adversarial attacks. The first two use the learned generative model as a proxy of the data manifold, and reject inputs that are far away from it. The third computes statistics for the classifier’s output probability vector, and rejects inputs that lead to under-confident predictions.

- We evaluate the robustness of the proposed generative classifier on MNIST and a binary classification dataset derived from CIFAR-10. We further show the advantage of generative classifiers over a number of discriminative classifiers, including Bayesian neural networks and discriminative LVMs.

- We improve the robustness of deep neural networks on CIFAR-10 multi-class classification, by fusing discriminatively learned visual features with the proposed generative classifiers. On defending a number of popular \( \ell_\infty \) attacks, the fusion model outperforms a baseline discriminative VGG16 network (Simonyan & Zisserman, 2014) by a large margin.

### 2. Deep Bayes: conditional deep LVM as a generative classifier

Denote \( p_D(x, y) \) the data distribution for the input \( x \in \mathbb{R}^D \) and label \( y \in \{y_c | c = 1, ..., C\} \), where \( y_c \) is the one-hot encoding vector for class \( c \). For a given \( x \in \mathbb{R}^D \) we can define the ground-truth label by \( y \sim p_D(y|x) \) if \( x \in \text{supp}(p_D(x)) \). We assume the data distribution \( p_D(x, y) \) follows the manifold assumption: for every class \( c \), the conditional distribution \( p_D(x|y_c) \) has its support as a low-dimensional manifold. Thus the training data \( D = \{(x^{(n)}, y^{(n)})\}_{n=1}^N \) is generated as follows:

\[
y^{(n)} \sim p_D(y), x^{(n)} \sim p_D(x|y) \nonumber
\]

A generative classifier first builds a generative model \( p(x, y) = p(x|y)p(y) \), and then, in prediction time, predicts the label \( y^* \) of a test input \( x^* \) using Bayes’ rule,

\[
p(y^*|x^*) = \frac{p(x^*|y^*)p(y^*)}{p(x^*)} = \text{softmax}_c[\log p(x^*, y_c)],
\]

where \( \text{softmax}_c \) denotes the softmax operator over the \( c \) axis. Here the output probability vector is computed analogously to many discriminative classifiers which use softmax activation in the output layer, so many existing attacks can be tested directly. However, unlike discriminative classifiers, the “logit” values prior to softmax activation have a clear meaning here, which is the log joint distribution \( \log p(x^*, y_c) \) of input \( x^* \) and a given label \( y_c \). Therefore, one can also analyse the logit values to determine whether the unseen pair \( (x^*, y^*) \) is legitimate, a utility which will be discussed further in section 3.

**Figure 1.** A visualisation of the graphical models, including both Generative (top row) and Discriminative ones (bottom row), as well as Fully connected and Bottleneck ones. The last character indicates the first node in the topological order of the graph. The graphs are colored in a consistent way as in the result figures.

**Naive Bayes** is perhaps the most well-known generative classifier; it assumes a factorised distribution for the conditional generator, i.e. \( p(x|y) = \prod_{c=1}^C p(x_c|y) \), which is inappropriate for e.g. image and speech data. Fortunately, we can leverage the recent advances in generative modelling and apply a deep generative model for the joint distribution \( p(x, y) \). More specifically, we use a deep latent variable model (LVM) \( p(x, z, y) \) to construct a generative classifier. Importantly, this leads to a conditional distribution \( p(x|y) = \int p(x, z, y) dz = p(z|y)p(x|z, y) \), which is much more powerful than naive Bayes. We refer to such generative classifiers that use deep generative models as deep Bayes classifiers.

Depending on the definition of the model \( p(x, z, y) \), the resulting classifier can be either generative or discriminative. Thus we evaluate the effect of different factorisation structures on the robustness of the induced classifier from the joint distribution \( p(x, z, y) \) (see Figure 1).

\[
\begin{align*}
p(x, z, y) &= p(z)p(y|z)p(x|z, y) \quad \text{(GFZ)} \\
p(x, z, y) &= p_D(y)p(z|y)p(x|z, y) \quad \text{(GFX)} \\
p(x, z, y) &= p(z)p(y|z)p(x|z) \quad \text{(GBZ)} \\
p(x, z, y) &= p_D(y)p(z|y)p(x|z) \quad \text{(GBY)} \\
p(x, z, y) &= p_D(x)p(z|x)p(y|z, x) \quad \text{(DFX)} \\
p(x, z, y) &= p(x)p(z|y)p(y|z, x) \quad \text{(DFZ)} \\
p(x, z, y) &= p_D(x)p(z|y)p(y|z) \quad \text{(DBX)}
\end{align*}
\]

We use the initial character “G” to denote generative classifiers and “D” to denote discriminative classifiers. Models with the second character as “F” have a fully connected graph, while “B” models have bottleneck structures. The last character of the model name indicates the first node in topological order. We do not test other architectures under this nomenclature, as either the graph contains directed cycles (e.g. \( x \rightarrow y \rightarrow z \rightarrow x \)), or \( z \) is the last node in topological order (e.g. \( x \rightarrow y \rightarrow (x, y) \rightarrow z \)) so that the marginalisation of \( z \) does not affect classification.
Within the generative classifiers in our design, different graphical models impose different assumptions on the data generation process. E.g. GFZ and GBZ assume there is a confounder \( z \) that affects both \( x \) and \( y \), while GFY and GBY assume the distribution over \( z \) is class-dependent. The bottleneck models GBZ, GBY and DBX, when compared with their fully-connected counterparts, enforces the usage of the latent code \( z \) for representation learning.

For training, we follow Kingma & Welling (2014) and Rezende et al. (2014) to introduce an amortised approximate posterior \( q(z|x, y) \), and train both \( p \) and \( q \) by maximising the variational lower-bound:

\[
\mathbb{E}_D[\mathcal{L}_{VI}(x, y)] = \frac{1}{N} \sum_{n=1}^{N} \mathbb{E}_q \left[ \log \frac{p(x_n, z_n, y_n)}{q(z_n|x_n, y_n)} \right]. \tag{1}
\]

After training, the predicted class probability vector \( y^* \) for a future input \( x^* \) is computed by an approximation to Bayes’ rule with importance sampling \( z^k \sim q(z^k|x^*, y_c) \):

\[
p(y^*|x^*) \approx \max_{c=1}^{C} \left[ \log \frac{1}{K} \sum_{k=1}^{K} \frac{p(x^*, z^k, y_c)}{q(z^k|x^*, y_c)} \right]. \tag{2}
\]

Therefore, for generative classifiers the probability of \( x \) under the generative model affects the predictions. By contrast, DFX and DBX do not know if \( x \) is close to the data manifold or not, as the \( p_D(x) \) term in these models is cancelled out in eq. (2). Model DFZ is somewhat intermediate, as it builds a generative model for the inputs \( x \) (thus \( p(x, z) \) is used in eq. (2)) but also directly parameterises the conditional distribution \( p(y|x, z) \).

3. Detecting adversarial attacks with generative classifiers

We propose detection methods for adversarial examples using the generative classifier’s logit values. As an illustrative example, consider a labelled dataset of “cat” and “dog” images. If an adversarial image of a cat \( x_{adv} \) is incorrectly labelled as “dog”, then the logit of “dog” will be significantly lower than normal. This means we can detect attacks using the logits \( \log p(x^*, y_c, c = 1, \ldots, C) \) computed on a test input \( x^* \). The goal here is to reject both unlabelled input \( x \) that have low probability under \( p(x) \) (as a proxy to \( p_D(x) \)), and labelled data \( (x, y) \) that have low \( p(x) \) values. Concretely, the proposed detection algorithms are as follows.

- **Marginal detection**: rejecting inputs that are far away from the manifold.

  One can select a threshold \( \delta \) and reject an input \( x \) if \( -\log p(x) > \delta \). To determine the threshold \( \delta \), we can compute \( d_D = \mathbb{E}_{x \sim D}[-\log p(x)] \) and \( \sigma_D = \sqrt{\mathbb{V}_{x \sim D}[-\log p(x)]} \), then set \( \delta = d_D + \alpha \sigma_D \).

- **Logit detection**: rejecting inputs using joint density.

  Given a victim model \( y = F(x) \), one can reject \( x \) if \( -\log p(x, F(x)) > \delta_y \). We can use the mean and variance statistics \( \mu_y, \sigma_y \) computed on \( \log p(x, y_c) \) and select \( \delta_y = \mu_y + \alpha \sigma_y \).

- **Divergence detection**: rejecting inputs with over-and/or under-confident predictions.

  Denote \( p(x) \) as a \( C \)-dimensional probability vector outputted by the classifier. For each class \( c \), we first collect the mean classification probability vector \( \bar{p}_c = \mathbb{E}_{y \sim D}(p(x)) \), then compute the mean \( \bar{d}_c \) and standard deviation \( \sigma_c \) on a selected divergence/distance measure \( D[\bar{p}_c||p(x)] \) for all \( (x, y_c) \in D \). A test input \( x^* \) with prediction label \( c^* = \arg\max_c p(x^*) \) is rejected if \( D[\bar{p}_c||p(x^*)] > \bar{d}_c + \alpha \sigma_c \). Therefore, an example \( x^* \) can be rejected if the probability vector \( p(x^*) \) is very different to the ones seen in training.

When \( D \) is the KL-divergence, we call this method **KL detection**. Other divergence/distance measures such as total variation (TV) can also be used.

For better intuition, we visualise the detection mechanisms in Figure 2 with a synthetic “two rings” binary classification example. In this case we sample \( 2 \times 1000 \) training data points by \( y \sim \text{Bern}(0.5), \theta \sim \text{Uniform}(0, 2\pi), x|y \sim \mathcal{N}(x; c_y + r_y \cos(\theta), \sin(\theta)^T, \sigma^2 I) \). We consider a generative classifier \( p(x, y) = p(x|y)p_D(y), p(x|y) = \mathcal{N}(x; \mu_y, \sigma^2 I), \mu_y = \arg\min ||x - c_y||_2 = r_y \). The \( \delta \) thresholds are selected to achieve 10% false positive rates on training data. From the visualisations we see that inputs that are far away from the model manifold are rejected by marginal/logit detection. At the same time, logit detection rejects data points that are not on the manifold of the given class. KL/TV detection does not construct manifold-aware acceptance regions, which is as expected since the proposed divergence detection method does not require the classifier to be generative. However, both detection methods have...
some success in rejecting uncertain predictions, especially for TV, which also rejects ambiguous inputs (see the ring-cress regions in the last two plots). Combining all three methods, we see that the rejected inputs are either far away from the manifold, or are ambiguous.

Detection methods using logit values have been used in e.g. Li & Gal (2017); Feinman et al. (2017), but it is unclear whether the logits values in discriminative classifiers have a semantic meaning. Song et al. (2018); Samangouei et al. (2018); Kurakin et al. (2018) trained a separate generative model for denoising/detection. But the features of the generator and the discriminative classifier can be very different, hence the generator cannot detect the “manifold attack” against the classifier (Gilmer et al., 2018). Unlike these approaches, we highlight three critical properties of generative classifiers and the accompanied detection methods:

1. the representations of the data manifold are the same for both the classifier and the detector (since they share the same generative model \( p(x, y) \approx p_D(x, y) \);
2. the logit values in generative classifiers have a clear semantic meaning: the log probability \( \log p(x, y) \) of generating the input \( x \) given the class label \( y = y_c \); also note that \( F(x) = \arg \max_y \log p(x, y_c) \);
3. the marginal/logit detection aims at rejecting \( x \) that has low \( \log p(x) \) and/or \( \log p(x, y) \). These probabilities are obtained as part of the classification procedure and so do not require the running of any extra models.

Assuming an accurate approximation \( p(x, y) \approx p_D(x, y) \), a new input \( x^* \) is accepted by marginal/logit detection only if \( x^* \) is close to the manifold of \( F(x^*) \) (thus \( p(x^*, y = F(x^*)) \) is high).\(^3\) So \( F(x^*) \) should be equal to the ground truth label \( y^* \) if \( x^* \) is not an ambiguous input (which will be detected by KL/TV detection). Therefore the “off-manifold” conjecture should hold for a powerful generative classifier, and below we present an empirical study to validate the assumptions of this conjecture.

4. Experiments

We carry out a number of tests on the deep Bayes classifiers, our implementation is available at https://github.com/deepgenerativeclassifier/DeepBayes.

The distributions \( q(z|\cdot) \) and \( p(z|\cdot) \) are factorised Gaussians, and the conditional probability \( p(x|\cdot) \), if required, is parameterised by an \( \ell_2 \) loss. Besides the LVM-based classifiers, we further train a Bayesian neural network (BNN) with Bernoulli dropout (dropout rate 0.3), as it has been shown in Li & Gal (2017) and Feinman et al. (2017) that BNNs are more robust than their deterministic counterparts. The constructed BNN has 2x more channels than LVM-based classifiers, making the comparison slightly “unfair”, as the BNN layers have more capacity. We use \( K = 10 \) Monte Carlo samples for all the classifiers.

The adversarial attacks are taken from the CleverHans 2.0 library (Papernot et al., 2017a). Three metrics are reported: accuracy of the classifier on crafted adversarial examples, mean minimum perturbation distance computed on adversarial examples that have successfully fooled the classifier, and detection rate on successful attacks. This detection rate is defined as the true positive (TP) rate of finding an adversarial example, and the detection threshold is selected to achieve a 5% false positive rate on clean training data.

The experiments are performed under various threat model settings. In the main text we present gradient-based attacks and gradient masking sanity checks, and provide a brief summary of further experiments presented in the appendix. Full table results can also be found in appendix F.

4.1. Gradient-based attacks

We first evaluate the robustness of generative classifiers against gradient-based attacks. These attacks are performed under a white-box setting against the classifier (Carlini & Wagner, 2017b): the attacker can differentiate through the classifier, but is not aware of the existence of the detector. We report results on two datasets (MNIST and CIFAR-binary) and two classes of attacks (\( \ell_\infty \) and \( \ell_2 \) attacks).

Datasets For MNIST tests, we use \( \dim(z) = 64 \) for the LVM-based classifiers. All classifiers achieve \( > 98\% \) clean test accuracy, therefore we apply attacks on the whole test dataset (10,000 datapoints). Since the robustness properties of MNIST classifiers might not extend to natural images (Carlini & Wagner, 2017b), we further consider the same set of evaluations on CIFAR-binary, a binary classification dataset containing “airplane” and “frog” images from CIFAR-10. We choose to work with this simpler dataset because (1) fully generative classifiers are less satisfactory for classifying clean CIFAR-10 images\(^2\), and (2) we want to evaluate whether the robustness properties of fully generative classifiers hold on natural images (c.f. Carlini & Wagner, 2017b). On CIFAR-binary, the generative classifiers use \( \dim(z) = 128 \) and obtain \( > 90\% \) clean test accuracy (see appendix). The attacks are then performed on the test images that all models initially correctly classify, leading to a test set of 1,577 instances for CIFAR-binary. For both datasets the images are scaled to \([0, 1]\).

\(^3\)Nalisnick et al. (2019) showed that existing generative models fail to detect all possible outliers. A solution to counter this issue in logit/marginal detection is to add a lower-bounding threshold so that an input-output pair \((x, y)\) will be rejected if \(-\log p(x, y) < \zeta_y\). We leave this investigation to future work.

\(^2\)The clean test accuracies for GFZ & GFY on CIFAR-10 are all \(< 50\%\); a conditional PixelCNN++ (Salimans et al., 2017) (with much deeper networks) achieves 72.4% clean test accuracy.
Are Generative Classifiers More Robust to Adversarial Attacks?

Figure 3. Victim accuracy, detection rates and minimum $\ell_{\infty}$ perturbation against white-box FGSM attacks on MNIST. The higher the better. The visualised adversarial examples (not necessarily successful) are crafted with with $\ell_{\infty}$ distortion $\epsilon$ growing from 0.1 to 0.5.

Figure 4. Victim accuracy, detection rates and minimum $\ell_{\infty}$ perturbation against white-box FGSM attacks on CIFAR-binary. The higher the better. The visualised adversarial examples (not necessarily successful) are crafted with with $\ell_{\infty}$ distortion $\epsilon$ growing from 0.01 to 0.2.

$\ell_{\infty}$ attacks The attacks in test are: fast gradient sign method (FGSM, Goodfellow et al., 2015), projected gradient descent (PGD, Madry et al., 2018) and momentum iterative method (MIM, Dong et al., 2018). We use the distortion strengths as $\epsilon \in \{0.1, 0.2, 0.3, 0.4, 0.5\}$ for MNIST, and $\epsilon \in \{0.01, 0.02, 0.05, 0.1, 0.2\}$ for CIFAR-binary.

Results are reported in Figure 3 for MNIST and Figure 4 for CIFAR-binary, respectively. For both datasets, generative classifiers perform generally better in terms of victim accuracy, especially GBZ is significantly more robust than the others on CIFAR-binary. By contrast, discriminative VAE-based classifiers are less robust, e.g. on MNIST, DFX & DFZ are not robust to the weakest attack (FGSM) even when $\epsilon = 0.2$ (where the distorted inputs are still visually close to the original digit “7”). Interestingly, DBX is relatively robust against FGSM on both datasets, which agrees with the preliminary tests in Alemi et al. (2017). Further investigations in appendix A.2 show that the bottleneck structure might be beneficial for defending certain attacks.

In terms of minimum perturbation which is computed across all $\epsilon$ settings, quantitatively the amount of distortion required to fool generative classifiers is much higher than that for discriminative ones. Indeed on both datasets, the visual distortion of the adversarial examples on generative classifiers is more significant.

For detection, generative classifiers successfully detect the adversarial examples with $\epsilon \geq 0.3$ on MNIST, which is reasonable as the visual distortion is already significant. Detection results on CIFAR-binary are less satisfactory though: marginal/logit detection fail to detect attacks with $\epsilon = 0.1$ (which attain both high success rate and induce visually perceptible distortion). These results suggest that the per-pixel $\ell_2$ loss might not be best suited for modelling natural images (c.f. Larsen et al., 2016; van den Oord et al., 2016), indeed we present improved robustness results in section 4.3, where the generative classifiers use an alternative likelihood function that is closely related to the perceptual loss (Dosovitskiy & Brox, 2016; Johnson et al., 2016).

3We manually assign the minimum perturbation of an input as $\epsilon_{\max} + 0.1$ if none of the attacks is successful.
Are Generative Classifiers More Robust to Adversarial Attacks?

**Figure 5.** Accuracy, $\ell_2$ distortion, and detection rates against white-box CW attacks on MNIST.

**Figure 6.** Clean inputs and CW adversarial examples ($c = 10$) crafted on GFZ, digits in red rectangles show significant ambiguity.

As a side note, DFZ, as an intermediate between generative and discriminative classifiers, has worse robustness results, but has good detection performance for the marginal and logit metrics. This is because with softmax activation, the marginal distribution $p(x)$ is dropped, but in marginal/logit detection $p(x)$ is still in use. KL detection works well for all classifiers, and on CIFAR-binary, discriminative classifiers start to dominate in this metric as $\epsilon$ increases. Inspecting the visualised adversarial examples on generative classifiers (Figure 4), we see that for large $\epsilon$ values, these inputs are significantly distorted, thus “not ambiguous”, indeed they are detected by marginal/logit detection methods.

### $\ell_2$ attack
We perform the Carlini & Wagner (CW) $\ell_2$ attack (Carlini & Wagner, 2017a), with a loss-balancing parameter $c \in \{0.1, 1, 10, 100, 1000\}$, so that the $\ell_2$ distortion regulariser in CW’s loss function decreases with larger $c$.

MNIST results are reported in Figure 5. Here GBZ is a clear winner: for a given level of $\ell_2$ distortion, GBZ is significantly more robust than the others. Other classifiers perform similarly on MNIST, however, these attack successes on generative classifiers is mainly due to the ambiguity of the crafted adversarial images. As visualised in Figure 6 (and Figure B.4 in appendix), the induced distortion from CW leads to ambiguous digits which sit at the perceptual boundary between the original and the adversarial classes.

### Summary
The tested generative classifiers are generally more robust than the discriminative ones against white-box gradient-based attacks. In particular, the generative classifier’s victim accuracy decreases as the distortion increases, but at the same time the TP rates for marginal/logit detection also increase. Therefore the two attacks in test fail to find near-manifold adversarial examples that fool both the classifier and the detector components of generative classifiers.

### 4.2. Sanity checks on gradient masking
If a successful defence against white-box gradient-based attacks is due to gradient masking, then this defence is likely to be less effective against attacks that do not differentiate through the victim classifier and the defence (Papernot et al.,
Table 1. Mean minimum $\ell_\infty$ perturbation (in red, computed on $\epsilon \in \{0.1, 0.2, 0.3, 0.4, 0.5\}$) and victim accuracy (in blue, for $\epsilon \leq 0.3$) for $\ell_\infty$ attacks on MNIST.

<table>
<thead>
<tr>
<th>Attack</th>
<th>GFZ</th>
<th>GBZ</th>
<th>GFY</th>
<th>GBY</th>
</tr>
</thead>
<tbody>
<tr>
<td>PGD (white)</td>
<td>0.23 / 7.71%</td>
<td>0.30 / 30.78%</td>
<td>0.21 / 5.52%</td>
<td>0.22 / 8.89%</td>
</tr>
<tr>
<td>MIM (white)</td>
<td>0.24 / 7.02%</td>
<td>0.24 / 4.97%</td>
<td>0.22 / 6.72%</td>
<td>0.21 / 1.54%</td>
</tr>
<tr>
<td>PGD (grey)</td>
<td>0.37 / 51.08%</td>
<td>0.36 / 50.64%</td>
<td>0.38 / 53.29%</td>
<td>0.36 / 48.66%</td>
</tr>
<tr>
<td>MIM (grey)</td>
<td>0.34 / 43.00%</td>
<td>0.33 / 40.94%</td>
<td>0.34 / 46.64%</td>
<td>0.33 / 40.06%</td>
</tr>
<tr>
<td>PGD (black)</td>
<td>0.40 / 61.93%</td>
<td>0.42 / 66.75%</td>
<td>0.38 / 56.35%</td>
<td>0.43 / 68.50%</td>
</tr>
<tr>
<td>MIM (black)</td>
<td>0.36 / 50.44%</td>
<td>0.38 / 59.86%</td>
<td>0.36 / 48.07%</td>
<td>0.39 / 61.78%</td>
</tr>
</tbody>
</table>

Table 2. Mean minimum $\ell_\infty$ perturbation (in red, computed on $\epsilon \in \{0.01, 0.02, 0.03, 0.1, 0.2\}$) and victim accuracy (in blue, for $\epsilon \leq 0.1$) for $\ell_\infty$ attacks on CIFAR-binary.

<table>
<thead>
<tr>
<th>Attack</th>
<th>GFZ</th>
<th>GBZ</th>
<th>GFY</th>
<th>GBY</th>
</tr>
</thead>
<tbody>
<tr>
<td>PGD (white)</td>
<td>0.11 / 21.81%</td>
<td>0.20 / 65.63%</td>
<td>0.11 / 25.81%</td>
<td>0.11 / 25.24%</td>
</tr>
<tr>
<td>MIM (white)</td>
<td>0.09 / 15.22%</td>
<td>0.13 / 37.60%</td>
<td>0.10 / 16.45%</td>
<td>0.09 / 14.39%</td>
</tr>
<tr>
<td>PGD (grey)</td>
<td>0.15 / 50.48%</td>
<td>0.23 / 77.30%</td>
<td>0.16 / 54.66%</td>
<td>0.17 / 57.96%</td>
</tr>
<tr>
<td>MIM (grey)</td>
<td>0.15 / 47.62%</td>
<td>0.21 / 75.71%</td>
<td>0.15 / 51.11%</td>
<td>0.16 / 53.84%</td>
</tr>
<tr>
<td>PGD (black)</td>
<td>0.19 / 68.36%</td>
<td>0.23 / 79.45%</td>
<td>0.20 / 70.13%</td>
<td>0.19 / 67.98%</td>
</tr>
<tr>
<td>MIM (black)</td>
<td>0.18 / 66.39%</td>
<td>0.23 / 78.38%</td>
<td>0.19 / 68.42%</td>
<td>0.19 / 66.52%</td>
</tr>
</tbody>
</table>

Table 3. Victim accuracy results against white-box PGD and black-box SPSA attack. We use $\epsilon = 0.3$ for MNIST and $\epsilon = 0.05$ for CIFAR-binary.

<table>
<thead>
<tr>
<th>Attack</th>
<th>MNIST</th>
<th>CIFAR-binary</th>
</tr>
</thead>
<tbody>
<tr>
<td>PGD</td>
<td>GYF</td>
<td>GYF</td>
</tr>
<tr>
<td>SPSA</td>
<td></td>
<td></td>
</tr>
<tr>
<td>PGD (white)</td>
<td>4.0%</td>
<td>68.2%</td>
</tr>
<tr>
<td>MIM (white)</td>
<td>29.7%</td>
<td>79.0%</td>
</tr>
<tr>
<td>PGD (grey)</td>
<td>7.4%</td>
<td>71.0%</td>
</tr>
<tr>
<td>MIM (grey)</td>
<td>5.4%</td>
<td>59.9%</td>
</tr>
<tr>
<td>PGD (black)</td>
<td>2.3%</td>
<td>55.9%</td>
</tr>
<tr>
<td>MIM (black)</td>
<td>3.3%</td>
<td>56.3%</td>
</tr>
</tbody>
</table>

2017b; Athalye et al., 2018). Therefore, here we present two types of attacks as sanity checks on gradient masking.

**Distillation-based attacks** We perform two attacks based on distilling the victim classifier using a “student” CNN. The two attacks differ in their threat models: in the **grey-box** setting the attacker has access to both the training data and the output probability vectors of the classifiers on the training set, while in the **black-box** setting the attacker only has access to queried labels on a given input. For the latter black-box setting, we follow Papernot et al. (2017b) to train a substitute CNN using Jacobian-based dataset augmentation (see appendix E.1). We then craft adversarial examples on the grey-/black-box substitutes using PGD and MIM, and transfer them to the victim classifiers. As a sanity check, these attacks with reasonably small $\epsilon$ values achieve $\sim 100\%$ success rates on fooling the substitutes ($\epsilon \geq 0.2$ for MNIST and $\epsilon \geq 0.1$ for CIFAR-binary, see appendix F).

We report the results for these transferred attacks in Table 1 for MNIST and Table 2 for CIFAR-binary, respectively, with a comparison to white-box attack results taken from the last section. It is clear that the adversarial examples crafted on substitute models do not transfer very well to the generative classifiers. Importantly, for a fixed $\epsilon$ setting, the white-box attacks achieve significantly higher success rates than their grey-/black-box counterparts, and the gap is at least $\geq 20\%$ for MNIST (with $\epsilon \leq 0.3$) and $\geq 30\%$ for CIFAR-binary (with $\epsilon \leq 0.1$). Furthermore, the mean minimum $\ell_\infty$ perturbation obtained by grey-/black-box attacks is significantly higher than those obtained by white-box attacks.

**SPSA (evolutionary strategies)** We consider another black-box setting that only accesses access to the logit values of the prediction given an input. We use the SPSA $\ell_\infty$ attack (Uesato et al., 2018), which is similar to the white-box attacks, except that gradients are numerically estimated using the logit values from the victim classifier. Results are presented in Table 3 for MNIST (using 1,000 randomly sampled test datapoints) and CIFAR-binary. Both experiments clearly show that SPSA performs much worse on generative classifiers when compared to white-box PGD.

**Summary** Both distillation-based attacks and score-based attack (SPSA) failed to obtain higher success rate than gradient-based attacks against generative classifiers. Therefore, gradient masking is unlikely to be responsible for the improved robustness of generative classifiers, as utilising the exact gradient yielded better success rates for the attacks.

### 4.3. Combining deep Bayes and discriminative features

This experiment examines the robustness of CIFAR-10 **multi-class** classifiers, with the generative classifiers trained on **discriminative** visual features. To do this, we download a pretrained VGG16 network (Simonyan & Zisserman, 2014) on CIFAR-10 (93.50% test accuracy), and use its features $\phi(x)$ as the input to the VAE-based classifiers: $p(y|x) = p(y|\phi(x))$. This means $p(x|\cdot)$ of the generative classifiers is defined by a perceptual loss (Dosovitskiy & Brox, 2016; Johnson et al., 2016), see appendix C for a discussion. The classifiers in test include GBZ, GYB and DBX. We use fully-connected neural networks for these classifiers, and select from VGG16 the 9th convolution layer ($\text{CONV}9$) and the first fully connected layer after convolution (FCL1) as the feature layers to ensure $\sim 90\%$ test accuracy.

Results on white-box $\ell_\infty$ attacks are visualised in Figure 8. For all LVM-based classifiers we see clear improvements in robustness and detection over the VGG16 baseline. In particular, GBZ and GYB with $\text{CONV}9$ features are overall better than DBX. More importantly, generative classifiers based on $\text{CONV}9$ features are significantly more robust than those based on FCL1 features. By contrast, for DBX, which is **discriminative**, the robustness results are very similar. This indicates that the level of feature representation has little effect for DBX, presumably DBX-$\text{CONV}9$ has learned high-level features that resembles FCL1. Also the logit detection method...
Are Generative Classifiers More Robust to Adversarial Attacks?

works much better on the fusion models when compared with the generative classifiers using \(\ell_2\) likelihood (c.f. Figure 4). These results suggest that one can achieve both high clean accuracy and better robustness/detection rates against adversaries by combining discriminatively learned visual features and generative classifiers.

4.4. Summary of additional studies

We present in appendix A further studies on the robustness properties of generative and discriminative classifiers.

- In appendix A.1, we designed a white-box attack targeting both the classifier and the detector, which also considers the usage of random \(z\) samples by the LVM-based classifiers (Biggio et al., 2013; Carlini & Wagner, 2017b). In results, although this attack can reduce detection levels, it comes with the trade-off of increasing accuracy, suggesting that this adversary cannot break both the classifier and detector working in tandem.

- In appendix A.2, we quantified the bottleneck effect by varying \(\text{dim}(z)\) in Bottleneck classifiers. Results indicate that using a small bottleneck improves the robustness of the classifier against \(\ell_{\infty}\) attacks.

- In appendix A.3, we evaluated the transferability of adversarial examples across different LVM-based classifiers. We found that these transferred attacks are relatively effective between generative classifiers, but not from generative to discriminative (and vice versa).

5. Discussion

We have proposed deep Bayes as a generative classifier that uses deep LVMs to model the joint distribution of input-output pairs. We have given evidence that generative classifiers are more robust to many recent adversarial attacks than discriminative classifiers. Furthermore, the logit in generative classifiers has a well-defined meaning and can be used to detect attacks, even when the classifier is fooled.

Concurrent to us, Schott et al. (2019) also demonstrated the robustness of generative classifiers on MNIST, in which their graphical model corresponds to GBY in our design, and the logits are computed by a tempered version of the variational lower-bound. However, their approach requires thousands of random \(z\) samples and tens of optimisation steps to approximate \(\log p(x|y)\) for every input-output pair \((x, y)\), making it much less scalable than our importance sampling technique. Indeed, we have scaled our approach to CIFAR-10, a natural image dataset, and the robustness results are consistent with those on MNIST.

Importantly, the graphical model structure has a significant impact on robustness, which is not mentioned in Schott et al. (2019). Our study shows that deep LVM-based generative classifiers generally outperform the (randomised) discriminative ones, and the bottleneck is useful for defending against \(\ell_{\infty}\) attacks. Our results corroborate with the Bayesian neural network literature, in particular Li & Gal (2017); Feinman et al. (2017); Carlini & Wagner (2017b), in showing that modelling unobserved variables are effective for defending against adversarial attacks.

While we have given strong evidence to suggest that generative classifiers are more robust to current adversarial attacks, we do not wish to claim that these models will be robust to all possible attacks. Aside from many recent attacks being designed specifically for discriminative neural networks, there is also evidence for the fragility of generative models; e.g. naive Bayes as a standard approach for spam filtering is well-known to be fragile (Dalvi et al., 2004; Huang et al., 2011), and recently Tabacof et al. (2016); Kos et al. (2017); Creswell et al. (2017) also designed attacks for (unconditional) VAEs. However, generative classifiers can be made more robust too, to counter these weaknesses. Dalvi et al. (2004) have shown that generative classifiers can be made more secure if aware of the attack strategy, and Biggio et al. (2011; 2014) further improved naive Bayes’ robustness by modelling the conditional distribution of the adversarial inputs. These approaches are similar to the adversarial training of discriminative classifiers (Tramèr et al., 2018; Madry et al., 2018), efficient ways for doing so with generative classifiers can be an interesting research direction.

But even with this note of caution, we believe this work offers exciting avenues for future work. Using generative classifiers offers an interesting way to evaluate generative models and can drive improvements in their ability to tackle high-dimensional datasets, where traditionally generative classifiers have been less accurate than discriminative classifiers (Efron, 1975; Ng & Jordan, 2002). In addition, the combination of generative and discriminative models is a compelling direction for future research. Overall, we believe that progress on generative classifiers can inspire better designs of attack, defence and detection techniques.
Are Generative Classifiers More Robust to Adversarial Attacks?

Acknowledgements
John Bradshaw acknowledges support from an EPSRC studentship.

References


Are Generative Classifiers More Robust to Adversarial Attacks?


