OA-GAN: Overfitting Avoidance Method of GAN oversampling based on xAI

Jiha Kim and Hyunhee Park*
dept. Information and Communication Engineering
Myongji University
Yongin, Korea
yaki5896@mju.ac.kr, hhpark@mju.ac.kr*

***Abstract*—The most representative method of deep learning is data-driven learning. These methods are often data-dependent, and lack of data leads to poor learning. There is a GAN method that creates a likely image as a way to solve a problem that lacks data. The GAN determines that the discriminator is fake/real with respect to the image created so that the generator learns. However, overfitting problems when the discriminator becomes overly dependent on the learning data. In this paper, we explain overfitting problem when the discriminator decides to fake/real using xAI. Depending on the area of the described image, it is possible to limit the learning of the discriminator to avoid overfitting. By doing so, the generator can produce similar but more diverse images.**

***Index Terms*—GAN, xAI, Generator, Discriminator, Overfitting**

I. Introduction

Deep learning learns data to generate models for classifying and predicting results. In the learning process, data is proceeded with training data, validation data, and test data. If there is no validation data, the model may become overly dependent on learning data, resulting in overfitting problems. In addition, if the data is too imbalanced, classification for a particular class may not be achieved or predicted. To address these overfitting and data imbalances, users must choose between oversampling and undersampling when dealing with data. If you have enough data, you can do undersampling, but usually use oversampling to increase the data. A typical oversampling method is data augmentation.

Data augmentation is a method for creating new images using existing images, and there are image-based methods and deep learning-based methods. Among these, image-based methods are commonly used, and are a method that transforms existing images and creates new ones. However, if you really need a new data sample, it does not work. Deep learning-based methods use existing images, but create new images from them. Typical methods include Generator Adversarial Nets (GAN) [1].

A GAN consists of a generator that produces images and a discriminator that identifies the images that are generated. Existing GANs cannot create images of RGB channels. This is why Deep Convolutional GAN (DCGAN) [2] is used.

Although DCGAN has the same mechanism as GAN, it has the advantage of creating both RGB channels. But, the question arises whether images made of DCGAN can be trusted. In fact, image learning is done in a fairly large epoch (e.g., usually the number of times the entire epoch multiplied by a dataset/batch), but it may not be satisfactory for whether the image is truly meaningful. In Reduced Net (RedNet) [3] paper, creating images using DCGAN was successful, but it is judged not to be meaningful. It is necessary to explain this model to determine whether the generated image is meaningful and reliable. methods for describing models include Local Interpretable Model-agnostic Explanations (LIME) [4] and SHapley Additive exPlanations (SHAP) [5].

The LIME method is a method that shows the basis for which areas the model used to judge when predicting data. One of the great advantages of LIME is that it can be applied regardless of how it is learned(i.e., Model-agnostic). SHAP methods utilize the independence between shapely value and feature. Shapely value utilizes the contribution of each feature. It is to utilize how much the prediction results change except for a particular feature.

xAI-GAN [6] using these xAI methods shows that the quality of images generated using MNIST and FMNIST datasets has increased by up to 23.18%. However, the created images are created with images similar to the learning data, indicating that the generator creates images similar to the learning data.

In this paper, we describe the generated and real images to prevent the discriminator from being overly dependent on data. To limit the learning of the discriminator by viewing the described images, we propose a method to prevent overfitting using a value called threshold.

II. OA-GAN model parameter

This section describes a model for discriminating and generating images using DCGAN. The data used to learn the model are the 'CIFAR-10'[1] dataset and the 'Dogs vs. Cats'[2] dataset from kaggle. The parameters for the model are described in the subsection below. The dataset used as an input is used as an RGB value of 3 channels. Scale from $0 \sim 255$ to $-1 \sim 1$. For Width and Height, the image of the CIFAR-10 dataset is 32 by 32, but the Dogs vs. Cats dataset is resized to 64 by 64 and 128 by 128. This changes the parameters of the input of the discriminator and the input/output of the constructor. For

[1]https://www.cs.toronto.edu/ kriz/cifar.html
[2]https://www.kaggle.com/c/dogs-vs-cats

common active functions, LeakyReLU is used on all layers except for output. LeakyReLU uses features that are unaffected together by a magnitude of 0.2, to influence the learning of the model.

A. Discriminator

Fig. 1 shows the layer of the discriminator used in this paper. The discriminator has four layers of convolutional layers. Use 3x3 filter and same padding for all convolutional layers. The number of filters is set to 64, 128, 128, and 256, respectively. Use stride 1 for the first convolutional layer and 2 for the rest of the layers. In the Fully connected layer, dropout is set to 0.4 to prevent overfitting. In the final output layer, use sigmoid to make a value between 0 and 1 to determine the fake/real.

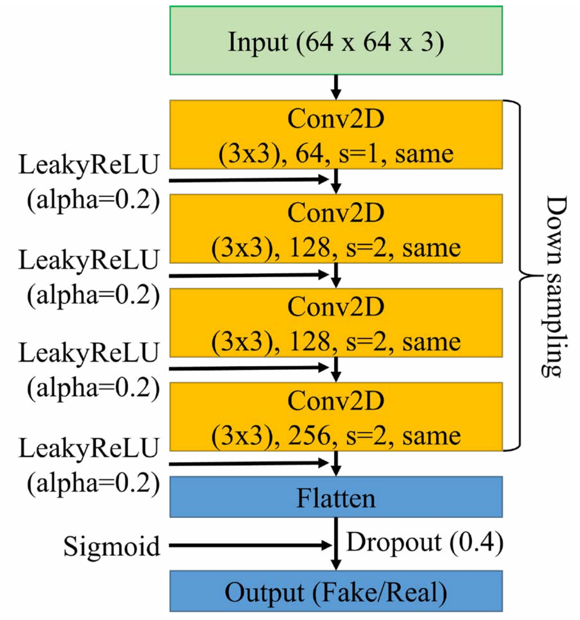

Fig. 1: Hyperparameters of Discriminator.

B. Generator

Fig. 2 shows the layer of the 64 by 64 image generator. The number of input nodes is typically determined by the output shape of the convolutional layer of the discriminator. The output shape of the convolutional layer used in this paper is 256 x 8 x 8. Therefore, the nodes to which noise is input total 16,384[3].

The Upsampling process is the opposite of the convolutional layer. First, noise is entered by node in the dense layer. It is output via an active function (e.g., LeakyReLU), which makes it an 8 x 8 x 256 feature map(i.e., This is the process of creating a feature map over the last filter in the convolutional layer). Using the feature map generated, upsampling to 16 x 16 x 128. Repeat this process to finally make a feature map of 64 x 64 x 128 into an image sample of 64 x 64 x 3.

[3]In the 128 by 128 model used together in this paper, the output shape is 256 x 16 x 16. Thus, the number of input nodes for 128 by 128 generator models is 65,536.

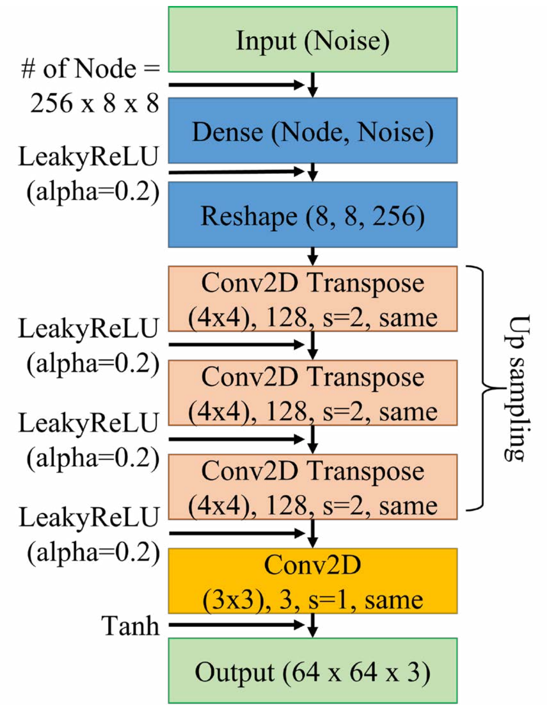

Fig. 2: Hyperparameters of Generator.

III. Learning process

The learning process of the eXplainable GAN model follows the learning process of the existing GAN. Fake/Real is determined by using 50% of the fake images and real images created by the uneducated generator. However, overfitting of the discriminator can occur here. Fig. 3 illustrates the results of performing DCGAN using the CIFAR-10 dataset and the images created. The described image can be seen active in all areas of the entered image.

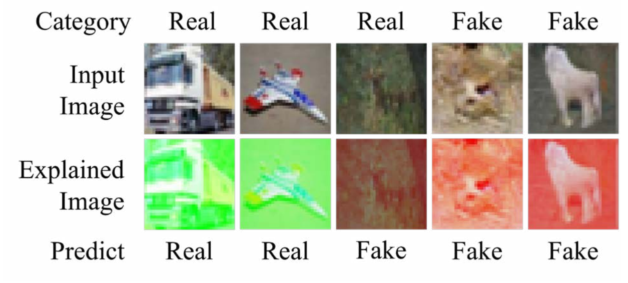

Fig. 3: Results describing the discriminator of DCGAN learned by the CAFIR-10 dataset.

Given that the entire area of the image was determined from the described results, the model may not have been determined using a particular area of the image. The fact that the discriminator judged the entire image means that it is too biased towards the learning data. Therefore, any image generated by the constructor activates all areas and produces results. In addition, the smaller the size of the image, the more active the entire image is in the region described.

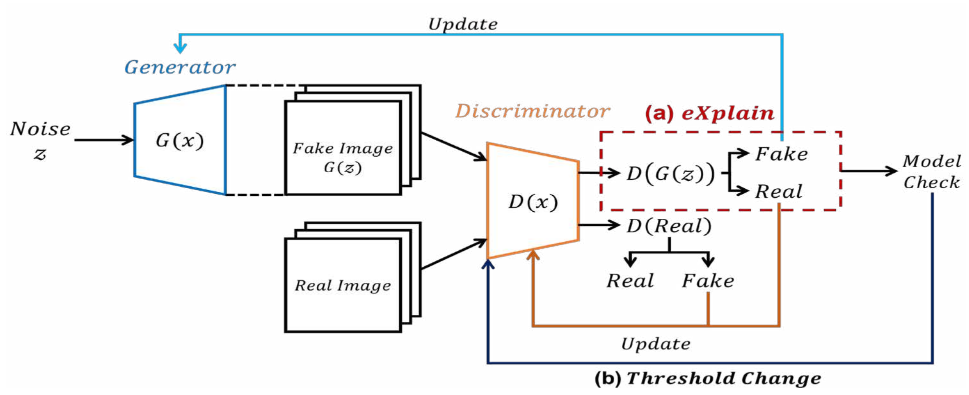

Fig. 4: Learning process of eXplainable OA-GAN.

Algorithm 1 pseudo-code of eXplainable OA-GAN

```
Input: Discriminator $D$
Input: Generator $G$
Input: Iteration
Input: Batch $m$
Input: (# of dataset)/Batch $Step$
Output: Trained models $D$ and $G$,
    Explained image with LIME
1:  Initialization Model $D(x)$ and $G(x)$
2:  for $i$ from $Iteration$ do
3:    for $s$ from $Step$ do
4:      Predict data
  •       Real images $X \in \{x_1, x_2, ...x_m\}$
  •       Fake images $Z \in \{z_1, z_2, ...z_m\}$
5:      if $(s < \frac{Step}{2})$ then
6:        Update model $D$ up to half of Step.
          $\nabla_{\theta_d} \frac{1}{m} \sum_{j=1}^{m} log(D(x_j)) + log(1 - D(G(z_i)))$
7:      else
8:        Set the average of the two losses ($D(X)$, $D(Z)$)
          of Model D to Threshold
          $Threshold = \frac{log(D(X))+log(D(Z))}{2}$
9:        if $Threshold > 0.5$ then
10:         $\nabla_{\theta_d} \frac{1}{m} \sum_{j=1}^{m} log(D(x_j)) + log(1 - D(G(z_j)))$
11:       else
12:         If the $Threshold$ value is less than 0.5, pass the
            update of $D$ to prevent overfitting.
13:       end if
14:     end if
        $\nabla_{\theta_g} \frac{1}{m} \sum_{j=1}^{m} log(1 - D(G(z_j)))$
15:   end for
16: end for
```

In this paper, we propose a method for controlling the process in which the discriminator proceeds with learning using a value called threshold. The threshold used here uses fake images and real images as the average of loss predicted by the discriminator, respectively. The optimization method used in GAN is to converge to 1/2 by Equation (1).

$$D^*(x) = \frac{p_{data}(x)}{p_{data}(x) + p_g(x)} \simeq \frac{1}{2} \tag{1}$$

It can be set up using the results described to not reflect results that deviate significantly from this approximation.

Fig. 4 and Algorithm 1 are illustrations and pseudo-codes to illustrate the methods proposed in this paper. Inputs in Algorithm 1 use initialized values. First, in the case of the model, the untrained discriminator D and generator G are input. Iteration means the total number of epochs to learn. Batch m represents the number of datasets to be trained at once in each epoch. A $step$ is a value obtained by dividing the total number of datasets by batch m, and training is performed as much as the corresponding $step$ in one epoch. As outputs, there are the trained model discriminator D and generator G, and the explained image using LIME is output.

The key parts of the proposed method are (a), (b) in Fig. 4 and line 8, 9 in Algorithm 1. $Step/2$ of each itemization follows the general GAN mechanism. Subsequently, the generated model is described using LIME as shown in Fig. 4(a). If the described image is determined to be overfitting, the threshold value is set through the procedure Fig. 4(b). This value is calculated as shown in line 8 of algorithm 1 and is used to prevent overfitting of the discriminator.

In the next learning, it proceeds through the process of line 8 and 9 of Algorithm 1. Line 9 in algorithm 1 allows learning only when the threshold value is greater than 0.5. The value of 0.5 is a value reflected based on Equation(1). When it becomes less than 1/2 of the optimized value, it acts as a threshold not to be reflected in learning.

IV. Simulation Result

The learning process in this paper is conducted in the same environment as Table 1.

Fig. 5 is the result of learning by specifying 1000 for each image size. Fig. 5-(a) was the result of learning using 64 by

TABLE I: Learning environment

Category	***Server***	***Desktop***
CPU	-	AMD Ryzen7 3700X
RAM for CPU	-	32 GBytes
GPU	TESLA T4	GeForce 1660 SUPER
RAM for GPU	16 GBytes	6 GBytes

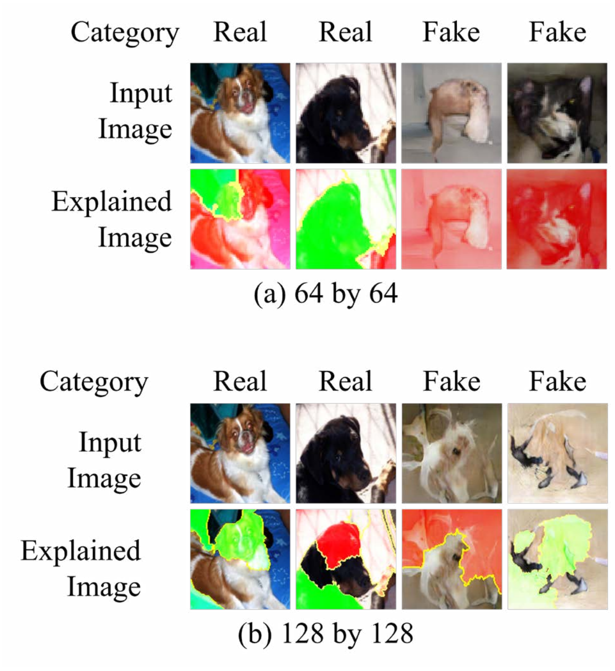

Fig. 5: Results from typical DCGAN learning processes.

64 images, which took approximately 24 hours to learn from the server. The smaller the image, the more likely it is to overfit the learning process. The 64 by 64 image shows that all regions are active in the described image, while the 128 by 128 image shows only regions that have a specific impact are active.

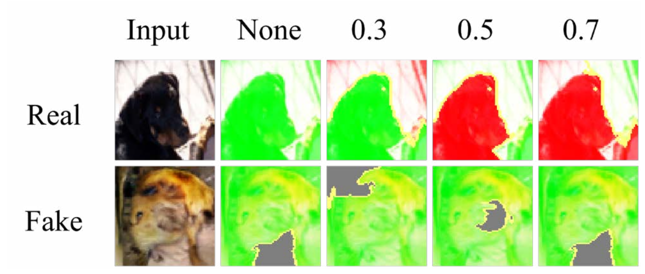

Fig. 6: Learning Results for 64 by 64 Images.

Fig. 6 shows the illustrated results of 64 by 64 images. Models learned from 64 by 64 images can be seen that the discriminator is overfitted with the learning data. Adjusting the threshold also allows the discriminator to judge the results by looking at all areas of the image. Even if we try to oversampling such a small image using GAN, it is hard to expect a new image. It seems that a slightly distorted and distorted image can be created.

Fig. 7 shows the results of describing 128 by 128 images. Compared with 64 by 64 images, we can see that the active region is somewhat specific. When the GAN model is described in this way, it can be seen that the area in which the

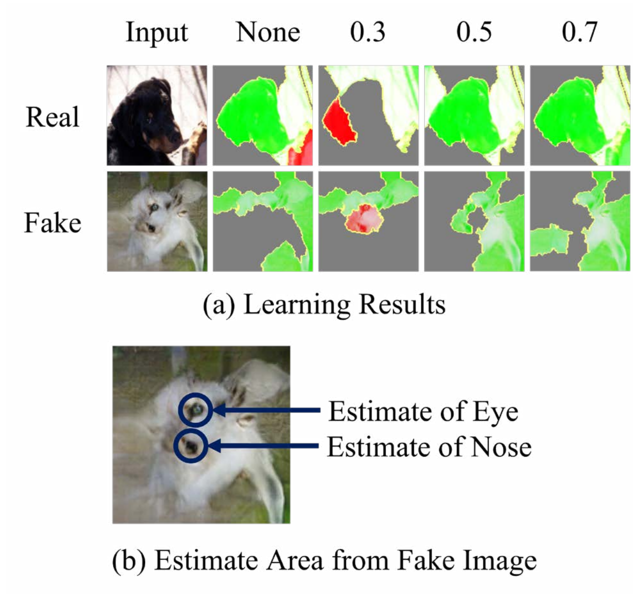

Fig. 7: Learning Results for 128 by 128 Images.

discriminator is based on a large size image rather than a small size image becomes clear.

In the Real image, when the threshold is None, you can see that the results include negative areas (i.e., red area). However, at 0.5, the same area is shown, but the negative area is not included. Fig. 7(b) shows the areas that can be estimated with eyes and nose in the fake image, and the results described in the fake image are as follows:

- None: Overfitting the learning data does not reflect the estimated areas of the eyes and nose in the results.
- 0.3: It finds areas that can be estimated with eyes and nose, but is used negatively for results.
- 0.5: It can be seen that the estimated areas of the eye and nose have a meaningful effect on the results.
- 0.7: Learning has become too rough to find the estimated area.

The above results suggest that 0.5 is appropriate for the value of the threshold, which can limit the learning of discriminator.

Fig. 3, 6, and 7 show that the larger the size, the clearer the basis for the features to judge. The discriminator tries to find a specific area as a basis for judgement, but the smaller the image, the more overlapping the area. If that happens, all areas of the image will have to be used as a basis. However, the larger the image, the more specific the area can be found.

V. Conclusion

In this paper, we propose a method to describe the GAN model with a total of three kinds of simulations. As mentioned in the previous section, smaller images can result in overfitting of discriminators that allow the model to view and judge all areas.

Therefore, the simulation results are described as 64 by 64 images and 128 by 128 images. When the model is described using LIME, it can be viewed as overfitting if it is explained by all areas of the output result image. Overfitting of discriminators is a problem in GAN models. When overfitted to the discriminator, all images generated by the generators are close to the training image. This problem can be a problem for GAN of 'generated similar image'.

The future work of this study is to propose a quantified scale to the user by quantifying the data described as LIME. In general, learning must be terminated in order to determine the extent to which the model has been learned when progressing learning. Subsequently, only the resulting images should be checked using xAI methods such as LIME et al.

However, there is a problem that users can accept subjectively. It is the future work of this study that allows this problem to be reflected in learning with quantified values.

Acknowledgment

This work was supported by Institute for Information & communications Technology Planning & Evaluation(IITP) grant funded by the Korea government(MSIT) (No. 2021-0-00990, Research on Advanced Core Technologies for WLAN based on eXplainable AI) and the National Research Foundation of Korea (NRF) grant funded by the Korea government (MSIT) (No. 2019R1F1A1060742, Big Data Analysis and Development of Security Protocol for Massive IoT Security from Cloud to Edge Computing).

References

[1] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, and Y. Bengio, "Generative adversarial nets," In Advances in Neural Information Processing Systems, pp. 2672–2680, 2014.

[2] A. Radford, L. Metz, and S. Chintala, "Unsupervised representation learning with deep convolutional generative adversarial networks," arXiv preprint arXiv:1511.06434, 2015.

[3] J. Kim, and H. Park. "Reduced CNN Model for Face Image Detection with GAN Oversampling," International Conference on Innovative Mobile and Internet Services in Ubiquitous Computing. Springer, Cham, 2021.

[4] M. T. Ribeiro, S. Singh, and C. Guestrin, ""Why should i trust you?" Explaining the predictions of any classifier," Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining. 2016.

[5] S. M. Lundberg, G. G. Erion, and S.-I. Lee, "Consistent individualized feature attribution for tree ensembles," arXiv preprint arXiv:1802.03888, 2018.

[6] V. Nagisetty, L. Graves, J. Scott, and V. Ganesh, "xAI-GAN: Enhancing Generative Adversarial Networks via Explainable AI Systems," arXiv preprint arXiv:2002.10438, 2020.