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 out-of-distribution 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 (OOD) detection.

1. OOD detection 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.

When the DLSGAN converged, one can know the true distribution of DLSGAN encoder output. Therefore, the log-likelihood of input data can be simply calculated through the DLSGAN encoder. The following equation shows the log-likelihood of the predicted latent code of input data.

$$ood\ score = sum(\log f(E(x)|\mu, v))$$

In the above equation, x and E represent input data and DLSGAN encoder, respectively. $E(x)$ represents d_z -dimensional predicted latent code of input data x. f represents probability density function of the i.i.d. latent random variable Z. μ and v represents 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.

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

2. Experiments

2.1 Experiments settings

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

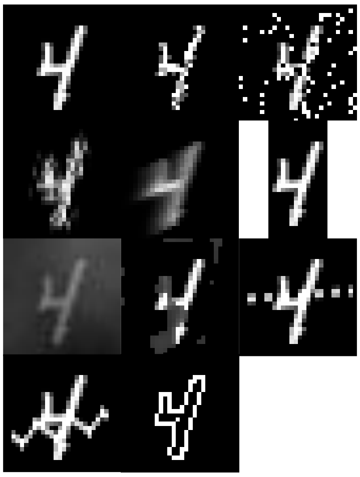

Figure 1. Samples of the dataset. The first image shows in-distribution data. Other images show OOD data.

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, and 10k test dataset of corrupted MNIST was used as OOD dataset. AUROC was used for OOD detection performance evaluation.

For the threshold value, 100 intervals from 0 to 1000 were used.

Following hyperparameters was used for DLSGAN training.

$$\lambda_{enc} = 1$$

$$\lambda_{r1} = 10$$

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

$$batch\ size = 32$$

Also, an exponential moving average with $decay\ rate = 0.999$ was used to approximate the element-wise variance of the predicted latent vector. NSGAN with R1 regularization [3] was used for DLSGAN training. DLSGAN was trained with $learning\ rate = 10^{-3}$ for the first 30 epochs and then trained with $learning\ rate = 10^{-5}$ for the next 30 epochs.

The following table shows the performance of trained DLSGAN.

FID [5]	11.2907
Precision [6]	0.6465
Recall [6]	0.5571
Fake PSNR	25.8172
Fake SSIM	0.8901
Real PSNR	17.8839
Real SSIM	0.6848

Figure 2. Performance of trained DLSGAN

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

2.2 Experiments results

	AUROC
Shot noise	0.8961
Impulse noise	1.0000
Glass blur	0.9914
Motion blur	0.9995
Stripe	1.0000
Fog	1.0000
Spatter	0.9785
Dotted line	0.9958
Zigzag	0.9989
Canny edges	0.9999

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 the OOD dataset. One can see that the proposed method almost perfectly detected OOD data.

3. Conclusion

In this paper, I found that the encoder of DLSGAN can be used to estimate the likelihood of input data. The proposed method shows high detection 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] J. Cho, and A. 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