Self-supervised Out-of-distribution Detection with Dynamic Latent Scale GAN

Jeongik Cho

jeongik.jo.01@gmail.com

Abstract

Dynamic latent scale GAN proposed a learning-based GAN inversion method with maximum likelihood estimation. In this paper, we propose a method for self-supervised out-of-distribution detection using the encoder of dynamic latent scale GAN. When the dynamic latent scale GAN converged, since the entropy of the scaled latent random variable is optimal to represent in-distribution data, in-distribution data is densely mapped to latent codes with high likelihood. This enables the log-likelihood of the predicted latent code to be used for out-of-distribution detection. The proposed method does not require mutual information of in-distribution data and additional hyperparameters for prediction. The proposed method also showed better out-of-distribution detection performance than the previous state-of-art method.

1. Introduction

Given an in-distribution (ID) data, detecting data that does not belong to ID data is called out-of-distribution (OOD) detection (or anomaly detection). The difference between OOD detection and simple classification is that in OOD detection, only ID data is given to the model, and the model should classify ID data and unseen OOD data. In general, since the range of OOD data is very wide compared to ID data, it is almost impossible to define an OOD dataset and use it for ID-OOD classification (i.e., supervised learning).

[1, 2] proposed an OOD detection method using statistics of a pre-trained model. However, since these methods require a pre-trained model, the model cannot be customized. For example, if the pre-trained model is a large model whose input is a high-resolution image, low-resolution images should be resized and input, which is inefficient. Also, large models may not be able to be used when computational performance is limited. Or, if the input resolution of the pre-trained is low, high-resolution images should be downsampled, which degrades OOD detection performance. Furthermore, there may not be a good pre-trained model for particular data domains.

[3, 4] proposed method when the mutual information (e.g., the label of image) of data is

given. However, those methods require mutual information of each data, so they cannot be applied when the mutual information is not given.

[5, 6] proposed method when the input is an image. These methods are difficult to use for data domains other than images.

In this paper, we propose AnoDLSGAN, a self-supervised learning method for OOD detection that does not require any mutual information of data or a pre-trained model. The proposed method uses an encoder of dynamic latent scale GAN (DLSGAN) [7] for OOD detection. Simply, log-likelihood predicted by the encoder of DLSGAN is used for OOD detection. Therefore, AnoDLSGAN does not require additional hyperparameters for OOD prediction.

2. Out-of-distribution detection with DLSGAN

DLSGAN [7] proposed a learning-based GAN inversion method with maximum likelihood estimation of the encoder. The encoder of DLSGAN maps input data to predicted latent code.

We found that the log-likelihood of the predicted latent code from the encoder of DLSGAN can be used for OOD detection. There are two characteristics that allow the DLSGAN encoder to be utilized for OOD detection.

First is the latent entropy optimality. As DLSGAN training progresses, the entropy of scaled latent random variable decreases, and the entropy of the scaled encoder output increases. When DLSGAN is converged, the generator generates ID data with a scaled latent random variable, and the entropy of scaled latent random variable and scaled encoder output becomes optimal entropy for expressing ID data with the generator and encoder. It means that ID data generated by the generator is densely mapped to latent codes with high likelihood. Therefore, by the pigeonhole principle, OOD data can only be mapped to latent codes with low likelihood.

Secondly, elements of DLSGAN encoder output are independent of each other and follow a simple distribution (the same as latent distribution). Therefore, it is very easy to calculate the log-likelihood of predicted latent code.

The following equation shows the negative log-likelihood of the predicted latent code of input data.

$$ood\ score = -\sum_{i=1}^{d_z} \log f(E_i(x)|\mu_i, v_i)$$

In the above equation, x and E represent the input data point and DLSGAN encoder, respectively. $E(x)$ represents the d_z -dimensional predicted latent code of input data point x. f represents the probability density function of the i.i.d. latent random variable Z. μ and v represent the latent mean vector and latent variance vector for the probability density function f. μ is the mean vector of latent random variable Z. v is the traced latent variance vector of DLSGAN. $E_i(x)$, μ_i, and v_i represent i-th element of $E(x)$, μ, and v, respectively.

Since each element of the encoder output $E(X)$ is independent of each other, the negative log-likelihood of the predicted latent code can be simply calculated by adding the negative log-likelihood of each element.

The $ood\ score$ is the negative log-likelihood of the predicted latent code $E(x)$. If the $ood\ score$ is greater than the threshold, the input data is classified as OOD data. Otherwise, it is classified as ID data.

AnoDLSGAN is a self-supervised OOD detection method, so it does not require any mutual information of ID data or a pre-trained model. Also, only one inference of encoder E is required to classify input data.

3. Experiments

3.1 Experiments settings

We used MNIST handwritten digits dataset [8] as an ID dataset and CMNIST (Corrupted MNIST) dataset [9], FMNIST (Fashion MNIST) [10], and KMNIST (Kuzushiji MNIST) [11] dataset as OOD datasets. For the preprocessing, we added padding to the images to make the resolution 32×32 and normalized the pixel values to be between –1 and 1.

Fig. 1 shows samples of ID images and OOD images. The first column of Fig. 1 shows ID images. Columns 13-23 (right part of the right white line) show far OOD images of the OOD dataset (CMNIST, FMNIST, KMNIST). Columns 2-12 (between the two white lines) show near OOD images. Near OOD images are generated by linear interpolation between far OOD images and ID images (i.e., $near\ OOD\ image = far\ OOD\ image \times k + ID\ image \times (1-k)$). We used $k = 0.1$ to generate near OOD images. The near OOD images are hard to distinguish for humans without looking very closely.

Images from the CMNIST dataset were generated by adding corruption to the images from the MNIST dataset. Therefore, each image from the CMNIST dataset has a corresponding original image from the MNIST dataset. When generating near OOD images with the CMNIST dataset, corresponding images from the MNIST dataset were used as ID images, not random images from the MNIST dataset.

We trained four types of models for OOD detection: AnoDLSGAN, InfoGAN [12], autoencoder, and classifier. Each model is trained only with the ID train dataset. AnoDLSGAN, InfoGAN, and autoencoder were trained without labels, while the classifier was trained with labels.

Training AnoDLSGAN is the same as training DLSGAN.

Training InfoGAN is the same as training DLSGAN without a dynamic latent scale. The latent random variable of InfoGAN has a larger entropy than the optimal entropy for expressing ID data. Therefore, ID data cannot densely map to latent space. To show that AnoDLSGAN has high OOD detection performance because ID data is densely mapped to latent space, we also experimented with InfoGAN for OOD detection.

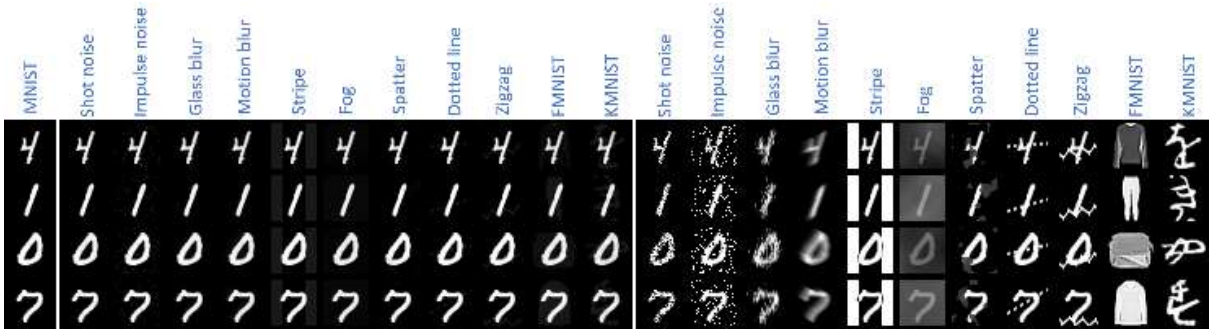

Figure 1. Sample images from datasets. Column 1: ID images, Columns 2-12: near OOD images, columns 13-23: far OOD images.

OOD detection with InfoGAN is the same as AnoDLSGAN except for the dynamic latent scale (i.e., the same OOD score function as AnoDLSGAN with traced latent variance vector is used for OOD detection).

The autoencoder is trained with reconstruction loss (l2 norm of the difference between an input image and reconstructed image). Reconstruction loss is also used for OOD detection with the autoencoder [13].

The classifier is trained with cross-entropy loss. Energy score [4] with ReAct [3] is used for OOD detection with the classifier.

Following hyperparameters were used for training models.

$$batch\ size = 32$$

$$optimizer = \begin{pmatrix} learning\ rate = 0.001 \\ beta_1 = 0 \\ beta_2 = 0.99 \end{pmatrix}$$

$$epoch = 30$$

$$learning\ rate\ decay\ rate\ per\ epoch = 0.95$$

$learning\ rate\ decay\ rate\ per\ epoch$ is a value multiplied by the learning rate for each epoch.

The following figures show the model architecture used in the experiments.

$Input$: $[32, 32, 1]$ $Image$	
$Conv\ layer\ (filter\ size = 128)$	
$Conv\ layer\ (filter\ size = 256)$	
$Downsample\ 2\times$	
$Conv\ layer\ (filter\ size = 256)$	
$Conv\ layer\ (filter\ size = 512)$	
$Downsample\ 2\times$	
$Conv\ layer\ (filter\ size = 512)$	
$Conv\ layer\ (filter\ size = 512)$	
$Downsample\ 2\times$	
$Conv\ layer\ (filter\ size = 1024)$	
$Flatten\ layer$	
$Adversarial\ (1)$	$Latent\ vector\ (256)$
	$Classifier\ (10)$

Figure 2. Encoder architecture

$Input$: $[256]$ $latent\ vector$
$Fully\ connected\ layer\ (1024\times 4\times 4)$
$Reshape\ [4, 4, 1024]$
$Upsample\ 2\times$
$Conv\ layer\ (filter\ size = 512)$
$Conv\ layer\ (filter\ size = 512)$
$Upsample\ 2\times$
$Conv\ layer\ (filter\ size = 256)$
$Conv\ layer\ (filter\ size = 256)$
$Upsample\ 2\times$
$Conv\ layer\ (filter\ size = 128)$
$Conv\ layer\ (filter\ size = 128)$
$To\ image\ layer$

Figure 3. Decoder architecture

In Figs. 2-3, the activation functions of all layers except the output layer are leaky ReLU, and the kernel size of all convolution layers is $[3\times 3]$. Equalized learning rate [16] is used for

all layers.

DLSGAN and InfoGAN use an encoder as a discriminator and a decoder as a generator. The encoder's classifier output was not used for GAN training. NSGAN with R1 regularization [14] was used for GAN training. Also, an exponential moving average with $decay\ rate = 0.999$ was used for traced variance vector v. Following hyperparameters were used for GANs training.

$$\lambda_{enc} = 1$$

$$\lambda_{r1} = 0.1$$

$$Z = (Z_i)_{i=1}^{256} \overset{i.i.d.}{\sim} N(0,1^2)$$

λ_{r1} is R1 regularization weight. The paper proposed R1 regularization used $\gamma/2$ as regularization weight, so based on that definition, γ is 0.2 when λ_{r1} is 0.1.

Autoencoder uses encoder and decoder. Adversarial output and classifier output of the encoder were not used for encoder training.

The classifier uses only an encoder. The adversarial output of the encoder was not used for classifier training.

We evaluated the model performance using 7-fold cross-validation. Each MNIST dataset has 70k images. Therefore, the ID train dataset has 60k images, the ID test dataset has 10k images, and the OOD test dataset has 10k images in each fold. The average area under the ROC curve (AUROC) was used for OOD detection performance evaluation.

3.2 Experiment results

Table 1 shows the basic model performance for each model. In table 1, FID [15] shows the generative performance of GANs. 10k generated images and 10k ID test images for each fold were used for FID evaluation. PSNR and SSIM show the difference between test images and reconstructed images.

We trained a new model for each near OOD dataset, far OOD dataset, and hyperparameters. Therefore, the basic model performance was evaluated several times. Table 1 shows the minimum/maximum values of the basic model performance for each experiment. AnoDLSGAN showed better reconstruction performance (PSNR and SSIM) than InfoGAN as [7]. However, the reconstruction performance of the autoencoder was much better than GANs. The classifier showed high accuracy.

	AnoDLSGAN (ours)	InfoGAN [12]	Autoencoder	Classifier
FID [15]	4.62~5.44	6.12~7.11	-	-
PSNR	15.56~15.93	13.79~14.18	29.08~29.10	-
SSIM ($\times 100$)	56.71~59.08	45.08~47.73	96.25~96.37	-
Accuracy ($\times 100$)	-	-	-	99.47~99.51

Table 1. Basic model performance

AUROC (× 100)		GAN		Autoencoder	Classifier					
		AnoDLSGAN (ours)	InfoGAN [12]	Reconstruction [13]	Energy[3,4] t=1, p=0.5	Energy t=1, p=0.9	Energy t=1, p=1.0	Energy t=10, p=0.5	Energy t=10, p=0.9	Energy t=10, p=1.0
NEAROOD	Shot noise	50.94	50.64	52.98	51.86	52.38	51.44	51.89	52.36	51.96
	Impulse noise	93.12	59.46	73.51	52.17	52.87	51.62	52.27	52.73	52.26
	Glass blur	62.22	52.19	48.30	54.44	55.66	53.38	54.52	55.58	54.70
	Motion blur	78.18	53.69	50.90	55.44	56.95	54.09	55.45	56.94	55.74
	Stripe	100.00	93.49	98.85	53.63	54.82	52.82	53.91	54.86	53.96
	Fog	95.26	61.20	76.12	58.21	60.54	56.31	58.66	60.62	58.77
	Spatter	63.61	51.42	52.48	51.08	51.39	50.83	51.11	51.38	51.14
	Dotted line	68.17	53.00	56.78	50.56	50.74	50.41	50.56	50.69	50.59
	Zigzag	85.78	56.90	65.78	51.16	51.53	50.86	51.17	51.45	51.21
	FMNIST	98.90	65.88	82.56	60.61	63.48	58.09	60.88	63.37	61.19
	KMNIST	97.94	65.88	84.74	60.76	63.62	58.17	60.94	63.40	61.27
FAROOD	Shot noise	97.21	68.52	99.74	72.65	76.81	70.25	70.08	77.84	73.48
	Impulse noise	100.00	100.00	100.00	82.31	85.79	79.03	76.55	86.79	83.09
	Glass blur	99.97	93.15	99.82	91.02	94.73	85.81	88.98	95.99	92.04
	Motion blur	100.00	95.07	97.96	89.67	93.83	83.94	88.82	95.16	90.91
	Stripe	100.00	100.00	100.00	90.06	93.82	85.49	87.55	94.13	90.82
	Fog	100.00	99.94	99.99	99.90	99.97	92.79	99.87	99.99	99.95
	Spatter	99.66	84.20	99.44	66.20	69.60	64.52	64.87	70.54	66.75
	Dotted line	99.37	84.71	99.97	61.99	64.43	60.91	58.23	63.97	61.67
	Zigzag	99.82	92.69	99.99	67.10	70.26	65.60	60.49	69.34	66.40
	FMNIST	100.00	99.85	99.98	95.53	97.89	89.68	91.98	97.97	96.00
	KMNIST	99.95	97.89	99.99	94.03	96.94	88.63	86.78	96.61	93.51

Table 2. OOD detection performance for each method. Each value in the table is the average AUROC multiplied by 100.

Table 2 shows the OOD detection performance for each method. 1000 intervals between the minimum and maximum values of the $ood\ score$ of the ID test dataset were used for the threshold value. Each value in the table is the average AUROC multiplied by 100. In "Energy" of Table 2, "t" represents the temperature of the energy score [4], and "p" represents the activation percentage of ReAct [3]. When p = 1.0, it is the same as that ReAct was not applied.

In Table 2, one can see that the overall performance of AnoDLSGAN is the best. The performance of AnoDLSGAN is not significantly different from that of autoencoder reconstruction in far OOD detection, but it shows significantly better performance in near OOD detection, even if the reconstruction performance of autoencoder is much better than AnoDLSGAN's.

Additionally, OOD detection with autoencoder reconstruction shows better performance than the energy score with ReAct. Comparing p=0.9 and p=1.0 in the energy score, one can see that there is a performance improvement when ReAct is applied as [3].

Energy score with ReAct showed good performance in easy OOD of far OOD datasets detection (FMNIST and KMNIST) but showed relatively poor performance in some CMNIST datasets (Spatter, Dotted line, Zigzag). Also, it could hardly distinguish the near OOD images.

Also, one can see that AnoDLSGAN has clearly better near OOD detection performance than InfoGAN. This indicates that AnoDLSGAN

performs better than InfoGAN because ID data is densely mapped to the latent space due to the latent entropy optimality of DLSGAN.

All methods failed to distinguish Shot noise near OOD images. Shot noise near OOD images are difficult to distinguish even for humans.

4. Conclusion

In this paper, we showed that the encoder of DLSGAN can be used for OOD detection. The latent entropy optimality of DLSGAN enables ID data to be densely mapped to latent codes with high likelihood. This characteristic of DLSGAN enables high OOD detection performance with the DLSGAN encoder. OOD detection with AnoDLSGAN is very simple and does not require a pre-trained model, mutual information of ID data, and additional hyperparameters for prediction. Also, AnoDLSGAN showed high OOD detection performance compared to the previous state-of-art method.

5. References

[1] Rui Huang, Andrew Geng, and Yixuan Li, "On the Importance of Gradients for Detecting Distributional Shifts in the Wild", in NIPS 2021, https://proceedings.neurips.cc/paper/2021/file/063e26c670d07bb7c4d30e6fc69fe056-Paper.pdf

[2] Rui Huang, and Yixuan Li, "MOS: Towards Scaling Out-of-distribution Detection for Large Semantic Space", in CVPR 2021,

https://openaccess.thecvf.com/content/CVPR2021/papers/Huang_MOS_Towards_Scaling_Out-of-Distribution_Detection_for_Large_Semantic_Space_CVPR_2021_paper.pdf

[3] Yiyou Sun, Chuan Guo, Yixuan Li, "ReAct: Out-of-distribution Detection With Rectified Activations", in NIPS 2021,

https://openreview.net/forum?id=IBVBtz_sRSm

[4] Weitang Liu, Xiaoyun Wang, John D. Owens, Yixuan Li, "Energy-based Out-of-distribution Detection", in NIPS 2020, https://proceedings.neurips.cc/paper/2020/file/f5496252609c43eb8a3d147ab9b9c006-Paper.pdf

[5] Jihun Yi and Sungroh Yoon, "Patch SVDD: Patch-level SVDD for Anomaly Detection and Segmentation", in ACCV 2020,

https://openaccess.thecvf.com/content/ACCV2020/papers/Yi_Patch_SVDD_Patch-level_SVDD_for_Anomaly_Detection_and_Segmentation_ACCV_2020_paper.pdf

[6] Xudong Yan, Huaidong Zhang, Xuemiao Xu, Xiaowei Hu, Pheng-Ann Heng, "Learning Semantic Context from Normal Samples for Unsupervised Anomaly Detection", in AAAI

2021,
https://ojs.aaai.org/index.php/AAAI/article/view/16420/16227

[7] Jeongik Cho, Adam Krzyzak, "Dynamic Latent Scale for GAN Inversion," In Proceedings of the 11th ICPRAM

[8] Yann LeCun, Corinna Cortes, Christopher J.C. Burges, "THE MNIST DATABASE of handwritten digits",

http://yann.lecun.com/exdb/mnist/

[9] Norman Mu, Justin Gilmer, "MNIST-C: A Robustness Benchmark for Computer Vision", in arXiv preprint,

https://arxiv.org/abs/1906.02337

[10] Han Xiao, Kashif Rasul, Roland Vollgraf, "Fashion-MNIST: a Novel Image Dataset for Benchmarking Machine Learning Algorithms", in arXiv preprint,

http://arxiv.org/abs/1708.07747

[11] Tarin Clanuwat, Mikel Bober-Irizar, Asanobu Kitamoto, Alex Lamb, Kazuaki Yamamoto, David Ha, "Deep Learning for Classical Japanese Literature", in arXiv preprint,

https://arxiv.org/abs/1812.01718

[12] Xi Chen, Yan Duan, Rein Houthooft, John Schulman, Ilya Sutskever, Pieter Abbeel, "InfoGAN: Interpretable Representation Learning by Information Maximizing Generative Adversarial Nets", in NIPS 2016,

https://papers.nips.cc/paper/2016/hash/7c9d0b1f96aebd7b5eca8c3edaa19ebb-Abstract.html

[13] Tensorflow autoencoder tutorial,

https://www.tensorflow.org/tutorials/generative/autoencoder

[14] Lars Mescheder, Andreas Geiger, Sebastian Nowozin, "Which Training Methods for GANs do actually Converge?", in arXiv preprint,

https://arxiv.org/abs/1801.04406v4

[15] Martin Heusel, Hubert Ramsauer, Thomas Unterthiner, Bernhard Nessler, Sepp Hochreiter, "GANs Trained by a Two Time-Scale Update Rule Converge to a Local Nash Equilibrium", in NIPS 2017,

https://papers.nips.cc/paper/2017/hash/8a1d694707eb0fefe65871369074926d-Abstract.html

[16] Tero Karras, Timo Aila, Samuli Laine, Jaakko Lehtinen, "Progressive Growing of GANs for Improved Quality, Stability, and Variation", in ICLR 2018,

https://openreview.net/forum?id=Hk99zCeAb