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

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

[1] Mila, Universit de Montral
[2] Harvard Medical School, Harvard University
[3] Carnegie Mellon University
[4] University of Illinois at Urbana-Champaign
[5] Salesforce Research
liuxiaofengcmu@gmail.com

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 the concise 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.

Introduction

Deep learning models provide state-of-the-art performance in various applications such as image classification (Krizhevsky, Sutskever, and Hinton 2012; Wang et al. 2020), caption generation (Xu et al. 2015), sequence modeling (Chung et al. 2014; Liu et al. 2018a) and machine translation (Xu et al. 2015). 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 and Gimpel 2016). 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 authentication, medical and self-driving systems (Liu et al. 2019b, 2017a, 2019a, 2020d,c,e, 2019c, 2020b,f, 2019f, 2018e,c; Han et al. 2020).

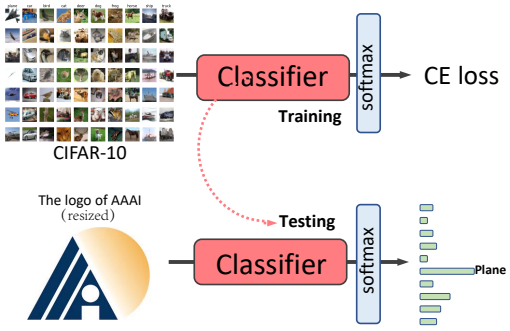

Figure 1: A network trained on CIFAR-10 will predict the resized 32×32×3 AAAI logo (OOD sample w.r.t. CIFAR-10) as the plane with high confidence.

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).

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 ver-

*Corresponding author. †Contribute equally

	Hendrycks 2016	Liang 2018	Devries 2018	Vyas 2018	Lee 2018	choi 2018	Hendrycks 2019
1	√	-	-	-	-	-	-
2	√	-	-	-	√	-	-
3	-	-	-	-	√	-	-
4	-	-	-	-	-	-	-

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

ifier". 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 representation 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 and adversarial attacks, due to the use of latent variables with explicit and accurate density estimation.

Compared with previous approaches for OOD, our proposed method has 4 main advantages (as shown in Tab. 1):

• 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.

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.

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 spaces, e.g., images (Liang, Li, and Srikant 2018).

OOD detection with deep neural networks has recently been an active research topic. (Hendrycks and Gimpel 2016) 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, Mazeika, and Dietterich 2019). However, enumerating all possible OOD distributions before deployment is usually not possible.

(Liang, Li, and Srikant 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 and Taylor 2018) augmented the classifier with a confidence estimation branch, and adjusted the objective using the predicted confidence score for training. (Lee et al. 2018b) trained a classifier simultaneously with a GAN, with an additional objective to encourage low confidence on generated samples. (Hendrycks, Mazeika, and Dietterich 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 and Gimpel 2016) typically needs re-train a classifier with modified structures or optimization objectives. This can make it hard to maintain the original accuracy and is computationally expensive.

(Lee et al. 2018a) propose to obtain the class conditional Gaussian distribution, and then define confidence score using the Mahalanobis distance between the sample and the closest class-conditional Gaussian distribution. However, it also needs the input pre-processing and model change. Besides, many previous methods (Liang, Li, and Srikant 2018; Vyas et al. 2018; Lee et al. 2018a) need OOD samples for hyper-parameter ($e.g.$, threshold for verification) selection, and these are usually not accessible.

The main difference of our model with Bayesian NN based calibration models (Nalisnick et al. 2019; Ovadia et al. 2019; Yao et al. 2019; Guo et al. 2017) is that our model does not need to modify the training procedure of the classifier. Bayesian NNs are notoriously hard and computationally expensive to train, and they need to be carefully designed and the model itself needs to be modified, which seriously limits their applications to real-world problems.

Recently, (Choi, Jang, and Alemi 2018) proposed an unsupervised OOD detector by estimating the Watanabe-Akaike Information Criterion. 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 modeling $p(x|y)$ is usually easier than directly modeling $p(x)$ 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 $p(x)$. For an adversarial attack (x', y') modified from (x, y), y' suppose to be a different class from y. However, x' is visually very much like x, namely $|x' - x|_{L^1} < \epsilon$. Therefore, given a wrong class y', $p(x'|y')$ can be very easily verified to be small, since x' is very close to a image in class y while it should be very different from an image in class y'.

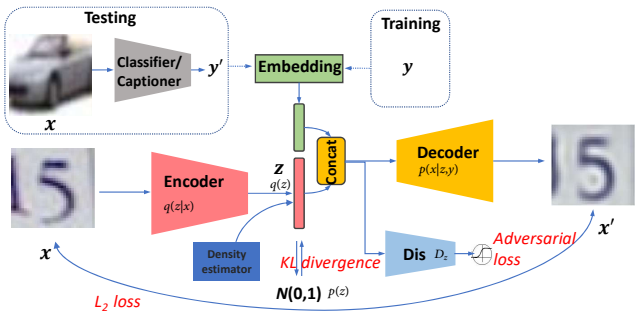

Figure 2: The architecture of our Deep Verifier Network (DVN). We use ground-truth label y of training example x in training while using the trained model prediction y' of testing image in testing.

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 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.

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 $q(z|x)$ 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):

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

The equality holds iff $q(z|x) = p(z|x, y)$, 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 (Liu et al. 2019e, 2020a; Liu 2020; He et al. 2020b,a).

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:

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

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 MI estimator is defined as:

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

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 $\hat{I}(y, z)$.

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, Grosse, and Salakhutdinov 2015), the k-sample importance weighting estimate of the log-likelihood is a lower bound of the ground truth likelihood $\mathcal{L}(x|y) = E_{x \sim p(\cdot|y)}[\log p(x|y)]$:

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

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 $p(x|z, y)$ 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 $\log p(x|y)$,

In-Dist	OOD	Validation on OOD samples			Validation on adversarial samples	
		TNR@TPR 95%	AUROC	Verification acc.	TNR@TPR 95%	AUROC
		ODIN / SUF / **Our DVN** / Glow based DVN / Pixel CNN based DVN			ODIN / SUF /**Our DVN** / Glow-DVN / Pixel-DVN	
CIFAR-10 DenseNet	SVHN	86.2/90.8/**96.4**/95.1/94.7	95.5/98.1/**99.0**/98.2/98.0	91.4/93.9/**95.1**/93.7/93.9	70.5/89.6/**95.2**/93.8/91.0	92.8/97.6/**98.1**/97.5/97.6
	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
	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
CIFAR-100 DenseNet	SVHN	70.6/82.5/**95.2**/93.0/92.8	93.8/97.2/**97.3**/97.1/96.8	86.6/91.5/**93.4**/92.4/92.5	39.8/62.2/**90.5**/85.7/86.0	88.2/91.8/**92.2**/90.9/91.0
	T-ImageN	42.6/86.6/**99.0**/96.4/96.5	85.2/97.4/**99.4**/96.8/95.6	77.0/92.2/**98.8**/95.8/95.0	43.2/87.2/**99.1**/98.5/98.5	85.3/97.0/**97.8**/96.9/96.4
	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/**99.9**/98.0/98.2	42.1/91.4/**98.6**/97.8/96.0	85.7/97.9/**98.3**/97.9/97.8
SVHN DenseNet	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
	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
	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
CIFAR-10 ResNet	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/**98.5**/97.6/97.4	86.5/95.5/**96.1**/95.5/95.3
	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
	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
CIFAR-100 ResNet	SVHN	62.7/91.9/**98.5**/96.5/96.6	93.9/98.4/**98.8**/98.3/98.0	88.0/93.7/**94.8**/92.9/93.2	12.2/41.9/**86.2**/82.4/83.5	72.0/84.4/**86.3**/84.7/84.2
	T-ImageN	49.2/90.9/**97.2**/95.6/95.3	87.6/98.2/**98.5**/98.0/97.7	80.1/93.3/**94.3**/93.0/93.1	33.5/70.3/**94.6**/92.2/91.8	83.6/87.9/**90.3**/86.6/86.5
	LSUN	45.6/90.9/**99.3**/98.5/98.8	85.6/98.2/**98.6**/96.7/97.0	78.3/93.5/**98.7**/96.9/95.7	31.6/56.6/**93.5**/90.2/90.1	81.9/82.3/**95.2**/92.9/92.8
SVHN ResNet	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
	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
	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

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 and our DVN is ResNet34 (He et al. 2016), while ODIN uses more powerful wide ResNet40 with width 4 (Zagoruyko et al. 2016).

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 $q(z) = \int q(z|x)p_d(x)dx$. 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 (Goodfellow et al. 2014; Liu et al. 2018b, 2017b, 2018d, 2019d). 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 can be $D_z = \frac{p(z)}{p(z)+q(z)}$ (Goodfellow 2016). 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.

$$x \in \begin{cases} \text{in-distribution (ID)}, & \text{if } L_k \geq \delta \\ \text{out-of-distribution (OOD)}, & \text{otherwise} \end{cases} \quad (5)$$

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. This differentiates our method with the other threshold based detector which need the OOD samples for hyper-parameter validation (Liang, Li, and Srikant 2018; Lee et al. 2018a). We note that the distribution of OOD samples is usually not accessible before the system deployment.

Theoretical Justification

The loss function we optimize can be written as:

$$L = L_1 + \lambda L_2 = E_{x,y\sim p_d}[-E_{q(z|x)}[\log p(x|z, y)] \quad (6)$$

$$+ \mathrm{KL}(q(z|x)||p(z)) + \lambda E_{q(z|x)}[\hat{I}(y, z)]] \quad (7)$$

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. (i) $-L_1$ is a variational lower bound of $E_{x,y\sim p_d}[\log p(x|y)]$.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 $(y, z) \sim E_{x\sim p_d}[p_d(y|x)q(z|x)]$ (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 $E_{z\sim q(z)}[p(x|y, z)] = p_d(x|y)$. 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 $E_{z\sim q(z)}[p(x|y, z)]$ is equal to the groundtruth likelihood.

This justifies why we need L_2 loss. Note that even with an encoder mapping everything to zero, the claim $E_{z\sim q(z)}[\log p(x|y, z)]$ still equals to the ground truth likelihood. In this case, $\log p(x|y, z) = \log p(x|y)$ and is a constant with respect to z.

[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.

Intuitive Justifications

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

$$L_1 = -E_{q(z|x)}[\log p(x|z,y)] + \mathrm{KL}(q(z|x)||p(z)) \quad (8)$$

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. 8 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. 8 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.

Replacing 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 is based on PixelCNN (Van den Oord et al. 2016) and Glow (Kingma and Dhariwal 2018) are compared in Tab. 2, which is 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 last subsection about 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.

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 True Negative Rate or False Positive Rate at 95% True Positive Rate (i.e., TNR@TPR95% or FPR@TPR95%), Area under the receiver operating characteristic curve (AUROC), Area under the precision-recall curve (AUPR) and Verification accuracy. We detailed these metrics in Supplementary Materials.

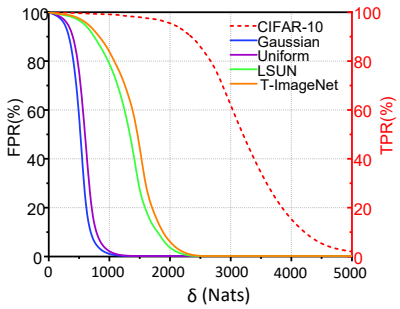

Figure 3: FPR (for OOD) and TPR (for ID) under different δ when using CIFAR-10 as the in-distribution dataset, and use Tiny-ImageNet(resize), LSUN and Gaussian/Uniform noise as OOD. CIFAR-10 only applicable to the TPR which use the dashed red line and indicated by the right axis while the other OOD datasets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

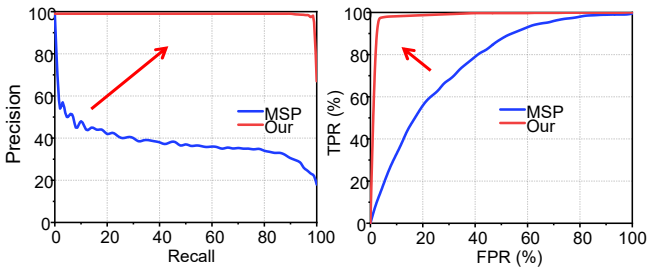

Figure 4: Comparison with baseline MSP (Hendrycks and Gimpel 2016) using DenseNet, with Tiny-ImageNet as in-distribution and LSUN as OOD.

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

Detecting OOD 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 split, which has 26,032 images. The **CIFAR-10/100** dataset (Krizhevsky and 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 dataset 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 $\mathcal{U}(0,1)$ or $\mathcal{N}(0.5,1)$ (Liang, Li, and Srikant 2018).

Setups. For fair comparisons, the backbones of the classifiers used here are the 100-layer DenseNet with growth

	Dataset	Method	Negative Sample	Pre-proce	Deep Fool	CW	BIM		Dataset	Method	Negative Sample	Pre-proce	Deep Fool	CW	BIM
DenseNet	CIFAR -10	KD+PU	FGSM	-	68.34	53.21	3.10	ResNet	CIFAR -10	KD+PU	FGSM	-	76.80	56.30	16.16
		LID	FGSM	-	70.86	71.50	94.55			LID	FGSM	-	71.86	77.53	95.38
		SUF	FGSM	Yes	87.95	83.42	99.51			SUF	FGSM	Yes	78.06	93.90	98.91
		Our	-	-	**96.14**	**96.38**	**99.82**			Our	-	-	**95.45**	**99.51**	**99.57**
	CIFAR -100	KD+PU	FGSM	-	65.30	58.08	66.86		CIFAR -100	KD+PU	FGSM	-	57.78	73.72	68.85
		LID	FGSM	-	69.68	72.36	68.62			LID	FGSM	-	63.15	75.03	55.82
		SUF	FGSM	Yes	75.63	86.20	98.27			SUF	FGSM	Yes	81.95	90.96	96.38
		Our	-	-	**97.01**	**98.55**	**99.94**			Our	-	-	**97.22**	**99.38**	**99.72**
	SVHN	KD+PU	FGSM	-	84.38	82.94	83.28		SVHN	KD+PU	FGSM	-	84.30	67.85	43.21
		LID	FGSM	-	80.14	85.09	92.21			LID	FGSM	-	67.28	76.58	84.88
		SUF	FGSM	Yes	93.47	96.95	99.12			SUF	FGSM	Yes	72.20	86.73	95.39
		Our	-	-	**98.14**	**99.35**	**100.00**			Our	-	-	**97.13**	**99.76**	**100.00**

Table 3: Comparison of AUROC (%) under different validation setups. The best results are bolded. We also compared the use of negative samples for training and input image pre-processing.

rate 12 (Liang, Li, and Srikant 2018; Lee et al. 2018a) and 34-layer ResNet (Lee et al. 2018a). 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, Li, and Srikant 2018; Lee et al. 2018a) need to be tuned on a validation set with 1,000 images from each in-distribution and OOD pair. We note 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.

Effects of the threshold. 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, Li, and Srikant 2018; Lee et al. 2018a). We argue that our 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 Tab. 2. For each in&out-of-distribution pair, we report the performance of ODIN (Liang, Li, and Srikant 2018), SUF (Lee et al. 2018a) and our DVN. Notably, DVN consistently outperforms the previous methods and achieves a new state-of-the-art. As shown in Tab. 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 examples are not presented.

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

Table 4: 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.

Considering the technical route of DVN is essentially different from ODIN and SUF, we compare it with the baseline, maximum softmax probability (MSP) (Hendrycks and Gimpel 2016), 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. Moreover, DVN outperforms MSP by a large margin.

Ablation studies. To demonstrate the effectiveness of each module, we provide the detailed ablation study w.r.t. the choice of VAE/PixelCNN/Glow, disentanglement of y, modifying $p(z)$ to $q(z)$ with GAN and conditioned encoder.

• PixelCNN/Glow-based DVN. We also compared with the DVN that use pixel CNN or Glow in Table 2. The pixelCNN/Glow-based DVN is consistently inferior than our solution. VAEs do have lower likelihood scores than Glow, but this gap is due to the different ways of computing likelihood of VAEs and flows. When computing the likelihood of a VAE, it is usually assumed that there is a unit Gaussian distribution at the output of the decoder. However the distribution of natural images is on a low dimensional manifold, so the likelihood number itself cannot be compared with Glow under this assumption. But VAEs are more robust than Glow due to the reason discussed in Sec 3.5, and in our experiments we found that Glows tend to put higher likelihood on OOD examples, which is bad for our usage.

• Disentangling y from z is critical to our model. Table 5 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

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

Table 5: 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 6: 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.

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.

• Without modifying $p(z)$ with $q(z)$. Since modeling $p(x|y)$ is the core of DVN, we cannot remove y. Here, we give another ablation study that without modifying $p(z)$ with $q(z)$. As shown in Table 6, 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 DVN.

• Encoder condition on y. 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. Table 7 gives a comparison of conditioning our encoder on x or (x, y).

Detecting Adversarial Examples

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. 2018a), we applied several attack methods to generate the negative samples, such as basic iterative method (BIM) (Kurakin, Goodfellow, and Bengio 2016), Deepfool (Moosavi-Dezfooli, Fawzi, and Frossard 2016), and Carlini-Wagner (CW) (Carlini and 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. 2018a) in Tab. 4, and show that the DVN can achieve the 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), 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. 2018a) are not required in our proposed DVN.

	TNR@TPR95%	AUROC
$q(z\|x)$	**98.4**	**99.2**
$q(z\|x, y)$	93.7	95.5

Table 7: The performance of DVN use $q(z|x)$ and $q(z|x, y)$ encoder. We use ResNet backbone, and choose CIFAR-10/SVHN as in-distribution/OOD.

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 8: 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.

OOD 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.

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. Tab. 8 summarizes the performance of DVN in image caption task and can be regarded as a powerful baseline.

Conclusions

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.

Acknowledgements

This work was partially supported by the Jiangsu NSF [grant number BK20200238].

References

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

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

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

Carlini, N.; and Wagner, D. 2017. Adversarial examples are not easily detected: Bypassing ten detection methods. In *ACM AISW*.

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

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

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

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

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

Goodfellow, I.; Bengio, Y.; Courville, A.; and Bengio, Y. 2016. *Deep learning*, volume 1. MIT Press.

Goodfellow, I.; Pouget-Abadie, J.; Mirza, M.; Xu, B.; Warde-Farley, D.; Ozair, S.; Courville, A.; and Bengio, Y. 2014. Generative adversarial nets. In *Advances in neural information processing systems*, 2672–2680.

Guo, C.; Pleiss, G.; Sun, Y.; and Weinberger, K. Q. 2017. On calibration of modern neural networks. In *ICML*, 1321–1330. JMLR. org.

Han, Y.; Liu, X.; Sheng, Z.; Ren, Y.; Han, X.; You, J.; Liu, R.; and Luo, Z. 2020. Wasserstein Loss-Based Deep Object Detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*.

He, G.; Liu, X.; Fan, F.; and You, J. 2020a. Classification-Aware Semi-Supervised Domain Adaptation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*.

He, G.; Liu, X.; Fan, F.; and You, J. 2020b. Image2Audio: Facilitating Semi-Supervised Audio Emotion Recognition With Facial Expression Image. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*.

Hendrycks, D.; and Gimpel, K. 2016. A Baseline for Detecting Misclassified and Out-of-Distribution Examples in Neural Networks. *CoRR* abs/1610.02136. URL http://arxiv.org/abs/1610.02136.

Hendrycks, D.; Mazeika, M.; and Dietterich, T. 2019. Deep Anomaly Detection with Outlier Exposure. *ICLR* .

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

Kingma, D. P.; and Dhariwal, P. 2018. Glow: Generative flow with invertible 1x1 convolutions. In *NIPS*, 10215–10224.

Krizhevsky, A.; Sutskever, I.; and Hinton. 2012. Imagenet classification with deep convolutional neural networks. In *NIPS*.

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

Lee; Lee; Lee; and Shin. 2018a. A Simple Unified Framework for Detecting Out-of-Distribution Samples and Adversarial Attacks. *NIPS* .

Lee, K.; Lee, H.; Lee, K.; and Shin, J. 2018b. Training Confidence-calibrated Classifiers for Detecting Out-of-Distribution Samples. *ICLR* .

Liang, S.; Li, Y.; and Srikant, R. 2018. Enhancing The Reliability of Out-of-distribution Image Detection in Neural Networks. *ICLR* .

Liu, X. 2020. Disentanglement for discriminative visual recognition. *arXiv preprint arXiv:2006.07810* .

Liu, X.; B.V.K, K.; Yang, C.; Tang, Q.; and You, J. 2018a. Dependency-aware Attention Control for Unconstrained Face Recognition with Image Sets. In *European Conference on Computer Vision*.

Liu, X.; Che, T.; Lu, Y.; and Yang, C. 2020a. AUTO3D: Novel view synthesis through unsupervisely learned variational viewpoint and global 3D representation. *ECCV* .

Liu, X.; Fan, F.; Kong, L.; Xie, W.; Lu, J.; and You, J. 2020b. Unimodal regularized neuron stick-breaking for ordinal classification. *Neurocomputing* .

Liu, X.; Guo, Z.; Jia, J.; and Kumar, B. 2019a. Dependency-aware Attention Control for ImageSet-based Face Recognition. In *IEEE Transactions on Information Forensics & Security*.

Liu, X.; Guo, Z.; Li, S.; You, J.; and B.V.K, K. 2019b. Dependency-aware Attention Control for Unconstrained Face Recognition with Image Sets. In *ICCV*.

Liu, X.; Han, X.; Qiao, Y.; Ge, Y.; Li, S.; and Lu, J. 2019c. Unimodal-uniform constrained wasserstein training for medical diagnosis. In *Proceedings of the IEEE International Conference on Computer Vision Workshops*, 0–0.

Liu, X.; Han, Y.; Bai, S.; Ge, Y.; Wang, T.; Han, X.; Li, S.; You, J.; and Lu, J. 2020c. Importance-Aware Semantic Segmentation in Self-Driving with Discrete Wasserstein Training. In *AAAI*, 11629–11636.

Liu, X.; Ji, W.; You, J.; Fakhri, G. E.; and Woo, J. 2020d. Severity-aware semantic segmentation with reinforced wasserstein training. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 12566–12575.

Liu, X.; Kong, L.; Diao, Z.; and Jia, P. 2017a. Line-scan system for continuous hand authentication. *Optical Engineering* 56(3): 033106.

Liu, X.; Kumar, B. V.; Ge, Y.; Yang, C.; You, J.; and Jia, P. 2018b. Normalized face image generation with perceptron generative adversarial networks. In *2018 IEEE 4th International Conference on Identity, Security, and Behavior Analysis (ISBA)*, 1–8. IEEE.

Liu, X.; Kumar, B. V.; Jia, P.; and You, J. 2019d. Hard negative generation for identity-disentangled facial expression recognition. *Pattern Recognition* 88: 1–12.

Liu, X.; Li, S.; Kong, L.; Xie, W.; Jia, P.; You, J.; and Kumar, B. 2019e. Feature-Level Frankenstein: Eliminating Variations for Discriminative Recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 637–646.

Liu, X.; Li, Z.; Kong, L.; Diao, Z.; Yan, J.; Zou, Y.; Yang, C.; Jia, P.; and You, J. 2018c. A joint optimization framework of low-dimensional projection and collaborative representation for discriminative classification. In *2018 24th International Conference on Pattern Recognition (ICPR)*, 1493–1498. IEEE.

Liu, X.; Lu, Y.; Liu, X.; Bai, S.; Li, S.; and You, J. 2020e. Wasserstein Loss With Alternative Reinforcement Learning for Severity-Aware Semantic Segmentation. *IEEE Transactions on Intelligent Transportation Systems* .

Liu, X.; Vijaya Kumar, B.; You, J.; and Jia, P. 2017b. Adaptive deep metric learning for identity-aware facial expression recognition. In *CVPR*.

Liu, X.; Xing, F.; Yang, C.; Kuo, C.-J.; El Fakhri, G.; and Woo, J. 2020f. Symmetric-Constrained Irregular Structure Inpainting for Brain MRI Registration with Tumor Pathology. In *MICCAI BrainLes*.

Liu, X.; Zou, Y.; Che, T.; Jia, P.; You, J.; and B.V.K, K. 2019f. Conservative Wasserstein Training for Pose Estimation. In *ICCV*. IEEE.

Liu, X.; Zou, Y.; Kong, L.; Diao, Z.; Yan, J.; Wang, J.; Li, S.; Jia, P.; and You, J. 2018d. Data Augmentation via Latent Space Interpolation for Image Classification. In *2018 24th International Conference on Pattern Recognition (ICPR)*, 728–733. IEEE.

Liu, X.; Zou, Y.; Song, Y.; Yang, C.; You, J.; and K Vijaya Kumar, B. 2018e. Ordinal Regression with Neuron Stick-breaking for Medical Diagnosis. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 0–0.

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

Moosavi-Dezfooli, S.-M.; Fawzi, A.; and Frossard, P. 2016. Deepfool: a simple and accurate method to fool deep neural networks. In *CVPR*.

Nalisnick, E.; Matsukawa, A.; Teh, Y. W.; Gorur, D.; and Lakshminarayanan, B. 2018. Do Deep Generative Models Know What They Don't Know? *arXiv preprint arXiv:1810.09136* .

Nalisnick, E.; Matsukawa, A.; Teh, Y. W.; and Lakshminarayanan, B. 2019. Detecting out-of-distribution inputs to deep generative models using a test for typicality. *arXiv preprint arXiv:1906.02994* .

Ovadia, Y.; Fertig, E.; Ren, J.; Nado, Z.; Sculley, D.; Nowozin, S.; Dillon, J.; Lakshminarayanan, B.; and Snoek, J. 2019. Can you trust your model's uncertainty? Evaluating predictive uncertainty under dataset shift. In *NIPS*, 13991–14002.

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

Reed, S.; Akata, Z.; Lee, H.; and Schiele, B. 2016a. Learning deep representations of fine-grained visual descriptions. In *CVPR*.

Reed, S.; Akata, Z.; Yan, X.; Logeswaran, L.; Schiele, B.; and Lee, H. 2016b. Generative adversarial text to image synthesis. In *International Conference on Machine Learning*, 1060–1069. PMLR.

Van den Oord, A.; Kalchbrenner, N.; Espeholt, L.; Vinyals, O.; and Graves. 2016. Conditional image generation with pixelcnn decoders. In *NIPS*.

Vyas, A.; Jammalamadaka, N.; Zhu, X.; Das, D.; and Willke, T. L. 2018. Out-of-Distribution Detection Using an Ensemble of Self Supervised Leave-out Classifiers. *ECCV* .

Wang, J.; Liu, X.; Wang, F.; Zheng, L.; Gao, F.; Zhang, H.; Zhang, X.; Xie, W.; and Wang, B. 2020. Automated interpretation of congenital heart disease from multi-view echocardiograms. *Medical Image Analysis* .

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

Yao, J.; Pan, W.; Ghosh, S.; and Doshi-Velez, F. 2019. Quality of uncertainty quantification for Bayesian neural network inference. *arXiv preprint arXiv:1906.09686* .