Unsupervised Out-of-distribution Detection with DLSGAN

Jeongik Cho

jeongik.jo.01@gmail.com

Abstract

DLSGAN proposed a learning-based GAN inversion method with maximum likelihood estimation. In this paper, I propose a method for unsupervised out-of-distribution detection using the encoder of DLSGAN. When the DLSGAN converged, since the entropy of the scaled latent random variable is optimal to express 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.

1. Out-of-distribution detection with DLSGAN

DLSGAN [4] 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.

In this paper, I propose a method for unsupervised out-of-distribution (OOD) detection using the encoder of DLSGAN. Simply, the log-likelihood of the predicted latent code of input data can be used for out-of-distribution detection. There are two characteristics that allow the DLSGAN encoder to be utilized for OOD detection. First is the 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 perfectly generates in-distribution data with a scaled latent random variable, and the entropy of scaled latent random variable and scaled encoder output becomes optimal entropy for expressing in-distribution data with generator and encoder. It means that in-distribution 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 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 mean and variance vector for the probability density function f. μ is mean vector of latent random variable Z. v is the same vector as traced 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 in-distribution data.

The proposed method is an unsupervised OOD detection method, so it does not require any mutual information of training data (e.g., the label of the training data), and only one inference of encoder E is required to classify input data.

2. Experiments

2.1 Experiments settings

I used MNIST handwritten digits dataset [1] as an in-distribution dataset and corrupted MNIST dataset [2], fashion MNIST [7], and KMNIST [8] dataset as an OOD dataset. The following figure shows samples of in-distribution data and OOD data.

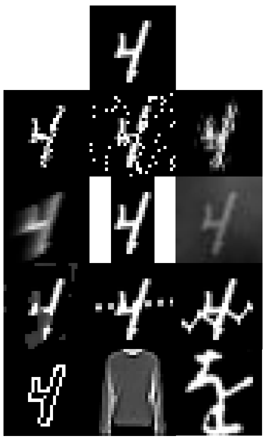

Figure 1. Samples of the dataset. The first top image shows in-distribution data. Other images show OOD data.

I trained DLSGAN to generate in-distribution data with an MNIST handwritten digits training dataset, then measured the OOD detection performance of the proposed method. 10k test dataset of the MNIST dataset was used as the in-distribution dataset for evaluation, and the 10k test dataset of each OOD dataset was used as the OOD dataset for evaluation. AUROC (area under ROC curve) was used for OOD detection performance evaluation.

Following hyperparameters was used for

DLSGAN training.

$$\lambda_{enc} = 1$$

$$\lambda_{r1} = 1$$

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

$$batch\ size = 32$$

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

$$epoch = 30$$

Also, an exponential moving average with $decay\ rate = 0.999$ was used to approximate the element-wise variance of the predicted latent vector for DLSGAN. NSGAN with R1 regularization [3] was used for DLSGAN training.

The following table shows the performance of trained DLSGAN.

FID [5]	5.4527
Precision [6]	0.7434
Recall [6]	0.6728
Fake PSNR	20.1497
Fake SSIM	0.7599
Real PSNR	16.4589
Real SSIM	0.6129

Figure 2. Performance of trained DLSGAN

10k generated images and 10k test images were used for DLSGAN performance evaluation.

2.2 Experiments results

For the threshold value, 1000 intervals between the minimum and maximum values of the $ood\ score$ of the in-distribution test dataset were used.

	AUROC
Shot noise	0.9732
Impulse noise	1.0000
Glass blur	0.9969
Motion blur	0.9999
Stripe	1.0000
Fog	1.0000
Spatter	0.9882
Dotted line	0.9953
Zigzag	0.9985
Canny edges	1.0000
Fashion MNIST	1.0000
KMNIST	0.9984

Figure 3. OOD detection performance

Figure 3 shows the OOD detection performance of the proposed method. Each row of the table shows the AUROC performance according to each OOD dataset. One can see that the proposed method almost perfectly detected OOD data.

3. Conclusion

In this paper, I found that the DLSGAN encoder can be used to estimate the likelihood of input data. The proposed unsupervised OOD detection method is very simple and showed high performance even for the OOD data very close to the in-distribution data.

4. References

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

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

[2] Norman Mu, Justin Gilmer, "MNIST-C: A Robustness Benchmark for Computer Vision"

https://arxiv.org/abs/1906.02337

[3] Lars Mescheder, Andreas Geiger, Sebastian Nowozin, "Which Training Methods for GANs do actually Converge?"

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

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

[5] 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"

https://arxiv.org/abs/1706.08500

[6] Tuomas Kynkäänniemi, Tero Karras, Samuli Laine, Jaakko Lehtinen, Timo Aila, "Improved Precision and Recall Metric for Assessing Generative Models"

https://proceedings.neurips.cc/paper/2019/hash/0234c510bc6d908b28c70ff313743079-Abstract.html

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

http://arxiv.org/abs/1708.07747

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

https://arxiv.org/abs/1812.01718