Detection of GAN Generated Images with CNNs

1st Burak Bekci
Computer Engineering
Istanbul Technical University
Istanbul, Turkey
bekciburak97@gmail.com

***Abstract*—Nowadays generating fake images with neural networks became a hot topic. Generating fake face images with generative adversarial network(GAN) models is one the problems that many researchers have been focused on. Withe researches made on them, GANs are getting better and better everyday. However, this rapid development of them may lead to serious security and privacy problems. In this paper, a security system that can detect generated fake face images is created. Using BGAN and DCGAN implementations with the CelebA dataset, fake face images are created. These images tried to discriminate from genuine images using feature extractor convolutional neural networks (CNN) and classifier models such as SVMs and Neural Networks.**

***Index Terms*—GAN, bioinformatics, CNN, deep learning, SVM**

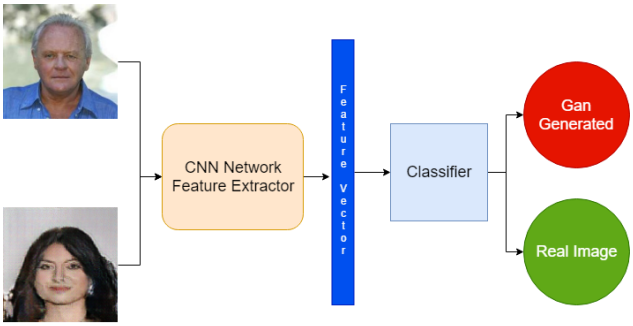

Fig. 1. The architecture of the project

I. Introduction

Generative adversarial networks are one of the topics that many people contribute to. After their first proposition in 2014 [3], they are widely used for creating fake images. Until then people have made very successful models that can generate very realistic images from many different contexts. GANs can be useful for generating imagesets and preventing dataset biases. However, generating images can be a problematic issue when the domain is humans. Generating realistic faces with computers with huge amount of speed can lead to serious privacy and security problems. With the huge amount of personal information and fake face images, it is more easy to create fake identities in nowadays. In order to prevent security attacks and privacy leaks, it is important to create a quick and robust systems that can detect created images. In this work GAN generated images were tried to discriminated from real images using CNNs as discriminators and classifier models. The system tried to be built in the project is given in the Figure: 1.

First of all the datasets are collected. No public dataset for GAN generated images were used yet they are created by using publicly available CelebA dataset [7] which contains more than a hundred thousand images of celebrities from all around the world. After real images were obtained, fake face images were created using BGAN [4] and DCGAN [11] implementations. In order to analyze the effect of image size to the classification results, images were created in different sizes. A combination of feature extractor and classifier model were used to be able to classify real and generated images. As a feature extractor, a VGG16 model pretrained with VggFace [8] data is used. For the classifier different models such as SVM and Neural Networks are tried.

II. Literature Review

The adventure of generative adversarial models began in 2014 [3]. Since then GANs become very popular and addressed by many researches [3], [5], [11], [12]. These researches improved initial GAN architecture and training process. Firstly, a Fully Convolutional architecture was published [11]. Then WGANs [1] come out. This GAN implementation made training steps more stable by enabling hyperparameter debugging models and achieved more meaningful learning curves. Lastly, high resolution images were created [5]. These images have a quality that can fool many humans. Generative models proved that generated images can be helpful to reduce dataset bias and generalizing models. They have been very beneficial since 2014 yet their possible drawbacks must be forethought. Generating face images with GANs can lead to privacy and security problems with huge scale and power. Therefore people are working to discriminate real images from generated ones [9], [6], [2]. These experiments tried to find a descriptive features like color cues [9] and image disparities in color spaces [6]. After features obtained, they discriminate the images using a classifiers like SVMs.

Detection of GAN generated images is an important ongoing research and continue to evolve with the developments in the generative models.

III. Methodology

The complete methodology of the project can be analyzed in following parts: collecting and creating imagesets, implemen-

tation of feature extractor model and classification comparison and results.

A. Collecting and Creating Imagesets

In the project obtained real images were used for both generating fake images and classification. Generated fake images and real images used for classification. The goal was creating fake face images and for that purpose generative adversarial models were used. In order to train GAN models, CelebA imageset used. That imageset contain more than a hindered thousand images of celebrities from all around the world. The faces of the celebrities in the images of CelebA dataset have been cropped and aligned. Therefore, no additional operation was performed before feeding GAN models.

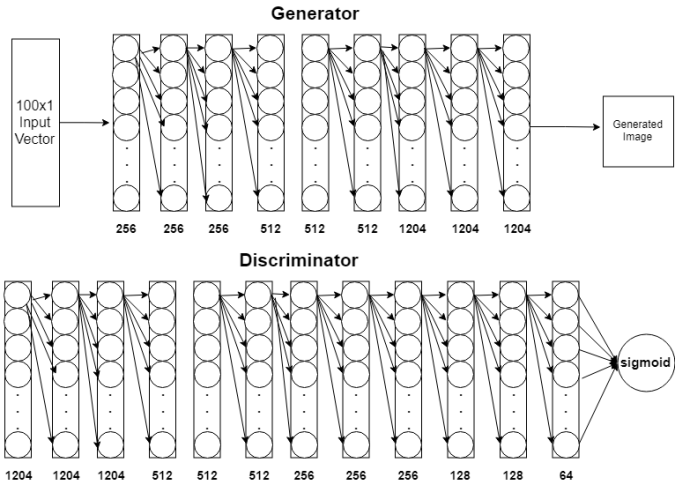

Fig. 2. The architecture of BGAN

Using the real images and GAN models, fake face images were created. While creating fake images, two different GAN models were used and images in different sizes were created. Fake images generated in the following shapes: 32x32, 64x64 and 128x128. Normal implementations of BGAN [4] and DCGAN [11] models are not compatible with those output shape. Therefore additional layers added to model implementations. The vanilla implementation of the BGAN contains block of fully connected dense layers and batch normalization layers. The output images have 1 channel and shape of 28 x 28. Since the desired output of face images are 32x32 with 3 channels, the BGAN model should be able to generate detailed information. Thus, additional dense layers added to both generator and discriminator parts of the generative model. The demonstration of the architecture of BGAN is given in the Figure 2

Additional operations also made for DCGAN to generate output images in the desired shapes. The implementation given in the paper of DCGAN [11] consists of fully connected convolutional layers. Only the discriminator part have a dense layer with softmax activation to generate class probabilities. The implementation generates images in the shape of 128x128 with 3 channels. To be able to generate images in smaller sizes, a convolutional layer with 1x1 filters and stride equal to 1 is used. This layer halves the output shape. The architecture proposed in the paper [11] of DCGAN is given in the Figure: 3

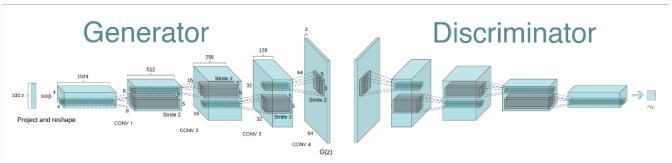

Fig. 3. The architecture of DCGAN [11]

The quality of the generated images were very different between the GAN implementations. The BGAN inputs had noises. There are many pixels in those images which have very different values from their neighbors. The main reason for that is a fully connected neural network with only dense layers used in the project cannot able to represent the patterns in the neighborhoods. For the BGAN model, it may be possible generated images with the same quality as DCGAN with deeper architectures. However, it will be waste of resources and time since that kind of architectures will require them more. Some of the generated images can be found in the Figure 4. The images in the upper row belongs to the DCGAN implementation and the below row images generated by BGAN implementation. It is clear that below images contains more noise.

Fig. 4. Images from GAN generated imageset. First row were generated by DCGAN and second row generated by BGAN model.

B. Feature Extractor Implementation

Convolutional Neural Networks used as feature extractors in the project. It is known that using pretrained models always boosts the training procedure. Rather than starting from random initialization, weights are loaded with a trained model's values. Furthermore, using closer domains for fine tuning boosts the model more than using a completely different domain. Because of these addressed issues, in the project a VGG16 model trained on VGGFace dataset [10] is used. Unfortunately, the pretrained weights of VGGFace model cannot used for the images whose size are 32x32 because of the incompatible input shape. For those images, a VGG16 model pretrained with ImageNet dataset is used. This did not effect the experiments, since both model was originated from VGG16 model. Only difference between them is the initial weights.

The feature extractor model and trained weights are obtained from a GitHub repository [8].

C. Implementation of Classifiers

In order to classify images, feature maps obtained from the feature extractor networks is used. These feature maps consists of 512 features and no dimensionality reduction is applied. Furthermore, the pretrained weights of the VGG16 model with VGGFace dataset is used as feature extractor. In order to compare the implementations of GANs, fully connected neural network classifier is used. This network trained and tested with the images whose size is 32x32 or 64x64. Moreover, in order to find the best classifier for discriminating GAN generated images, a wide set of classifiers tried. SVMs, K nearest neighbor algorithm, Random Forest classifiers and neural networks are the model that have been analyzed. These models trained and tested with the images generated by DCGAN and have a size of 128x128.

IV. Experiments and Results

Throughout the project, many experiments has been made. In this section the results of these experiments will be mentioned. Before going into further detail, evaluation metrics should be specified. In every classification problem, accuracy is the first metric that comes into one's mind. Although it can be misleading to use accuracy metric in unbalanced datasets, it is very common and mostly sufficient. The test datasets used in the project have nearly same number of samples. Therefore there was no harm to use accuracy as a metric. Other than accuracy False Acceptance Rate (FAR) and False Rejection Rate (FRR) metrics were used. This metrics are very useful if the system needs a significant type of security. For example, it is possible for a system to be more tolerable to the errors that have been made by labeling real images as GAN generated than the errors that have been made by labeling GAN generated images as real.

FAR is the rate of the fake images tagged as real to the total number of fake images. Likewise, FRR told us that among the real images how many of them tagged as generated. The formula of them is given below:

$$FAR = \frac{\#generated \quad images \quad tagged \quad as \quad real}{\# \quad generated \quad images}$$

$$FRR = \frac{\#real \quad images \quad tagged \quad as \quad generated}{\# \quad real \quad images}$$

Result of the experiments were given in terms of FAR, FRR and accuracy. The results were given in the Table I. In order to create this table, feature extractor model described in the III-B section and a neural network consists of 2 hidden layers with 64 neurons each is used. The model trained for 20 epoch for each dataset and stopped earlier if validation loss does not change more than 10^{-4} for 10 epochs. It should be noted that no training process last until 20 epochs. Therefore each of the models were learned the dataset enough.

Metrics	32x32 output size		64x64 output size	
	BGAN	DCGAN	BGAN	DCGAN
FAR	0.0 %	6.64%	0.6%	9.7%
FRR	0.03%	1.9%	0.4%	7.4%
Accuracy	99.98%	95.67 %	99.4%	91.38%

TABLE I
OBTAINED RESULTS FOR BOTH GAN IMPLEMENTATIONS.

A. Quality of the Images

The first work package of the project after collecting real face imageset was generating the fake images. As previously mentioned in the section III-A, BGAN and DCGAN models used for this purpose. Again as mentioned in the section III-A, quality of the outputs of BGAN were worse than DCGAN. However, this conclusion was acquired by looking the outputs only. The images of BGAN have more noise than the ones generated by DCGAN. This difference may made huge difference when tested with humans yet it is important to make systematically experiment with computers.

Obtained results from the classifiers shows that our initial conclusion was correct. In the table I, it is clear that classifier models did very well on BGAN imagesets. However, their performance is not same with the DCGAN imagesets.

We conclude that DCGAN implementation have better to represent human features and generated images that are more likely to be human. The main reason behinds this may be the architectures of the models. Fully convolutional architecture of the DCGAN is better than the fully dense architecture of the BGAN in terms of representing humanlike features.

B. Effect of the Image Size

From the results given in the Table I, it is clear that increasing output size of generated images decrease the performance of the classifier system. Since more information can be kept in the high resolution images, GAN generated images will have better features to represent genuine images too. This situation also noted in the paper proposing a new way of training generative adversarial models (PGGAN) [5] in which a high resolution images were generated by GANs.

In the experiments, classifier model have worse performance on the images with the output size of 64x64 then the images whose output size 32x32. This shows that more detailed imposter images can imitate genuine images better.

C. Performance of Classifier Models

The most important part of the project was building a reliable and robust system to discriminate real images from generated ones. The last step of the process was selecting a suitable classifier. In order to select the perfect classifier, feature maps obtained from the feature extractor CNN described in the section III-B, tried to be classified with the Random Forest Classifier, SVM, KNeighbors Algorithm and Neural Networks. Evaluation made based on the models FAR, FRR and accuracy scores. The whole results is given in the Table II.

In the experiment the images with the size of 128x128 and generated by DCGAN is used. Images from DCGAN implementation is used because previously they proved that they are better to imitate genuine images than the images generated by BGAN. Furthermore, it is not possible to compare DCGAN and BGAN with the size of 128x128 since no image with that size is generated by BGAN.

Metrics	Classifier Model			
	Random Forest	SVM	KNeighbors	Neural Net
FAR	9.45 %	6.40%	6.36 %	5.90 %
FRR	6.01 %	4.92 %	6.23 %	5.23 %
Accuracy	92.25 %	94.33 %	93.7 %	94.44 %

TABLE II
OBTAINED RESULTS FROM CLASSIFICATIONS.

The accuracy results show that each classifier can be used in the system without losing too much reliability. However, the FAR and FRR scores also must be considered while selecting classifiers. Random Forest classifier fooled by GAN generated images relatively more times than other models. Therefore Random Forest classifier is the only model that one may not want to use.

Remaining three models can be used in the system to discriminate GAN generated images, yet it is important to not the best performances. In general performance, created Neural Network achieved best results. Second best performance achieved by SVMs. Also SVMs have the best FRR rate meaning that they were better to understanding real images.

V. CONCLUSION

In the project a system that can discriminate GAN generated images from the genuine images were created. Firstly, the genuine images were collected and imposter images were created using BGAN and DCGAN implementations. To be able to classify the images, a model combination of a feature extractor and a classifier was used. The feature extractor model was a fully connected convolutional neural network which was trained on VGGFace dataset before. As a classifier, different kinds of classifiers were tried and most appropriate one is selected. Quality of the outputs of the GAN implementations, effect of image sizes to the classification results and varying classifier results were addressed issues in the project. The results and comments were given about these issues.

In the project the created system proved its performance on BGAN and DCGAN implementations. However, generative models keep changing everyday and generated images getting better. It is necessary to updating the systems to keep pace with these developments.

REFERENCES

[1] Martin Arjovsky, Soumith Chintala, and Lon Bottou. Wasserstein gan, 2017.

[2] Tai Do Nhu, In Na, and S.H. Kim. Forensics face detection from gans using convolutional neural network, 10 2018.

[3] Ian J. Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial networks, 2014.

[4] R Devon Hjelm, Athul Paul Jacob, Tong Che, Adam Trischler, Kyunghyun Cho, and Yoshua Bengio. Boundary-seeking generative adversarial networks, 2017.

[5] Tero Karras, Timo Aila, Samuli Laine, and Jaakko Lehtinen. Progressive growing of gans for improved quality, stability, and variation, 2017.

[6] Haodong Li, Bin Li, Shunquan Tan, and Jiwu Huang. Detection of deep network generated images using disparities in color components, 2018.

[7] Ziwei Liu, Ping Luo, Xiaogang Wang, and Xiaoou Tang. Deep learning face attributes in the wild. In *Proceedings of International Conference on Computer Vision (ICCV)*, 2015.

[8] Refik Can Malli. Vggface model, 2016.

[9] Scott McCloskey and Michael Albright. Detecting gan-generated imagery using color cues, 2018.

[10] O. M. Parkhi, A. Vedaldi, and A. Zisserman. Deep face recognition. In *British Machine Vision Conference*, 2015.

[11] Alec Radford, Luke Metz, and Soumith Chintala. Unsupervised representation learning with deep convolutional generative adversarial networks, 2015.

[12] Junbo Zhao, Michael Mathieu, and Yann LeCun. Energy-based generative adversarial network, 2016.