Deep Verifier Networks: Verification of Deep Discriminative Models with Deep Generative Models

Tong Che [*] [1] Xiaofeng Liu [*] [2 3] Site Li [3] Yubin Ge [4] Ruixiang Zhang [1] Caiming Xiong [5] Yoshua Bengio [1]

Abstract

AI Safety is a major concern in many deep learning applications such as autonomous driving. Given a trained deep learning model, an important natural problem is how to reliably verify the model's prediction. In this paper, we propose a novel framework — deep verifier networks (DVN) to detect unreliable inputs or predictions of deep discriminative models, using separately trained deep generative models. Our proposed model is based on conditional variational auto-encoders with disentanglement constraints to separate the label information from the latent representation. We give both intuitive and theoretical justifications for the model. Our verifier network is trained independently with the prediction model, which eliminates the need of retraining the verifier network for a new model. We test the verifier network on both out-of-distribution detection and adversarial example detection problems, as well as anomaly detection problems in structured prediction tasks such as image caption generation. We achieve state-of-the-art results in all of these problems.

1. Introduction

Deep learning models provide state-of-the-art performance in various applications such as image classification (Krizhevsky et al., 2012), caption generation (Xu et al., 2015), sequence modeling (Chung et al., 2014) and machine translation (Wu et al., 2016). However, such performance is based on the assumption that the training and testing data are sampled from the same distribution (Goodfellow et al., 2016). Without this assumption, deep learning models can fail silently by producing high confidence incorrect predictions even on completely unrecognizable or irrelevant inputs (Amodei et al., 2016). For instance, the models trained on MNIST can produce 91% confidence on random noise images (Hendrycks & Gimpel, 2017). Generally speaking, the behavior of a trained deep learning model on a slightly different test distribution is unpredictable. One such problematic case is also shown in Fig. 1. Unfortunately, there is very little control over the test distribution in real-world deployments due to dynamically changing environments or malicious attacks (Guo et al., 2017). In fact, well calibrating the predictive uncertainty of DNNs is important for many production systems, $e.g.$, authentication devices, medical diagnosis and self-driving vehicles (Lee et al., 2018b).

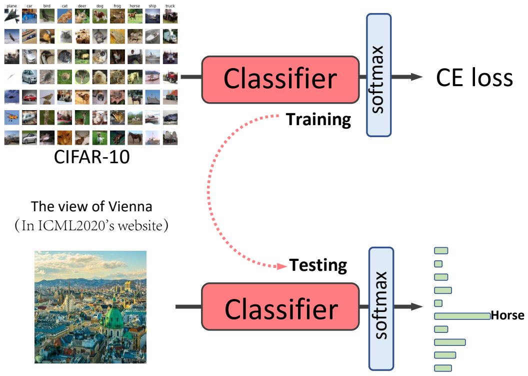

Figure 1. A network trained on CIFAR-10 with ten-class softmax output will predict the resized 32x32x3 view of Viena in ICML 2020's website (OOD sample w.r.t. CIFAR-10) as the horse with high confidence.

Being overconfident on out-of-distribution (OOD) inputs has raised concerns about the safety of artificial intelligence (AI) systems. Recent research efforts try to address these concerns by developing models that can identify anomalous inputs, $i.e.$, OOD samples (Amodei et al., 2016). Formally, the OOD detection problem can be formulated as a binary classification problem where the objective is to decide whether a test sample is from the training distribution ($i.e.$, in-distribution, ID) or from a different distribution ($i.e.$, OOD).

[*]Equal contribution [1]MILA [2]Harvard [3]Carnegie Mellon [4]UIUC [5]Salesforce Research. Correspondence to: Tong Che <tong.che@umontreal.ca>, Xiaofeng Liu <liuxiaofengcmu@gmail.com>.

Submission to the 37^{th} *International Conference on Machine Learning*, Vienna, Austria, PMLR 108, 2020. Copyright 2020 by the author(s).

	Hendrycks & Gimpel	Liang et al.	Devries & Taylor	Vyas et al.	Lee et al.	Choi et al.	Hendrycks
1	√	×	×	×	×	×	×
2	√	×	×	×	√	×	×
3	×	×	×	×	√	×	×
4	×	×	×	×	×	×	×

Table 1. Summary comparison of the characteristics of the recent related methods.

In this paper, we propose to verify the predictions of deep discriminative models by using deep generative models that try to generate the input conditioned on the label selected by the discriminative model. We call this concept ”deep verifier”. The high-level idea is simple: we train an inverse verification model $p(x|y)$ on the training data pairs (x, y). Intuitively speaking, for an input-output pair (x, y) with y picked by the predictive model, we verify whether the input x is consistent with y, by estimating if $p(x|y)$ is larger than a threshold. We design a density estimator of $p(x|y)$ using modified conditional VAEs. To ensure that the class code y is not ignored as a conditioning variable, we impose a disentanglement constraint based on minimizing mutual information between latent variable z and the label y. Although many different kinds of density estimators can be used in theory, we argue that the design of our model is robust to OOD samples and adversarial attacks, due to the use of latent variables with explicit and accurate density estimation.

Compared with previous approaches of OOD detection, our proposed method has four main advantages:

1. The verifier is trained independently of OOD distributions. Users do not need to figure out OOD samples before deployment of the system.

2. The verifier only needs to be trained once. No need to retrain the verifier for a new classifier.

3. The verifier can detect ordinary OOD samples and malicious adversarial attacks in a unified manner.

4. The framework is very general, so that it applies to structured prediction problems as well, such as image captioning.

We summarize the comparison of thesse four advantages with previous methods in Table 1.

The proposed solution achieves the state-of-the-art performance for detecting either OOD or adversarial samples in all tested classification scenarios, and can be generalized well for structured prediction tasks ($e.g.$, image caption). In Sec 3.4, we analysed why DVN is useful for both OOD and Adversarial examples.

2. Related Work

Detecting the OOD samples in a low-dimensional space using traditional non-parametric density estimation, nearest neighbor and clustering analysis have been well-studied (Pimentel et al., 2014). However, they are usually unreliable in high-dimensional complex spaces, $e.g.$, of images (Liang et al., 2018).

OOD detection with deep neural networks has recently been an active research topic. (Hendrycks & Gimpel, 2017) found that trained DNNs usually have higher maximum softmax output for in-distribution examples than anomalous one. A possible improvement of this baseline is to consider both the in-distribution and out-of-distribution training samples during training (Hendrycks, 2019). However, enumerating all possible OOD distributions before deployment is usually not possible.

(Liang et al., 2018) proposed that the difference between maximum probabilities in softmax distributions on ID/OOD samples can be made more significant by using adversarial perturbation pre-processing during training. (Devries & Taylor, 2018) augmented the classifier with a confidence estimation branch, and adjusted the objective using the predicted confidence score for training. (Lee et al., 2018a) trained a classifier simultaneously with a GAN, with an additional objective to encourage low confidence on generated samples. (Hendrycks, 2019) proposed to use real OOD samples instead of generated ones to train the detector. (Vyas et al., 2018) labels a part of training data as OOD samples to train the classifier, and they dynamically change the partition of ID and OOD samples. These improvements based on (Hendrycks & Gimpel, 2017) typically needs retrain a classifier with modified structures or optimization objectives. This can make it hard to maintain the original accuracy and is computationally expensive.

Recently, (Lee et al., 2018b) proposed a new framework of anomaly detection by first obtaining the class conditional Gaussian distribution using Gaussian discriminative analysis, and then define confidence score using the Mahalanobis distance between the sample and the closest class-conditional Gaussian distribution. By modeling each class of in-distribution samples independently, it showed remarkable results for OOD and adversarial attacks detection. Note however that their methods also needs the input pre-processing and model change. Besides, many previous methods (Liang et al., 2018; Vyas et al., 2018; Lee et al., 2018b) need OOD samples for hyper-parameter ($e.g.$, threshold for verification) selection, and these are usually not accessible in the real world.

Recently, (Choi et al., 2018) proposed an unsupervised OOD detector by estimating the Watanabe-Akaike Information Criterion, which is in turn estimated using an ensemble

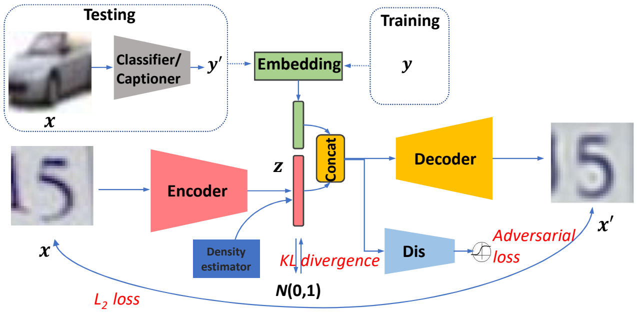

Figure 2. The architecture of our proposed Deep Verifier Network (DVN). We use ground-truth label y of training example x during training while using the trained model prediction y' of testing image during testing.

of generative models. The goal of our model is different from WAIC in that rather than just detecting OOD samples, DVNs aim to verify the predictions of a supervised predictive model, i.e., estimating $p(x|y)$ not just $p(x)$. We argue that model $p(x|y)$ is usually easier than directly model $p(x)$ as the former distribution contains less modes. Another motivation for modelling $p(x|y)$ instead of $p(x)$ is that for an adversarial attack and its classifier prediction (x', y'), it is usually much easier to verify x' is not in $p(x|y')$ than to verify x' is not in $p(x)$.

3. Methodology

This paper targets the problem of verification of deep predictive models, as follows. Let $x \in \mathcal{X}$ be an input and $y \in \mathcal{Y}$ be the ground-truth value to be predicted. The in-distribution examples are sampled from the joint data-generating distribution $p_{\text{in}}(x,y) = p_{\text{in}}(y|x)p_{\text{in}}(x)$. We propose to reverse the order of the prediction process of $p(y|x)$ and try to compute the conditional probability $p(x|y)$, where y is the label value guessed by the classifier to be verified (e.g., the one with the highest probability according to the deep network). We evaluate whether the input x is consistent with that y.

The predictive model to be verified $p_\theta(y|x)$ is trained on a dataset set drawn from the $p_{\text{in}}(x,y)$, and may encounter samples from both $p_{\text{in}}(x,y)$ and $p_{\text{out}}(x,y)$ (*i.e.*, out-of-distribution or adversarial samples) at test time. Note there is some subtle difference between OOD (unlikely under $p_{\text{in}}(x)$) and adversarial examples (unlikely under the ground truth joint, but with high $p_{\text{in}}(x)$, especially if a small amount of noise is allowed).

Our goal is to verify if the pair (x,y) for y guessed by the predictive model given x is consistent with $p_{\text{in}}(x,y)$. We train a verifier network $q_\phi(x|y)$ as an approximation to the inverse posterior distribution $p(x|y)$. Modelling $p(x|y)$ instead of $p(x)$ as a verification has many advantages: (1) Usually $p(x)$ is much more diverse than the conditional distribution $p(x|y)$, so modelling $p(x|y)$ is much easier than modelling $p(x)$. (2) Modelling $p(x|y)$ allows us to provide a unified framework for verifying OODs, adversarial examples, and mis-classifications of the classifier.

3.1. Basic Model

Our basic model is a conditional variational auto-encoder shown in Fig. 2. The model is composed of two deep neural networks, a stochastic encoder $p_\phi(z|x)$ which takes input x to predict a latent variable z and a decoder $p(x|z,y)$ which takes both latent variable z and the label y to reconstruct x. One problem with training of conditional variational auto-encoders is that the decoder can ignore the effect of input label y, passing all information through the continuous latent variable z. This is not desirable as we want to use the decoder to model the conditional likelihood $p(x|y)$, not $p(x)$. Hence in this paper, we train the encoder so that it outputs a z which is approximately independent of y. The encoder and decoder are thus jointly trained to maximize the evidence lower bound (ELBO):

$$\log p(x|y) \geq \mathbb{E}_{q(z|x)}[\log p(x|z,y)] - \text{KL}(q(z|x)||p(z)) \tag{1}$$

The equality holds if and only if $q(z|x) = p(z|x,y)$, where $p(z|x,y)$ is the ground truth posterior. We note that the conditional GAN is not applicable here since its objective does not optimize the likelihood.

3.2. Disentanglement constraints for anomaly detection

To achieve this independence, we propose to add a disentanglement penalty to minimize the mutual information between z and y. Namely, besides the ELBO loss, we also minimize the mutual information estimator $\hat{I}(z,y)$ together with the loss, yielding:

$$L = -\mathbb{E}_{q(z|x)}[\text{log}p(x|z,y) + \lambda\hat{I}(y,z)] + \text{KL}(q(z|x)||p(z)) \tag{2}$$

In this paper, we use deep Infomax (Hjelm et al., 2018) as the proxy for minimizing the mutual information (MI) between z and y. The mutual information estimator is defined as:

$$\hat{I}(z,y) = \mathbb{E}_{p(y,z)}[-s_+(-T(y,z))] - \mathbb{E}_{p(y)p(z)}[s_+(T(z,y))] \tag{3}$$

where s_+ is the softplus function and $T(y,z)$ is a discriminator network. Just like GAN discriminators, T is trained to maximize $\hat{I}(y,z)$, in order to get a better lower-bound estimation of the (JS-version) mutual information, while L (and in particular the encoder and decoder) is optimized (considering T fixed) to minimize $\hat{I}(y,z)$.

3.3. Measuring the likelihood as anomaly score

Our anomaly verification criterion is based on estimating the log-likelihood $\log p(x|y)$ for test samples. Importance sampling is a possible solution to provide an unbiased estimate of $p(x|y)$ when we have a VAE. Following IWAE (Burda et al., 2015), the k-sample importance weighting estimate of the log-likelihood is a lower bound of the ground truth likelihood $\mathcal{L}(x|y) = \mathbb{E}_{x\sim p(\cdot|y)}[\log p(x|y)]$:

$$\mathcal{L}_k(x|y) = \mathbb{E}_{z_1,\ldots,z_k\sim q(z|x)}[\log \frac{1}{k}\sum_{i=1}^{k}\frac{q(x,z_i|y)}{q(z_i|x)}]. \tag{4}$$

where $q(z)$ is a corrected density described below. We use the fact that $\mathcal{L}_k(x|y) \to \mathcal{L}(x|y)$ as $k \to \infty$ to estimate the likelihood. As will be discussed below, we want the decoder $q(x|z,y)$ be evaluated on the same input distribution for z as it is trained, which is not exactly the original Gaussian prior $p(z)$, so we will form a refined estimator of the prior, denoted $p*(z)$. The quantities $\mathcal{L}_k(x|y)$ form a monotonic series of lower bounds of the exact log-likelihood $\log p(x|y)$, with $\mathcal{L}_1 \leq \mathcal{L}_2 \leq \cdots \mathcal{L}_k \leq \log p(x|y))$. They have the property that when $k \to \infty$, $\mathcal{L}_k \to \log p(x|y)$. In our experiments we chose $k = 100$ for a good approximation of the exact likelihood,

In our algorithm, the distribution of z actually fed into decoder $p(x|z,y)$ during training is $q(z) = \int q(z|x)p_d(x)$. However, this distribution $q(z)$ can be drastically different from the Gaussian prior $p(z)$. So instead of using the Gaussian $p(z)$ as a prior for the decoder network in Eq. 4, we use $q(z)$ and estimate the corrected likelihood of x under this directed generative model, as $p(x,z|y) = q(z)p(x|z,y)$. In order to estimate the density of $q(z)$, we propose to train an additional discriminator D_z to distinguish $p(z)$ and $q(z)$. D_z is trained to discriminate the real distribution of latent variable $q(z) = \int p_d(x)e(z|x)dx$ ($p_d(x)$ is the data distribution of x, $e(z|x)$ is the encoder network) and Gaussian prior distribution $p(z)$, with ordinary GAN loss. Both $q(z)$ and $p(z)$ are easy to sample, so a discriminator is easy to train with the samples. In the GAN, the optimal discriminator D_z (Goodfellow, 2016) can be

$$D_z = \frac{p(z)}{p(z) + q(z)} \tag{5}$$

After D_z is trained (in theory optimally) and since $p(z)$ is known (*i.e.*, Gaussian), we can estimate $q(z) = \frac{1-D_z(z)}{D_z(z)}p(z)$.

We classify a sample x as an OOD sample if the log-likelihood is below the threshold δ and the x is an in-distribution sample, otherwise.

$$x \in \begin{cases} \text{in-distribution (ID),} & \text{if } L_k \geq \delta \\ \text{out-of-distribution (OOD),} & \text{otherwise} \end{cases} \tag{6}$$

We set δ to the threshold corresponding to 95% true positive rate (TPR), where the TPR refer to the probability of in-distribution validation samples are correctly verified as the in-distribution. Therefore, the threshold selection in our model is only tuned on in-distribution validation datasets, while most of previous methods also need the OOD samples for hyper-parameter validation (Liang et al., 2018; Lee et al., 2018b). We note that the distribution of OOD samples is usually not accessible before the system deployment.

3.4. Theoretical Justification

The loss function we optimize can be written as:

$$\begin{aligned} L = & L_1 + \lambda L_2 = \mathbb{E}_{x,y\sim p_d}[-\mathbb{E}_{q(z|x)}[\log p(x|z,y)] \\ & + \mathrm{KL}(q(z|x)||p(z)) + \lambda\mathbb{E}_{q(z|x)}[\hat{I}(y,z)]] \end{aligned} \tag{7}$$

where $p(x|z,y)$ is the decoder we are training. In this section, we use the following convention. Symbol p means probability distributions induced by the decoder, and symbol q means probability distributions induced by the encoder. Also denote p_d for real data distributions. Specifically, we define joint distribution $q(z,x,y) = q(z|x)p_d(x,y)$[1].

We have the following theorem that justifies the two parts of the above loss.

Theorem 1

(i) $-L_1$ *is a variational lower bound of* $\mathbb{E}_{x,y\sim p_d}[\log p(x|y)]$*.The bound is tight when* q *is expressive enough and* z, y *are conditionally independent given* x*.*

(ii) If we have $I(y,z) = 0$*, where* $(y,z) \sim \mathbb{E}_{x\sim p_d}[p_d(y|x)q(z|x)]$ *(namely* $L_2 \approx 0$*), and assume that the decoder is perfect in sense that* $p(x|y,z) = q(x|y,z)$*, then we have our evaluation metric* $\mathbb{E}_{z\sim q(z)}[p(x|y,z)] = p_d(x|y)$*. Namely, if* $I(y,z) = 0$*, and the decoder is trained to optimal, , then no matter what the encoder looks like, the likelihood estimator we are using is* $\mathbb{E}_{z\sim q(z)}[p(x|y,z)]$ *is equal to the groundtruth likelihood. This justifies why we need loss* L_2*.*

Proof 3.1 *For* (i)*, we have:*

$$\begin{aligned} -L_1 &= \mathbb{E}_{x,y\sim p_d}[\mathbb{E}_{q(z|x)}[\log p(x|z,y)] - KL(q(z|x)||p(z))] \\ &= \mathbb{E}_{x,y\sim p_d}[\mathbb{E}_{q(z|x)}[\log p(x,z|y) - \log p(z|y)] \\ &\qquad - KL(q(z|x)||p(z))] \\ &= \mathbb{E}_{x,y\sim p_d}[\mathbb{E}_{q(z|x)}[\log p(x|y)] - KL(q(z|x)||p(z|x,y))] \\ &\leq \mathbb{E}_{x,y\sim p_d}[\log p(x|y)] \end{aligned} \tag{8}$$

[1] In this paper we assume $q(z|x) = q(z|x,y)$, the motivation is during test time, y may be a wrong label, we don't want it to confuse the encoder. See detailed ablation in our Appendix.

The bound is tight if $\mathbb{E}[KL(q(z|x)||p(z|x,y))] = 0$*, which is equivalent to* $\mathbb{E}[KL(q(z|x)||p(z|x))] = 0$ *if* z, y *are conditionally independent give* x.

For (ii), *we have that if* y, z *are independent, namely* $q(y,z) = p_d(y)q(z)$,*we have:* $q(x|y,z) = q(x,y,z)/q(y,z) = q(z|x)p_d(x,y)/p_d(y)q(z) = q(z|x)p_d(x|y)/q(z)$. *So we have:*

$$\begin{aligned}\mathbb{E}_{z\sim q(z)}[p(x|y,z)] &= \mathbb{E}_{z\sim q(z)}[q(x|y,z)] \\ &= \mathbb{E}_{z\sim q(z|x)}[p_d(x|y)] = p_d(x|y)\end{aligned}$$

3.5. Intuitive Justifications

We now present an intuitive justification for the above algorithm. First, consider the following part of our training loss:

$$L_1 = \mathbb{E}_{q(z|x)}[\mathrm{log}p(x|z,y)] - \mathrm{KL}(q(z|x)||p(z)) \qquad (9)$$

It is well known that deep neural networks can generalize well for in-distribution samples, but their behavior out-of-distribution is less clear. Suppose x is an out-of-distribution sample, with y be the corresponding output of the classifier. Then the behavior of the stochastic encoder $q(z|x)$ is undefined. We denote $q(z) = \int q(z|x)p_d(x)$ the distribution to train $q(x|y,z)$. There are two cases: (1) $q(z|x)$ maps x to z with low density in $q(z)$. This case can be easily detected because $q(z)$ is easily computable. In this case the second term in Eq. 9 is a large negative number. (2) $q(z|x)$ maps x to z with high density in $q(z)$. Then since we train the decoder network with the input distribution $q(z)$ and because y and z are approximately independent, so (z,y) looks like an in-distribution input for decoder $p(x|z,y)$. Thus $p(x|y,z)$ should map to some in-distribution x' with class label y. Since input x is an OOD sample and reconstruction x' is an in-distribution sample, the reconstruction has to be bad. In this case, the first term in Eq. 9 is a large negative number. So in both cases, the log-likelihood score L_k derived from our model should be a large negative number. This is why our model is robust to both adversarial and OOD samples.

3.6. Can We Replace VAEs with Other Density Estimators?

In theory, we can use any other density estimator besides our modified conditional VAE (such as auto-regressive models and flow-based models) to estimate $p(x|y)$. However, our experiments and previous observations suggest that these other models may have drawbacks that would make them less suitable for this task. The comparison with the DVN that based on PixelCNN (Van den Oord et al., 2016) and Glow (Kingma & Dhariwal, 2018) are compared in Table. 2, which consistently inferior than our VAE solution. Auto-regressive models are quite slow and may ignore the conditioning label y (Bowman et al., 2015). Flow-based models were found to be less robust to adversarial examples, assigning higher likelihood on OOD samples than in-distribution samples (Nalisnick et al., 2018). We have intuitively explained in Sec. 3.5 why our modified cVAE based model does not suffer from the same problem as flow-based models, thanks to our disentanglement regularizer, which relies on the existence of a latent space.

4. Experimental results

In this section, we demonstrate the effectiveness of the proposed DVN on several classification benchmarks, and show its potential for the image captioning task. We choose the DenseNet (Huang et al., 2017) and ResNet (He et al., 2016) architectures as the backbones of our experiments.

For evaluation, we measure the following metrics as indicators of effectiveness in distinguishing in- vs out-of distribution samples. Following the definitions in previous work, we denote the in-distribution images as positive samples, while the OOD ones as the negative samples.

• True Negative Rate or False Positive Rate at **95%** True Positive Rate (i.e., **TNR@TPR95%** or **FPR@TPR95%**). Let TP, TN, FP, and FN denote true positive, true negative, false positive and false negative, respectively. We measure TNR = TN / (FP+TN) or FPR = FP / (FP+TN), when TPR = TP / (TP+FN) is 95%.

• Area under the receiver operating characteristic curve (**AUROC**). The ROC curve is a graph plotting TPR against the false positive rate = FP / (FP+TN) by varying a threshold. The area under ROC is the probability that an in-distribution sample has a higher certainty score than an OOD sample.

• Area under the precision-recall curve (**AUPR**). The PR curve is a graph plotting the precision = TP / (TP+FP) against recall = TP / (TP+FN) by varying the threshold.

• **Verification accuracy** is defined by $1 - min_\delta\{p_{in}(\mathcal{L} \le \delta)p(x \in p_{in}) + p_{out}(\mathcal{L} > \delta)p(x \in p_{out})\}$, where $\mathcal{L}$ is the predicted certainty score, $p(x \in p_{in})$ or $p(x \in p_{out})$ is the probability of positive or negative samples in the test set. It corresponds to the maximum classification probability over all possible thresholds.

We note that AUROC, AUPR and verification accuracy are threshold (δ)-independent evaluation metrics.

4.1. Detecting out-of-distribution samples for classification

Datasets. The Street View Housing Numbers (**SVHN**) dataset (Netzer et al., 2011) consists of color images depicting house numbers, which range from 0 to 9. Images have a resolution of 32×32. For our tests, we use the official training set split which contains 73,257 images, and the test set

In-Dist	OOD	Validation on OOD samples			Validation on adversarial samples		
		TNR@TPR 95%	AUROC	Verification acc.	TNR@TPR 95%	AUROC	Verification acc.
		ODIN / SUF / **Our DVN** / Glow based DVN / Pixel CNN based DVN			ODIN / SUF /**Our DVN** / Glow based DVN / Pixel CNN based DVN		
CIFAR-10	SVHN	86.2/90.8/**92.4**/91.1/90.7	95.5/98.1/**99.0**/98.2/98.0	91.4/93.9/**95.1**/93.7/93.9	70.5/89.6/**91.2**/89.8/90.0	92.8/97.6/**98.1**/97.5/97.6	86.5/92.6/**94.2**/93.1/93.4
DenseNet	T-ImageN	92.4/95.0/**96.2**/95.1/94.8	98.5/98.8/**99.0**/98.4/98.2	93.9/95.0/**97.3**/96.4/96.6	87.1/94.9/**95.6**/94.7/94.3	97.2/98.8/**99.1**/98.8/98.6	92.1/95.0/**97.4**/96.5/96.2
	LSUN	96.2/97.2/**98.6**/97.5/97.3	99.2/**99.3**/**99.3**/98.9/98.9	95.7/96.3/**96.8**/96.2/96.0	92.9/97.2/**97.9**/97.2/97.3	98.5/99.2/**99.3**/98.7/98.8	94.3/96.2/**97.5**/96.6/96.3
CIFAR-100	SVHN	70.6/82.5/**85.2**/83.0/82.8	93.8/97.2/**97.3**/97.1/96.8	86.6/91.5/**93.4**/92.4/92.5	39.8/62.2/**70.5**/65.7/66.0	88.2/91.8/**92.2**/90.9/91.0	80.7/84.6/**86.3**/85.4/85.7
DenseNet	T-ImageN	42.6/86.6/**89.0**/86.4/86.5	85.2/**97.4**/**97.4**/96.8/95.6	77.0/92.2/**93.8**/91.8/92.0	43.2/87.2/**89.1**/88.5/88.5	85.3/97.0/**97.8**/96.9/96.4	77.2/91.8/**93.0**/92.3/92.0
	LSUN	41.2/91.4/**93.7**/92.5/93.1	85.5/98.0/**98.2**/97.6/97.5	77.1/93.9/**94.9**/93.0/93.2	42.1/91.4/**93.6**/91.8/92.0	85.7/97.9/**98.3**/97.9/97.8	77.3/93.8/**95.4**/94.2/94.6
SVHN	CIFAR-10	71.7/96.8/**97.4**/95.7/96.2	91.4/98.9/**99.2**/98.8/98.2	85.8/95.9/**96.5**/95.1/95.0	69.3/97.5/**97.8**/97.4/97.0	91.9/98.8/**99.1**/98.1/98.0	86.6/96.3/**97.4**/96.6/96.7
DenseNet	T-ImageN	84.1/99.9/**100**/98.3/98.0	95.1/**99.9**/**99.9**/98.5/98.4	90.4/98.9/**99.2**/98.0/97.7	79.8/**99.9**/**99.9**/96.4/98.3	94.8/99.8/**99.9**/96.7/97.1	90.2/98.9/**99.4**/97.6/97.7
	LSUN	81.1/**100**/**100**/98.7/98.5	94.5/**99.9**/**99.9**/97.9/98.2	89.2/99.3/**99.6**/98.8/98.4	77.1/**100**/**100**/98.2/98.5	94.1/99.9/**100**/96.8/96.5	89.1/99.2/**99.5**/97.2/98.1
CIFAR-10	SVHN	86.6/96.4/**98.4**/97.3/97.0	96.7/99.1/**99.2**/98.5/98.6	91.1/95.8/**97.3**/96.2/96.1	40.3/75.8/**78.5**/77.6/77.4	86.5/95.5/**96.1**/95.5/95.3	77.8/89.1/**92.2**/91.2/91.0
ResNet	T-ImageN	72.5/97.1/**98.0**/97.0/96.9	94.0/99.5/**99.6**/98.5/98.5	86.5/96.3/**96.9**/94.7/94.9	96.6/95.5/**97.1**/96.2/95.9	93.9/99.0/**99.2**/98.3/98.1	86.0/95.4/**96.3**/95.5/95.1
	LSUN	73.8/98.9/**99.0**/97.6/97.7	94.1/**99.7**/**99.7**/97.8/97.5	86.7/97.7/**97.9**/96.3/96.0	70.0/98.1/**98.9**/96.8/96.5	93.7/**99.5**/**99.5**/97.6/97.7	85.8/97.2/**98.0**/96.8/96.5
CIFAR-100	SVHN	62.7/91.9/**93.5**/92.5/92.6	93.9/98.4/**98.8**/98.3/98.0	88.0/93.7/**94.8**/92.9/93.2	12.2/41.9/**46.2**/42.4/43.5	72.0/84.4/**86.3**/84.7/84.2	67.7/76.5/**79.4**/77.5/77.4
ResNet	T-ImageN	49.2/90.9/**91.2**/90.6/90.3	87.6/98.2/**98.5**/98.0/97.7	80.1/93.3/**94.3**/93.0/93.1	33.5/70.3/**74.6**/72.2/71.8	83.6/87.9/**90.3**/86.6/86.5	75.9/84.6/**89.8**/85.2/85.8
	LSUN	45.6/90.9/**92.3**/89.5/88.8	85.6/98.2/**98.6**/96.7/97.0	78.3/93.5/**95.7**/93.9/93.7	31.6/56.6/**63.5**/60.2/60.1	81.9/82.3/**85.2**/82.9/82.8	74.6/79.7/**81.9**/80.0/78.9
SVHN	CIFAR-10	79.8/98.4/**99.4**/97.9/97.5	92.1/99.3/**99.9**/98.1/98.2	89.4/96.9/**97.5**/96.3/96.3	79.8/94.1/**94.5**/93.7/93.5	92.1/97.6/**98.7**/96.5/96.2	89.4/94.6/**94.8**/93.7/93.0
ResNet	T-ImageN	82.1/99.9/**100**/98.5/98.4	92.0/**99.9**/**99.9**/96.3/96.5	89.4/99.1/**99.2**/95.8/96.7	80.5/99.2/**99.7**/98.5/98.3	92.9/99.3/**99.5**/97.2/97.0	90.1/98.8/**99.3**/98.0/98.2
	LSUN	77.3/**99.9**/**99.9**/96.4/96.4	89.4/**99.9**/**99.9**/97.6/97.4	87.2/99.5/**100**/99.0/98.9	76.3/**99.9**/**99.9**/96.5/97.4	90.7/**99.9**/99.8/96.8/96.7	88.2/99.5/**99.8**/98.3/98.1

Table 2. OOD verification results of image classification under different validation setups. All metrics are percentages and the best results are bolded. The backbone classifier in SUF (Lee et al., 2018b) and our DVN is ResNet34 (He et al., 2016), while ODIN (Liang et al., 2018) use more powerful wide ResNet 40 with width 4 (Zagoruyko & Komodakis, 2016).

split, which has 26,032 images. The **CIFAR-10/100** dataset (Krizhevsky & Hinton, 2009) consists of 10/100 classes colour images. The training set has 50,000 images, while the test set has 10,000 images. The **TinyImageNet** dataset[2] is a subset of the ImageNet dataset (Deng et al., 2009). Its test set contains 10,000 images from 200 different classes. It contains the original images, downsampled to 32×32 pixels. The Large-scale Scene UNderstanding dataset (**LSUN**) (Yu et al., 2015) has a test set with 10,000 images from 10 different classes. The LSUN (crop) and LSUN (resize) are created in a similar downsampling manner to the TinyImageNet datasets. The **Uniform noise** and **Gaussian noise** dataset are with 10,000 samples respectively, which are generated by drawing each pixel in a 32×32 RGB image from an i.i.d uniform distribution of the range [0, 1] or an i.i.d Gaussian distribution with a mean of 0.5 and variance of 1 (Liang et al., 2018).

Setups. For fair comparisons, the backbones of the classifiers used here are the 100-layer DenseNet with growth rate 12 (Liang et al., 2018; Lee et al., 2018b) and 34-layer ResNet (Lee et al., 2018b). They are trained to classify the SVHN, CIFAR-10, CIFAR-100 and Tiny-ImageNet datasets, of which test set is regarded as the in-distribution dataset in our testing stage. The dataset different from its training dataset is considered as OOD. We use four convolution or deconvolution layers for the encoder and decoder structure, and z is a 128-dimension vector. The discriminator is a two-layer fully connected layer network with sigmoid output and binary cross-entropy loss. The hyper-parameters in previous methods (Liang et al., 2018; Lee et al., 2018b) need to be tuned on a validation set with 1,000 images from each in-distribution and OOD pair. Noticing that the threshold of the DVN is tuned on in-distribution only. This corresponds to a more realistic scenario, since the OOD nature of real-world applications is usually uncontrollable.

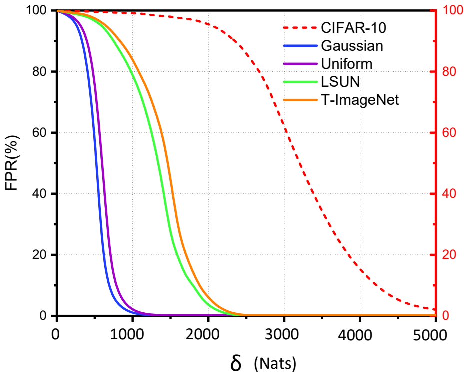

Figure 3. FPR (for OOD) and TPR (for ID) under different δ when using CIFAR-10 as the in-distribution dataset, and use Tiny-ImageNet(resize), LSUN and Gaussian/Uniform noise as OOD. CIFAR-10 only applicable to the TPR which use the dashed red line and indicated by the right axis while the other OOD datasets use the left FPR axis.

Effects of the threshold and performance across datasets. How the hyper-parameters ($e.g.$, δ) generalize across different OOD datasets is a challenging aspect of the system deployment. Most of the previous methods require a small set of OOD samples, with δ calibrated by evaluating the verification error at different values of δ. However, the more realistic scenario is that we do not have access to the OOD examples in the testing stage.

A promising trend is improving the performance on an unknown OOD when using the model tuned on a similar OOD (Liang et al., 2018; Lee et al., 2018b). We argue that our

[2] https://tiny-imagenet.herokuapp.com/

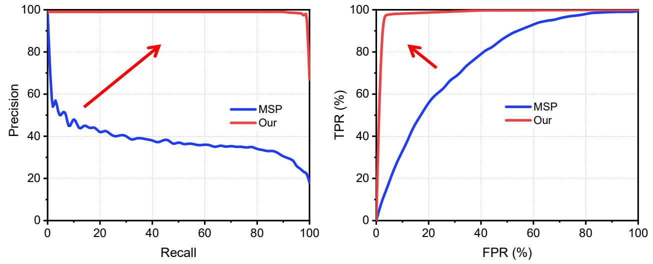

Figure 4. Comparison with baseline MSP (Hendrycks & Gimpel, 2017) using DenseNet, with Tiny-ImageNet as in-distribution and LSUN as OOD.

	CIFAR-10	CIFAR-100
ODIN/SUF	4.81	22.37
DenseNet/DVN	**4.51**	**22.27**

Table 3. Test error rate of classification on CIFAR-10/100 using DenseNet as backbone. Our DVN does not re-train or modify the structure of the original trained classifier.

DVN is essentially free from such worries, since it does not need any OOD sample in the validation.

To investigate how the threshold affects the FPR and TPR, Fig. 3 shows their relationship when training on CIFAR-10 and different OOD datasets are used in the test stage, with a DenseNet backbone. Note that the TPR (red axis) is used for in-distribution dataset CIFAR-10 (red dashed line), while FPR is used for OODs. We can observe that the threshold corresponding to 95% TPR can produce small FPRs on all OOD datasets. When the OOD images are sampled from some simple distributions ($e.g.$, Gaussian or Uniform), the available window of threshold δ can be larger.

Comparison with SOTA. The main results are summarised in Table. 2. For each in&out-of-distribution pair, we report the performance of ODIN (Liang et al., 2018), SUF (Lee et al., 2018b) and our DVN. Notably, DVN consistently outperforms the previous methods and achieves a new state-of-the-art. As shown in Table. 3, the pre-processing and model change in ODIN and SUF increase the error rate of the original classifier for the in-distribution test, while DVN does not affect the classification accuracy on the accepted in-distribution datasets (i.e., CIFAR-10 or 100), when the OOD do not present.

Considering the technical route of DVN is essentially different from ODIN and SUF, we compare it with the baseline, maximum softmax probability (MSP) (Hendrycks & Gimpel, 2017), w.r.t. ROC and PR in Fig. 4. DVN shares some nice properties of MSP, $e.g.$, fixed classifier and single forward pass at the test stage, while DVN outperforms MSP by a large margin.

Disentangle	TNR@TPR95%	AUROC
√	**98.4**	**99.2**
-	62.6	84.7

Table 4. The performance of DVN w/o disentanglement of y from z with ResNet backbone, and using CIFAR-10/SVHN as in-distribution/OOD, respectively.

$q(z)$	TNR@TPR95%	AUROC
√	**98.4**	**99.2**
-	95.3	96.7

Table 5. The performance of DVN w/o replace $p(z)$ with $q(z)$. We use ResNet backbone, and choose CIFAR-10/SVHN as in-distribution/OOD.

Ablation study. We also compared with the DVN that use pixel CNN or Glow rather than VAE as shown in Table. 2. The pixelCNN/Glow-based DVN is consistently inferior than our solution.

Disentangling y from z is critical to our model. Table 4 validates the contribution of this manipulation w.r.t. both threshold dependent and independent metrics. One can see that the DVN with disentanglement significantly outperforms its counterparts without disentanglement. This also implies the DVN has successfully learned to sufficiently minimize the mutual information between z and y to circumvent the challenge of conditioning x on y.

Since modeling $p(x|y)$ is the core of DVN, we cannot remove y. Here, we give another ablation study that without modify $p(z)$ with $q(z)$. are shown in . As shown in Table 5, there is a large margin between the DVN with or without disentanglement w.r.t. TNR@TPR95 and AUROC. The results demonstrate that disentangle y from z is of essential important for our framework.

4.2. Detecting adversarial examples

Deal with popular adversarial attackers. To detect adversarial examples, we train our DenseNet and ResNet-based classification network and DVN using the training set of CIFAR-10, CIFAR-100 or SVHN datasets, and their corresponding test sets are used as the positive samples for the test. Following the setting in (Lee et al., 2018b), we applied several attack methods to generate the negative samples, such as basic iterative method (BIM) (Kurakin et al., 2016), Deepfool (Moosavi-Dezfooli et al., 2016), and Carlini-Wangner (CW) (Carlini & Wagner, 2017). The network structures are the same as for OOD verification.

We compare the DVN with the strategies in KD+PU (Feinman et al., 2017), LID (Ma et al., 2018), SUF (Lee et al., 2018b) in Table 6, and show that the DVN can achieve the

	Dataset	Method	Negative Sample	Pre-proce	Deep Fool	CW	BIM
DenseNet	CIFAR -10	KD+PU	FGSM	-	68.34	53.21	3.10
		LID	FGSM	-	70.86	71.50	94.55
		SUF	FGSM	Yes	87.95	83.42	**99.51**
		Our	-	-	**90.14**	**86.38**	99.42
	CIFAR -100	KD+PU	FGSM	-	65.30	58.08	66.86
		LID	FGSM	-	69.68	72.36	68.62
		SUF	FGSM	Yes	75.63	86.20	98.27
		Our	-	-	**80.01**	**88.55**	**99.04**
	SVHN	KD+PU	FGSM	-	84.38	82.94	83.28
		LID	FGSM	-	80.14	85.09	92.21
		SUF	FGSM	Yes	93.47	96.95	**99.12**
		Our	-	-	**94.14**	**97.35**	**99.12**
DenseNet	CIFAR -10	KD+PU	FGSM	-	76.80	56.30	16.16
		LID	FGSM	-	71.86	77.53	95.38
		SUF	FGSM	Yes	78.06	93.90	98.91
		Our	-	-	**82.45**	**95.51**	**99.07**
	CIFAR -100	KD+PU	FGSM	-	57.78	73.72	68.85
		LID	FGSM	-	63.15	75.03	55.82
		SUF	FGSM	Yes	81.95	90.96	96.38
		Our	-	-	**85.22**	**93.38**	**97.72**
	SVHN	KD+PU	FGSM	-	84.30	67.85	43.21
		LID	FGSM	-	67.28	76.58	84.88
		SUF	FGSM	Yes	72.20	86.73	95.39
		Our	-	-	**86.13**	**89.38**	**96.10**

Table 6. Comparison of AUROC (%) under different validation setups. The best results are bolded. We also compaired the use of negative sampe for training and input image pre-processing.

state-of-the-art performance in most cases w.r.t. AUROC. In the "detection of unknown attack setting", we can not access the adversarial examples of the test stage in the training or validation. Therefore, the previous works choose to use another attack generation method, $i.e.$, fast gradient sign method (FGSM) (Goodfellow et al., 2014), to construct a validation set of adversarial examples. In here, we do not need another attack method as a reference, since the threshold of the DVN is only related to the validation set of in-distribution samples. Moreover, the pre-processing and model change as in (Lee et al., 2018b) are not required in our proposed DVN.

The performance of dealing the adaptive attackers (be aware of deep verifier), spatially transformed adversarial attackers and unrestricted adversarial attackers are shown in the supplementary materials.

4.3. Detecting out-of-distribution samples for image captioning

For detecting OOD samples in the image captioning task, we choose Oxford-102 and CUB-200 as the in-distribution datasets. Oxford-102 contains 8,189 images of 102 classes of flower. CUB-200 contains 200 bird species with 11,788 images. Each of them has 10 descriptions that are provided by (Reed et al., 2016a). For these two datasets, we use 80% of the samples to train our caption generator, and the remaining 20% for testing in a cross-validation manner. The LSUN and Microsoft COCO datasets are used as our OOD dataset.

In-Dist	OOD	Validation on OOD samples		
		TNR@TPR 95%	AUROC	Verif acc.
Oxford	CUB	55.6	72.3	79.5
	LSUN	50.5	71.8	76.2
	COCO	40.3	74.4	73.3
CUB	Oxford	39.8	68.4	72.5
	LSUN	36.3	65.4	69.5
	COCO	35.4	60.7	71.0

Table 7. OOD verification results of image caption under different validation setups. We use CUB-200, LSUN and COCO as the OOD of Oxford-102, while using Oxford-102, LSUN and COCO as OOD of CUB-200.

The captioner used in here is a classical image caption model (Xu et al., 2015). We choose the generator of GAN-INT-CLS (Reed et al., 2016b) as our decoder's backbone, and replace its Normal distribution vector as the output of encoder z. A character level CNN-RNN model (Reed et al., 2016a) is used for the text embedding which produces the 1,024-dimension vector given the description, and then projected to a 128-dimension code c. We configure the encoder and decoder with four convolutional layers and the latent vector z is a 100-dimension vector. The input of the discriminator is the concatenation of z and c, which results in a 228-dimension vector. A two-layer fully connected network with sigmoid output unit is used as the discriminator. Table 7 summarizes the performance of DVN in image caption task and can be regarded as a powerful baseline.

5. Conclusion and Future Work

In this paper, we propose to enhance the performance of anomaly detection by verifying predictions of deep discriminative models using deep generative models. The idea is to train a conditional verifier network $q(x|y)$ as an approximation to the inverse posterior distribution. We propose Deep Verifier Networks (DVNs) which are based on a modified conditional variational auto-encoders with disentanglement constraints. We show our model is able to achieve state-of-the-art performance on benchmark OOD detection and adversarial example detection tasks.

For future work, it would be interesting to integrate DVNs to safe AI systems. For instance, ordinary image classifiers such as DenseNet have perfect accuracy for in-distribution queries, but their behaviors are undefined on adversarial queries. Robust image classifiers sacrifice some accuracy for robustness to adversarial examples. We can use DVNs to form a two-step prediction procedure: first using an ordinary classifier to get an initial prediction, then verify the prediction with DVN. If the prediction does not pass the verification, we switch to the robust image classifier. We believe our method would provide some advantages for safe AI systems.

References

Amodei, D., Olah, C., Steinhardt, J., Christiano, P., Schulman, J., and Mané, D. Concrete problems in ai safety. *arXiv preprint arXiv:1606.06565*, 2016.

Bowman, S. R., Vilnis, L., Vinyals, O., Dai, A. M., Jozefowicz, R., and Bengio, S. Generating sentences from a continuous space. *arXiv preprint arXiv:1511.06349*, 2015.

Burda, Y., Grosse, R., and Salakhutdinov, R. Importance weighted autoencoders. *arXiv preprint arXiv:1509.00519*, 2015.

Carlini, N. and Wagner, D. Adversarial examples are not easily detected: Bypassing ten detection methods. In *Proceedings of the 10th ACM Workshop on Artificial Intelligence and Security*, pp. 3–14. ACM, 2017.

Choi, H., Jang, E., and Alemi, A. A. Waic, but why? generative ensembles for robust anomaly detection. *arXiv preprint arXiv:1810.01392*, 2018.

Chung, J., Gulcehre, C., Cho, K., and Bengio, Y. Empirical evaluation of gated recurrent neural networks on sequence modeling. *arXiv preprint arXiv:1412.3555*, 2014.

Deng, J., Dong, W., Socher, R., Li, L.-J., Li, K., and Fei-Fei, L. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pp. 248–255. Ieee, 2009.

Devries, T. and Taylor, G. W. Learning confidence for out-of-distribution detection in neural networks. 2018.

Feinman, R., Curtin, R. R., Shintre, S., and Gardner, A. B. Detecting adversarial samples from artifacts. *arXiv preprint arXiv:1703.00410*, 2017.

Goodfellow, I. Nips 2016 tutorial: Generative adversarial networks. *arXiv preprint arXiv:1701.00160*, 2016.

Goodfellow, I., Bengio, Y., and Courville, A. *Deep learning*. MIT press, 2016.

Goodfellow, I. J., Shlens, J., and Szegedy, C. Explaining and harnessing adversarial examples. *arXiv preprint arXiv:1412.6572*, 2014.

Guo, C., Pleiss, G., Sun, Y., and Weinberger, K. Q. On calibration of modern neural networks. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pp. 1321–1330. JMLR. org, 2017.

He, K., Zhang, X., Ren, S., and Sun, J. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 770–778, 2016.

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

Hendrycks, Dan, M.-M. D. T. Deep anomaly detection with outlier exposure. *ICLR*, 2019.

Hjelm, R. D., Fedorov, A., Lavoie-Marchildon, S., Grewal, K., Trischler, A., and Bengio, Y. Learning deep representations by mutual information estimation and maximization. *arXiv preprint arXiv:1808.06670*, 2018.

Huang, G., Liu, Z., Van Der Maaten, L., and Weinberger, K. Q. Densely connected convolutional networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 4700–4708, 2017.

Kingma, D. P. and Dhariwal, P. Glow: Generative flow with invertible 1x1 convolutions. In *Advances in Neural Information Processing Systems*, pp. 10215–10224, 2018.

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

Krizhevsky, A., Sutskever, I., and Hinton, G. E. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pp. 1097–1105, 2012.

Kurakin, A., Goodfellow, I., and Bengio, S. Adversarial examples in the physical world. *arXiv preprint arXiv:1607.02533*, 2016.

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

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

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

Ma, X., Li, B., Wang, Y., Erfani, S. M., Wijewickrema, S., Schoenebeck, G., Song, D., Houle, M. E., and Bailey, J. Characterizing adversarial subspaces using local intrinsic dimensionality. *arXiv preprint arXiv:1801.02613*, 2018.

Moosavi-Dezfooli, S.-M., Fawzi, A., and Frossard, P. Deepfool: a simple and accurate method to fool deep neural networks. In *Proceedings of the IEEE conference on* *computer vision and pattern recognition*, pp. 2574–2582, 2016.

Nalisnick, E., Matsukawa, A., Teh, Y. W., Gorur, D., and Lakshminarayanan, B. Do deep generative models know what they don't know? *arXiv preprint arXiv:1810.09136*, 2018.

Netzer, Y., Wang, T., Coates, A., Bissacco, A., Wu, B., and Ng, A. Y. Reading digits in natural images with unsupervised feature learning. 2011.

Pimentel, M. A. F., Clifton, D. A., Lei, C., and Tarassenko, L. A review of novelty detection. *Signal Processing*, 99 (6):215–249, 2014.

Reed, S., Akata, Z., Lee, H., and Schiele, B. Learning deep representations of fine-grained visual descriptions. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 49–58, 2016a.

Reed, S., Akata, Z., Yan, X., Logeswaran, L., Schiele, B., and Lee, H. Generative adversarial text to image synthesis. *arXiv preprint arXiv:1605.05396*, 2016b.

Van den Oord, A., Kalchbrenner, N., Espeholt, L., Vinyals, O., Graves, A., et al. Conditional image generation with pixelcnn decoders. In *Advances in neural information processing systems*, pp. 4790–4798, 2016.

Vyas, A., Jammalamadaka, N., Zhu, X., Das, D., and Willke, T. L. Out-of-distribution detection using an ensemble of self supervised leave-out classifiers. *ECCV*, 2018.

Wu, Y., Schuster, M., Chen, Z., Le, Q. V., Norouzi, M., Macherey, W., Krikun, M., Cao, Y., Gao, Q., Macherey, K., et al. Google's neural machine translation system: Bridging the gap between human and machine translation. *arXiv preprint arXiv:1609.08144*, 2016.

Xu, K., Ba, J., Kiros, R., Cho, K., Courville, A., Salakhudinov, R., Zemel, R., and Bengio, Y. Show, attend and tell: Neural image caption generation with visual attention. In *International conference on machine learning*, pp. 2048–2057, 2015.

Yu, F., Seff, A., Zhang, Y., Song, S., Funkhouser, T., and Xiao, J. Lsun: Construction of a large-scale image dataset using deep learning with humans in the loop. *arXiv preprint arXiv:1506.03365*, 2015.

Zagoruyko, S. and Komodakis, N. Wide residual networks. *arXiv preprint arXiv:1605.07146*, 2016.