Deep Fake Image Detection

Zehao Hui
hzh98@bu.edu

Zhaoyuan Fu
fuzy@bu.edu

Haoxiang Sun
shx95@bu.edu

Abstract

The advent of deep learning has led to dramatic improvements in the quality of generated media. Techniques such as Deepfake have achieved great results in the field of generated images, which are also making it more difficult for human viewers to distinguish between real images and generated ones. Along with this is the ongoing phenomenon of prevalent scams on the Internet. Under such circumstances, it is critical to develop and utilize tools to assist the resolution of the generated images. In this project, we first analyze existing detection methods, then re-implement well-performed detecting models from two types of mainstream detection methods, train on different objects and evaluate their performance, and finally perform Ensemble Learning to take the advantages of both detection approaches, improve the detection accuracy, and thus create a more robust and generalizing method of fake image detection.

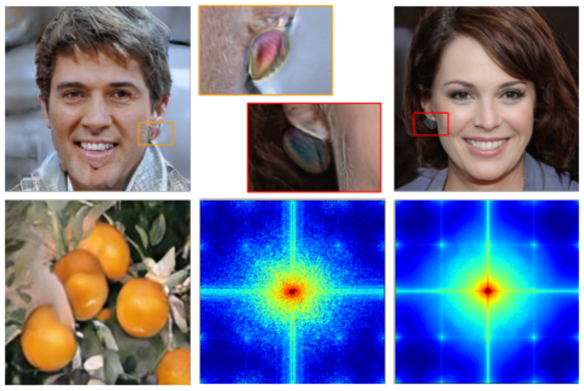
Figure 1: Examples of GAN synthetic images with their features. Top: Spatial Domain Features. Bottom: Frequency Domain Features

1 Task

In recent years, people have proposed and implemented a large number of methods for artificially synthesizing pictures and media files based on deep learning. Generative Adversarial Networks (GANs) in particular have brought huge quality improvements. GANs enable the possibilities to regenerate images as well as modify existing ones. Based on these functions, some practical software or programs are gradually developed, such as improving the clarity of pictures, or intelligently retouching pictures.

However, the technology can also be used for malicious purposes, such as generating fake profiles on social networks or generating fake news. Users could easily be confused by GAN-generated images because they may differ from real images by only a small amount. Therefore, it is urgently needed for automated tools that can reliably distinguish between authentic and manipulated content.

Naturally, our task is fake image detection: given an input image, the model output whether the image is authentic or generated.

2 Related Work

A "contrastive learning based approach" has been raised in (Cozzolino et al., 2021), which aims to make the method more generalized and robust in practical scenarios. The performance of this method was tested against other methods to illustrate its merits.

(Gragnaniello et al., 2021) and (Gragnaniello et al., 2022) together pointed out three state-of-the-art methods for GAN image detection: Learning spatial domain features, Learning frequency domain features and Learning features that generalize, and introduced, tested and compared several different GAN detectors from these three methods. They also brought up certain vulnerabilities. In the first method of learning spatial domain features, an example is extracting artificial fingerprints from each individual GAN and attribute a generated image based on the fingerprint it contains. This method has proven its effectiveness, however, it only applies to the specific GAN that the fingerprint was extracted from, thus preventing the method to generalize towards different GAN architectures. In the second method of learning frequency domain features, e.g., learning on Fourier spectrum, the artificial traces hidden in the spectrum can also be remedied by purposely enhancing it, and leads to the performance decrease of this method.

3 Approach

Based on the disadvantages discussed above, we propose to combine the two methods of learning spatial and frequency domain features together, i.e., to combine models that take these two approaches through Ensemble Learning, and create a method that take advantages from both spatial and frequency domain.

To start with, we begin by re-implementing well-performed detectors that belong to above two methods, to gain further understanding of what aspects of these different models can impact on the performance of GAN image detection. Detectors based on very deep networks that belong to the first method are introduced in (Marra et al., 2018) as follows. Detector from the second method of learning frequency domain features is a model that takes image spectrum as input. We also propose a new model by providing another type of spectrum input to this model.

3.1 Learning Spatial Domain Features

3.1.1 Xception

This network (Chollet, 2017) brings to the extreme some ideas of Inception (Szegedy et al., 2015) like the use of multi-resolution representations of the input, adopting fully separable filters. In each layer, the 3d filtering of the input feature maps is implemented as a 1d depthwise convolution followed by a 2d pointwise convolution.

3.1.2 DenseNet

It is a very deep Convolutional Neural Network (CNN) architecture (Huang et al., 2017), based on the residual network (ResNet) proposed in (He et al., 2016), which pursues effectively feature propagation and reuse within the network. Unlike in early CNN architectures, for each layer of DenseNet, the feature-maps of all preceding layers are used as inputs, and its own feature-maps are used as inputs into all subsequent layers.

Despite the deepness of these two CNN architectures, they all greatly improved their performance

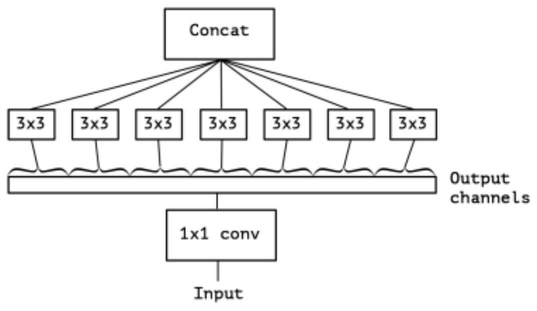

Figure 2: An "extreme" version of Inception module, with one spatial convolution per output channel of the 1x1 convolution

through vastly reducing the number of parameters to learn.

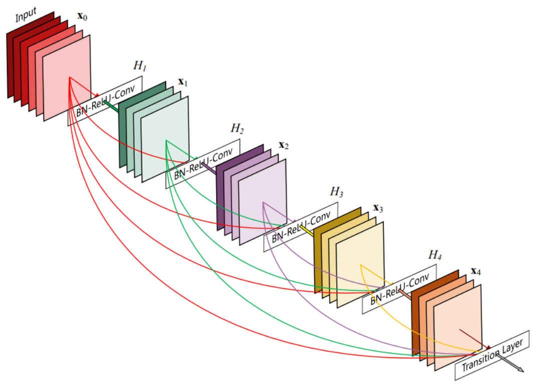

Figure 3: A 5-layer dense block with a growth rate of k = 4. Each layer takes all preceding feature-maps as input.

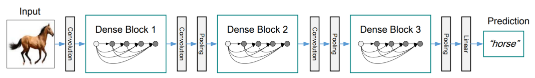

Figure 4: A deep DenseNet with three dense blocks. The layers between two adjacent blocks are referred to as transition layers and change feature-map sizes via convolution and pooling.

3.2 Learning Frequency Domain Features

3.2.1 Spec

This detector proposed in (Zhang et al., 2019) is based on pre-trained ResNet34 with image Fast Fourier Transform (FFT) spectrum as input, which exploits the presence of spectral peaks caused by the upsampling operations routinely performed in most GAN architecture.

FFT, known as Fast Fourier Transform, is an algorithm that computes the Discrete Fourier Transform (DFT) of a sequence, or its inverse (IDFT). Fourier analysis converts a signal from its original domain (often time or space) to a representation in the frequency domain and vice versa.

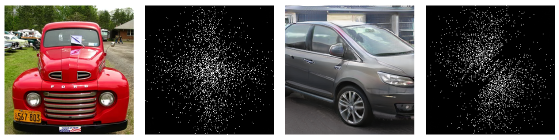
Figure 5: FFT of real and fake images. Fake image has shown some unnatural groove in the center.

Assume the low-resolution signal $x(n)$, $n = 0, ..., N-1$, $X(k) = \sum_{n=0}^{N-1} x(n) exp(\frac{-i2\pi}{N}kn)$, By inserting 0, we have a 2N-point sequence $x^{'}(n)$,$n = 0, ..., 2N-1$, where $x^{'}(2n) = x(n)$ and $x^{'}(2n+1) = 0$ for $n = 0, ..., N-1$. Assume the DFT of $x^{'}(n)$ is $X^{'}(k)$. For $k < N$, considering the inserted 0,

$$X^{'}(k) = \sum_{n=0}^{2N-1} x^{'}(n) exp(\frac{-i2\pi}{2N}kn)$$

$$= \sum_{n=0}^{N-1} x(n) exp(\frac{-i2\pi}{2N}k(2n))$$

For $k \geq N$, let $k^{'} = k - N$, thus $k^{'} = 0, ..., N-1$, and,

$$X^{'}(k) = \sum_{n=0}^{N-1} x(n) exp(\frac{-i2\pi}{2N}(k^{'}+N)(2n))$$

$$= \sum_{n=0}^{N-1} (x(n) exp(\frac{-i2\pi}{N}nk^{'} - i2n\pi))$$

$$= X(k^{'})$$

3.2.2 DCT-Spec

We propose the 4-th model by providing another type of spectrum input, i.e., Discrete Cosine Transform (DCT), introduced in (Frank et al., 2020), into the pipeline of Spec.

DCT, known as Discrete Cosine Transform, is another transform method that only uses cosine functions to express signals, which is closely related to Fourier transform. Because there is a large spatial correlation between image pixels, DCT can greatly reduce these correlations, so that the image energy is concentrated in the upper left area. The data obtained after transformation are so-called DCT coefficients.

More formally, let and imput image be given by the matrix $I \in \mathbb{R}^{N_1 \times N_2}$, where the entries (specifying the pixel values) are denoted by $I_{x,y}$, and its DCT-transformed representation by the matrix $D \in \mathbb{R}^{N_1 \times N_2}$. The 2D-DCT is given by a function $\mathcal{D} : \mathbb{R}^{N_1 \times N_2} \to \mathbb{R}^{N_1 \times N_2}$ that maps and image $I = \{I_{x,y}\}$ to its frequency representation $D = \{D_{k_x,k_y}\}$, with

$$D_{k_x,k_y} = w(k_x)w(k_y) \sum_{x=0}^{N_1-1} \sum_{y=0}^{N_2-1}$$

$$(I_{x,y} cos[\frac{\pi}{N_1}(x+\frac{1}{2})k_x] cos[\frac{\pi}{N_2}(y+\frac{1}{2})k_y])$$

for $\forall k_x = 0, 1, 2, ..., N_1 - 1$ and $\forall k_y = 0, 1, 2, ..., N_2 - 1$, and where $w(0) = \sqrt{\frac{1}{4N}}$ and $w(k) = \sqrt{\frac{1}{2N}}$ for $k > 0$.

3.3 Ensemble Learning

After implementing above four detectors, we perform Ensemble Learning to combine their results together for a more robust and generalizing method of fake image detection.

We choose Voting Classifier for the ensemble method. A Voting Classifier is a machine learning model that trains on an ensemble of numerous models and predicts an output (class) based on their highest probability of chosen class as the output. It simply aggregates the findings of each classifier passed into Voting Classifier and predicts the output class based on the highest majority of voting. The idea is instead of creating separate dedicated models and finding the accuracy for each them, we create a single model which trains by these models and predicts output based on their combined majority of voting for each output class. Voting Classifier supports two types of voting: Hard voting and Soft voting.

3.3.1 Hard Voting

In hard voting, the predicted output class is a class with the highest majority of votes i.e the class which had the highest probability of being predicted by each of the classifiers. Suppose three classifiers predicted the output class(A, A, B), so

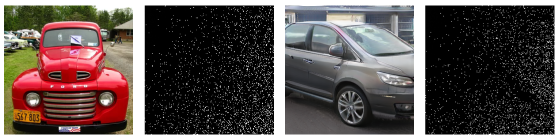
Figure 6: DCT of real and fake images, same as FFT. Fake image has also shown some unnatural groove in the right-bottom corner.

here the majority predicted A as output. Hence A will be the final prediction.

3.3.2 Soft Voting

In soft voting, the output class is the prediction based on the average of probability given to that class. Suppose given some input to three models, the prediction probability for class A = $(0.30, 0.47, 0.53)$ and $B = (0.20, 0.32, 0.40)$. So the average for class A is 0.4333 and B is 0.3067, the winner is clearly class A because it had the highest probability averaged by each classifier.

Thanks to our models' capabilities of generating probability on each class (real / fake), we choose the Soft Voting method for our Voting Classifier.

4 Datasets

We adapted datasets from (Wang et al., 2020) as our datasets, to be described as follows.

4.1 Training Set

The available training set comprising 362K real images extracted from the LSUN dataset (Yu et al., 2015), equally divided into 20 semantic categories: Airplane, Bicycle, Bird, Boat, Bottle, Bus, Car, Cat, Chair, Cow, Dining Table, Dog, Horse, Motorbike, Person, Potted Plant, Sheep, Sofa, Train and TV Monitor. And the same number of 362K fake images are generated by 20 ProGAN models, each trained on a the corresponding LSUN semantic category. All images have a resolution of 256×256.

4.2 Validation Set

A subset of 4K images from the training set is picked out for validation.

4.3 Test Set

We utilize a major part of the available test set, and it consists of many state-of-the-art GANs in addition to the 20 semantic categories from ProGAN: CycleGAN(Zhu et al., 2017) with semantic category (Apple, Orange, Horse, Zebra, Summer, Winter), StyleGAN(Karras et al., 2019) with semantic category (Bedroom, Car, Cat), StyleGAN2(Karras et al., 2020) with semantic category (Car, Cat, Church, Horse), three face GANs (Deepfake(Rossler et al.), StarGAN(Choi et al., 2018), WFIR(wfi)), and two GANs that provide images of different objects (BigGAN(Brock et al., 2018), Second Order Attention Network (SAN)(Dai et al., 2019)). These images from different GANs vary in their sizes.

4.4 Pre-processing

For FFT images, given an image as input, we apply the 2D DFT to each of the RGB channels and get 3 channels of frequency spectrum F, and discard the phase information. Then we compute the logarithmic spectrum $log(F)$ and normalize it to $[1, 1]$. Finally we output the spectrum as an image as our Spec input.

The DCT follows similar pipeline but with DFT changed to DCT.

5 Evaluation Metrics

We use Cross Entropy Loss for all of our models, defined as

$$loss(x, class) = -log(\frac{exp(x[class])}{\sum_j exp(x[j])})$$
$$= -x[class] + log(\sum_j exp(x[j]))$$

And we use accuracy as our evaluation metric, defined as

$$\text{Accuracy} = \frac{\text{\# of images correctly classified}}{\text{\# of all images}}$$

We evaluate our models in four perspectives: Performance on the same semantic category from ProGAN and its overall average, and on similar categories from unseen different GANs and its overall average.

6 Results

6.1 Training Settings

We only re-implement the architecture of these models, without applying the same transforms as they deployed to their datasets in their original paper, for the reason that we want to have a relatively faster training speed within our limited computing resources, and uniformed training settings enable us to compare and analysis their results in a more reasonable way. We also don't utilize the common pretrain technique as we believe the pretrained models have already learned abundant information on the pretrain datasets, resulting the models harder to learn on our datasets in a short period of time.

Specifically, we apply the transforms of (Random Horizontal Flip, Resize to 299×299) for Xception, because some modifications on images have proven to improve models' robustness, and its architecture allows larger images, which can contribute to better performance. For the DenseNet, we changed the resize transform from 299×299 to 32×32 because the original DenseNet was trained on datasets with this size. For the Spec and the DCT-Spec, we only resize the pre-processed images to 224×224 to preserve as much spectrum information as possible.

After examining available datasets, we decide to train our four models on 5 semantic categories which have both corresponding category in ProGAN and similar categories in other GANs in the test set: Car - Car in ProGAN / StyleGAN / StyleGAN2, Cat - Cat in ProGAN / StyleGAN / StyleGAN2, Horse - Horse in ProGAN / CycleGAN / StyleGAN2 and Zebra in CycleGAN, Person - face images in Deepfake / StarGAN / WFIR, Sofa - Sofa in ProGAN and Bedroom in StyleGAN. We train them separately, i.e., for each model, we train different 5 actual models solely on one category, and this gives us 20 actual models in total. Then we perform Ensemble Learning on all four models within each category, which gives us 5 ensembled classifier in the end.

We use the batch size of 16 for all models. The optimizer for all models is SGD with momentum and the starting learning rate is 0.001, with a $20\times$ smaller learning rate for the last layer, and weight decay is at the same magnitude of 0.001 and $1e-5$, respectively. We also adapt the idea of Early Stopping by saving the model that performs the best on validation set within our 15 epochs. It is worth noticing that our goal isn't to maximize a specific model's performance, but rather to verify the effectiveness of Ensemble Learning on combining advantages from different approaches together.

6.2 Averaged Results

Our full test data are available in our code repository. Here we take the overall average for our analysis.

Average Accuracy	ProGAN (same category)	ProGAN (overall)	Different GANs (similar category)	Different GANs (overall)
Xception	99.2	90.2	68.36	64.21
DenseNet	73.4	60.23	54.09	50.93
Spec	83.8	74.03	55.81	49.15
DCT-Spec	87.4	76.74	56.72	51.37
Ensemble	98	86.99	66.87	55.34

Table 1: Average Accuracy

From the table it's obvious that the accuracy of all models decrease as they move from same category in same GAN, to different category, to different GANs but similar categories, and to different GANs in different categories. In our data, even for the best performing Xception, ensemble learning (69.15%) still outperforms the corresponding Xception model (66.95%) in the fake category of sofa.

7 Conclusion

Models perform best when tested on the same GAN within same category.

When they encounter unseen GANs, they are highly likely to still perform decently on similar categories, but may not work as well on unseen GANs and unseen categories.

Soft Ensemble Learning can boost overall performance based on each model's capability. And it can also use some models' advantages to make up for others' disadvantages, as we've seen above for the Xception.

To further improve our method, models can be fine-tuned, more models can be added, data augmentation can be deployed. In conclusion, we prove that the Ensemble Learning can successfully take advantages from different models from different approaches, and make it more robust and generalizing to act as a universal GAN detector.

References

Which face is real?

Andrew Brock, Jeff Donahue, and Karen Simonyan. 2018. Large scale gan training for high fidelity natural image synthesis. *arXiv preprint arXiv:1809.11096*.

Yunjey Choi, Minje Choi, Munyoung Kim, Jung-Woo Ha, Sunghun Kim, and Jaegul Choo. 2018. Stargan: Unified generative adversarial networks for multi-domain image-to-image translation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 8789–8797.

François Chollet. 2017. Xception: Deep learning with depthwise separable convolutions. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1251–1258.

Davide Cozzolino, Diego Gragnaniello, Giovanni Poggi, and Luisa Verdoliva. 2021. Towards universal gan image detection. In *2021 International Conference on Visual Communications and Image Processing (VCIP)*, pages 1–5. IEEE.

Tao Dai, Jianrui Cai, Yongbing Zhang, Shu-Tao Xia, and Lei Zhang. 2019. Second-order attention network for single image super-resolution. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 11065–11074.

Joel Frank, Thorsten Eisenhofer, Lea Schönherr, Asja Fischer, Dorothea Kolossa, and Thorsten Holz. 2020. Leveraging frequency analysis for deep fake image recognition. In *International Conference on Machine Learning*, pages 3247–3258. PMLR.

Diego Gragnaniello, Davide Cozzolino, Francesco Marra, Giovanni Poggi, and Luisa Verdoliva. 2021. Are gan generated images easy to detect? a critical analysis of the state-of-the-art. In *2021 IEEE International Conference on Multimedia and Expo (ICME)*, pages 1–6. IEEE.

Diego Gragnaniello, Francesco Marra, and Luisa Verdoliva. 2022. Detection of ai-generated synthetic faces. In *Handbook of Digital Face Manipulation and Detection*, pages 191–212. Springer, Cham.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778.

Gao Huang, Zhuang Liu, Laurens Van Der Maaten, and Kilian Q Weinberger. 2017. Densely connected convolutional networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4700–4708.

Tero Karras, Samuli Laine, and Timo Aila. 2019. A style-based generator architecture for generative adversarial networks. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 4401–4410.

Tero Karras, Samuli Laine, Miika Aittala, Janne Hellsten, Jaakko Lehtinen, and Timo Aila. 2020. Analyzing and improving the image quality of stylegan. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 8110–8119.

Francesco Marra, Diego Gragnaniello, Davide Cozzolino, and Luisa Verdoliva. 2018. Detection of gan-generated fake images over social networks. In *2018 IEEE Conference on Multimedia Information Processing and Retrieval (MIPR)*, pages 384–389. IEEE.

A Rossler, D Cozzolino, L Verdoliva, L Verdoliva, C Riess, J Thies, and M FaceForensics+ Nießner. Learning to detect manipulated facial images. arxiv 2019. *arXiv preprint arXiv:1901.08971*.

Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. 2015. Going deeper with convolutions. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1–9.

Sheng-Yu Wang, Oliver Wang, Richard Zhang, Andrew Owens, and Alexei A Efros. 2020. Cnn-generated images are surprisingly easy to spot... for now. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 8695–8704.

Fisher Yu, Ari Seff, Yinda Zhang, Shuran Song, Thomas Funkhouser, and Jianxiong Xiao. 2015. Lsun: Construction of a large-scale image dataset using deep learning with humans in the loop. *arXiv preprint arXiv:1506.03365*.

Xu Zhang, Svebor Karaman, and Shih-Fu Chang. 2019. Detecting and simulating artifacts in gan fake images. In *2019 IEEE International Workshop on Information Forensics and Security (WIFS)*, pages 1–6. IEEE.

Jun-Yan Zhu, Taesung Park, Phillip Isola, and Alexei A Efros. 2017. Unpaired image-to-image translation using cycle-consistent adversarial networks. In *Proceedings of the IEEE international conference on computer vision*, pages 2223–2232.

A Detailed Roles

Name	Task	File names	No. Lines of Code
Zehao Hui	Implement Xception and DenseNet, implement ensemble learning, train and test 20% of the data, aggregate and organize the results, write parts of the report	DenseNet.ipynb, Xception.ipynb, Ensemble_ Learning.ipynb, ResNet34DCT_sofa_best.pt, Test Result_final.xlsx, Test Result Ensemble Report.pdf	300
Zhaoyuan Fu	Implement Spec and DCT-Spec, train and test 70% of the data, analyze data results, write parts of the report, do final project presentation	DCT-Spec.ipynb, Spec.ipynb all other ".pt" files, Test Result_final.xlsx, Test Result Others Report.pdf	500
Haoxiang Sun	Train and test 10% of the data, aggregate and organize the results	Xception.ipynb, Test Result_final.xlsx	50

B Code repository

Our codes are available at https://github.com/FaustineHui/EC523_finalproject

Acknowledgements: We adapted Xception, DenseNet and the pipelines of training and testing from https://github.com/amilworks/GanDetection under the MIT license. Majority of our work have done on pre-processing, training, Ensemble Learning, and testing results.