Fake Face Detection via Deep Learning

Emily Mazor, Ran Li
Department of Computer Science
emily.mazor@mail.utoronto.ca, layla.li@mail.utoronto.ca

Abstract

Our paper will focus on facial image recognition. Specifically, we want our neural networks to distinguish between real and GAN-generated faces. We use VGG-16 [9] (pretrained on ImageNet) and a CNN with a high-pass filter [8] (which we will refer to as CNN-HPF) and perform sensitivity analysis on their hyperparameters. We extend the algorithms to two new datasets, one which contains real and StyleGAN2-generated faces [1], and one which contains real and PGGAN-generated faces. Our goal in applying the CNNs to this data is to detect whether a face is real or fake, and to generalize these algorithms beyond ImageNet.

1 Introduction

The rapid development of face manipulation technology makes the manufacture of fake face more accessible than before [2]. Generative adversarial networks (GANs) are widely applied in fake face generation [10]. High quality GAN-generated faces often cannot be distinguished as fake by the human eye.

The advancement of deep learning provides new techniques for detection. CNNs are good at extracting and summarizing features of a given image that cannot be observed by human eyes. In this paper, we practice two methods in fake face detection. The first model uses pretrained VGG-16 [9] and the second model is a CNN with a high pass filter [8] (CNN-HPF). We compare the performance of these two models and investigate whether the type of GAN used to generate fake data can influence performance of the models.

2 Fake Face Generators

Generative modeling is an unsupervised learning task in machine learning that involves automatically discovering and learning the regularities or patterns in input data in such a way that the model can be used to generate or output novel samples that plausibly could have been drawn from the original data set [3].

PGGAN

The generation of high-resolution images is difficult because higher resolution makes the discriminator easier to tell the generated images apart from the training images [5]. Progressive growing of GANs (PGGAN) proposed by Terro Karras, et al. from NVIDA in 2017 [5] is a solution to the problem of training stable GAN models for larger images through progressively increasing the number of layers during the training process. Instead of training the whole GAN on high-resolution image, they first construct a simple GAN training on low-resolution images in the beginning, and then gradually add more layers to adapt the model to high-resolution images during the training stage. The incremental layer approach allows the model to learn coarse detail first and learn finer detail step by step, instead of learning all scales of details simultaneously [5]. See Figure 1 for illustration of this model with the final block (consisting of a fully-connected layer and Softmax) proposed by Nhu et al. [2], which

Preprint. Under review.

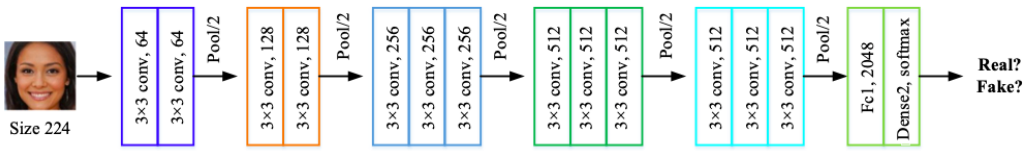


Figure 1: VGG Structure

we use in our training. The figure is from Liu et a [11], which further explores VGG, Deep FG, and CDNN.

StyleGAN
The Style Generative Adversarial Network is an extension to the GAN architecture to give control over the disentangled style properties of generated images. Its generator starts from a learned constant input and adjusts the style of the image at each convolutional layer based on the latent code, thus controlling the strength of the image features at different scales [6].

3 Detection Methods

Deep face recognition systems usually contain three main modules: face processing, deep feature extraction and face matching [2].

VGG
Visual Geometry Group (VGG), proposed by A. Zisserman and K. Simonyan in 2014 [9], is a standard deep learning architecture with multiple layers. In practice, we use pre-trained weights from VGG-16 without the fully connected layer to get feature vectors with hidden size 512, then we perform fine-tuning on the classifier, as outlined in Nhu et al. [2]. We choose stochastic gradient descent (SGD) as optimizer.

Convolutional Neural Network with high pass filter
The second model that we build is based on the idea that the main difference between the two kinds of images would be reflected on the residual domain of the images [7].Therefore, we first perform a high pass filter to transform the original image into its residual domain. The resulting residuals are then fed into three groups where each group consists of a convolutional layer with kernel size 3 by 3, a LeakyReLU activation function and a max pooling layer with size 2 by 2 and stride 2. Note that the input channel is 3 (RGB color image) and after each group, we have channel doubled. After three groups, we flatten the features and feed them into two fully connected layer, equipped with LeakyReLU function. We have 2 units in the Output of fully connected layers and they are finally taken into the softmax function for output probability. See Figure 2 [8] for illustration of the architecture.

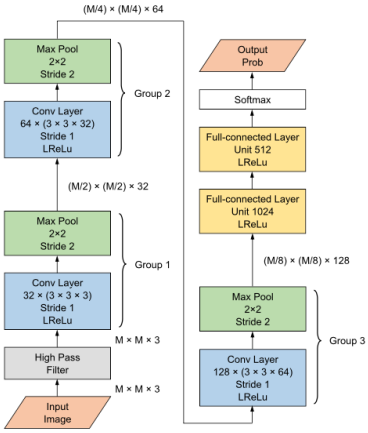


Figure 2: CNN-HPF Structure

We train this neural network using Adam optimizer. All the weights are initialized using a truncated Gaussian distribution with zero mean and standard deviation of 0.01. The high pass filter we use is $\begin{pmatrix} 0 & -1 & 0 \\ -1 & 4 & -1 \\ 0 & -1 & 0 \end{pmatrix}$.

4 Dataset

We use two datasets in our analysis. The first is the 'Fake-Vs-Real-Faces (Hard)' dataset from Kaggle [1]. It contains real facial images from CelebA and fake facial images generated with StyleGAN2. For our second dataset, we use the same real facial images from the Kaggle dataset, and PGGAN-

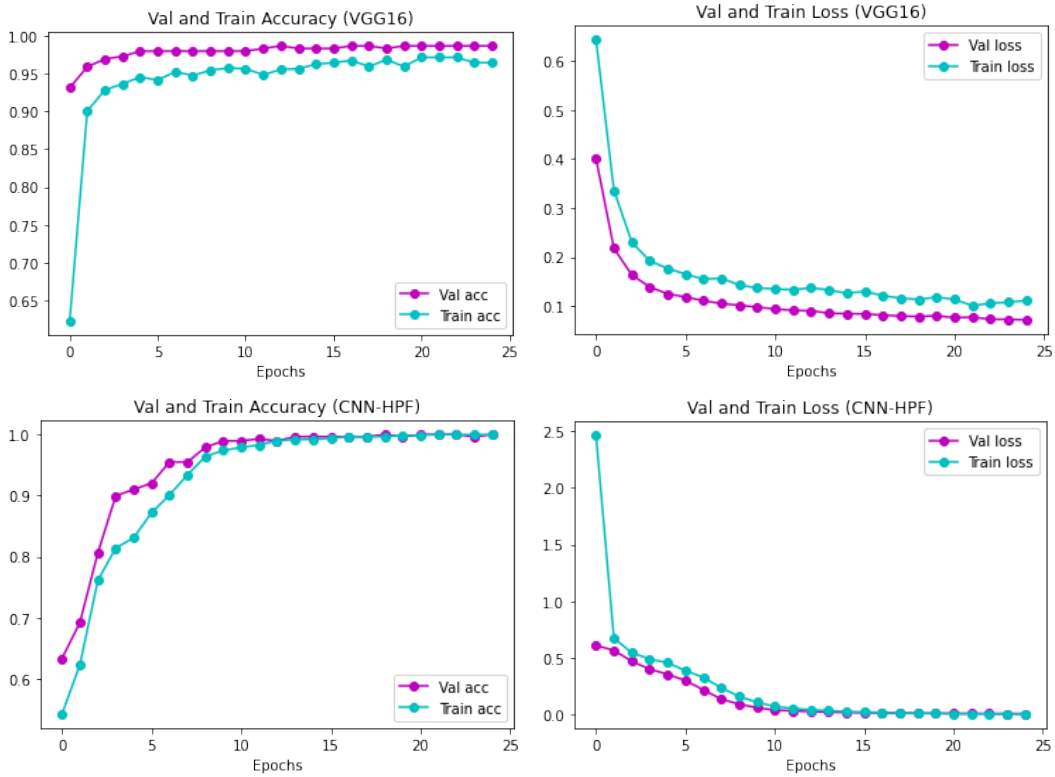
generated fake facial images which we generated using an implemented PGGAN face generator from Pytorch Hub [4].

5 Experiment Results

5.1 Hyperparameter sensitivity analysis

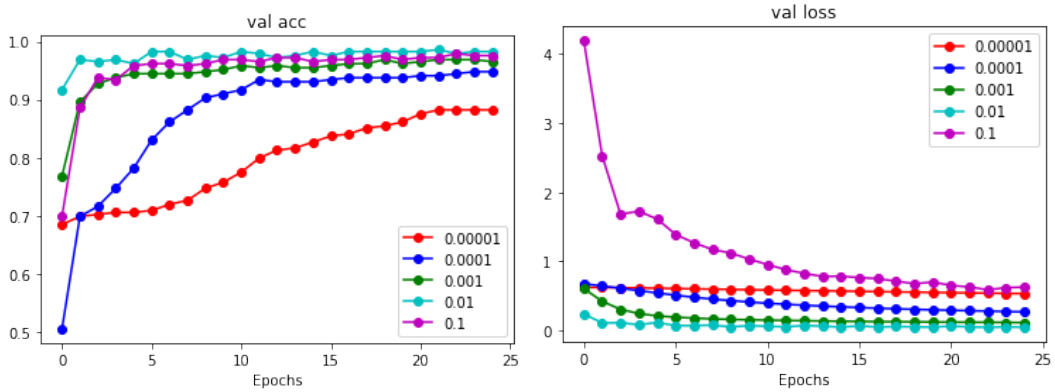
We tune the VGG16 and CNN-HPF hyperparameters on the StyleGAN2-generated faces as the fake face dataset.

Epochs



We run the VGG16 and CNN-HPF models for 25 epochs and observe the validation and training accuracy and loss. The accuracy and loss converge around 15-20 epochs for both models. For future training, we continue training for 25 epochs for good measure.

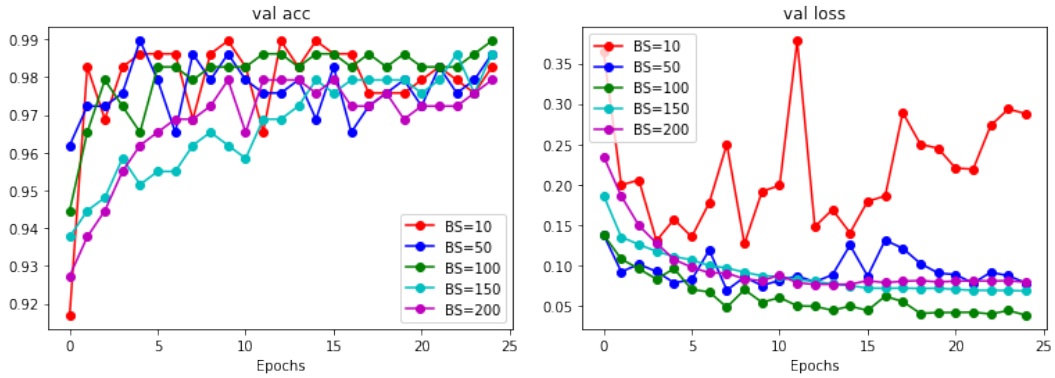
Learning rate



We train the VGG16 model with learning rates 0.00001, 0.0001, 0.001, 0.01, 0.1, each for 25 epochs.

The validation accuracy plot shows that learning rates of 0.1, 0.01, 0.001, and 0.0001 all converge to a very high accuracy of > 0.9. The validation loss graph shows that learning rates of 0.01 and 0.001 both converge to a loss of almost 0, with the other learning rates converging to values not much higher. Based on both validation loss and accuracy, however, a learning rate of 0.01 is clearly superior because 1) it reaches the highest accuracy and lowest loss, and 2) it converges the quickest. In fact, when using a learning rate of 0.01, the model converges after about 5 epochs. See Appendix A.1 for learning rate tuning using CNN-HPF. We find that the ideal learning rate for CNN-HPF is 0.001.

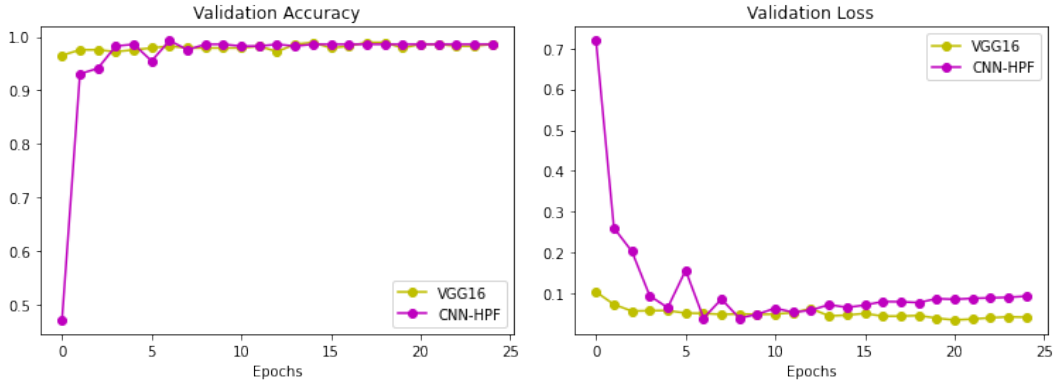
Batch size



We train the VGG16 model with batch sizes 10, 50, 100, 150, 200, each for 25 epochs. We find that in terms of validation accuracy, all the batch sizes reach an accuracy of > 0.97 after 25 epochs. Batch sizes 10, 50, and 100 all seem to reach the highest accuracy of 0.99, with batch size 50 reaching it after only 5 epochs. With a batch size of 100, however, the model is the most stable in terms of accuracy, staying above 0.98 after 5 epochs, while the other two batch sizes jump down to 0.96 between epochs 5 and 25. In terms of validation loss, batch size 100 is the clear winner. Thus, we will use batch size 100 for VGG16. See Appendix A.1 for batch size tuning using CNN-HPF. We find that there is no clear ideal batch size for CNN-HPF, so We will use batch size 100 in our training to stay consistent with VGG16.

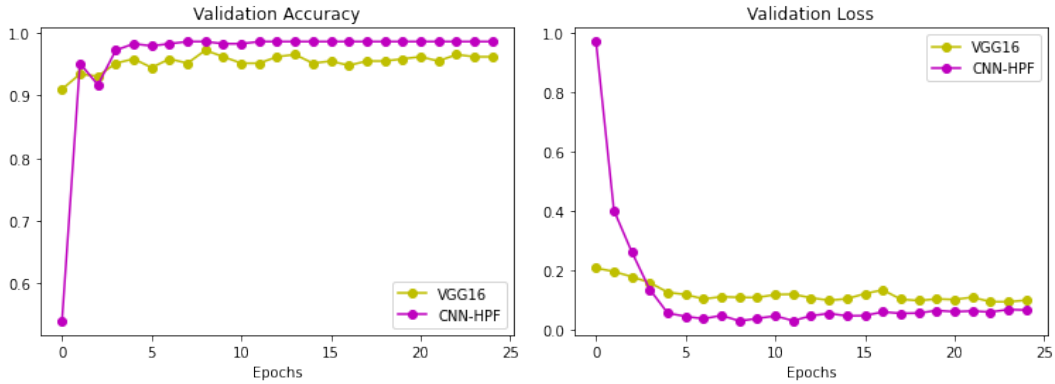
5.2 Comparing the detection models

StyleGAN2 dataset



We train both models using the tuned hyperparamers on the dataset consisting of real and StyleGAN2-generated faces. We observe that both models reach almost 100% accuracy and loss < 0.1. The main performance difference is that VGG16 begins with much higher accuracy and much lower loss than CNN-HPF. This is expected as our VGG16 model is pretrained, while CNN-HPF is not. Regardless, the two models have almost the same performance after CNN-HPF converges.

PGGAN dataset



We repeat the training, this time on the dataset with PGGAN-generated faces. The performance of both models is almost identical to their performance on the StyleGAN2 dataset.

6 Conclusion

Overall, we find that VGG16 and CNN-HPF are both exteremely effective at detecting fake vs real faces, both for the StyleGAN2 and PGGAN-generated fake faces. As evidenced by the graphs, the models have almost the exact same performance once they converge, with CNN-HPF outperforming VGG-16 ever-so-slightly, although the difference is almost negligible. We also find that there is not a big difference in model performance between the two datasets. This is interesting since the StyleGan2-generated faces are higher quality than the PGGAN-generated faces, so one might expect the model to perform better on the PGGAN dataset.

A topic for future research would be tuning models to distinguish between fake faces generated by different GANs, i.e., distinguishing between StyleGAN2 and PGGAN-generated faces. In our case, the faces we generated with PGGAN are lower quality than the StyleGAN2-generated faces and detecting between the two types can be done by the human eye for the most part. If the faces generated by both types of GANs are very high quality, however, the human eye will not be able to distinguish between them, so tuning neural networks to do that work is an interesting task to research.

References

[1] Hamza Boulahia. Fake-vs-real-faces (hard), Feb 2022.

[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] FAIR HDGAN. Progressive growing of gans (pgan).

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

[6] Tero Karras, Samuli Laine, and Timo Aila. A style-based generator architecture for generative adversarial networks, 2018.

[7] Haodong Li, Weiqi Luo, Xiaoqing Qiu, and Jiwu Huang. Identification of various image operations using residual-based features. *IEEE Transactions on Circuits and Systems for Video Technology*, 28(1):31–45, 2018.

[8] Huaxiao Mo, Bolin Chen, and Weiqi Luo. Fake faces identification via convolutional neural network. In *Proceedings of the 6th ACM Workshop on Information Hiding and Multimedia Security*, IHMMSec '18, page 43–47, New York, NY, USA, 2018. Association for Computing Machinery.

[9] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition, 2014.

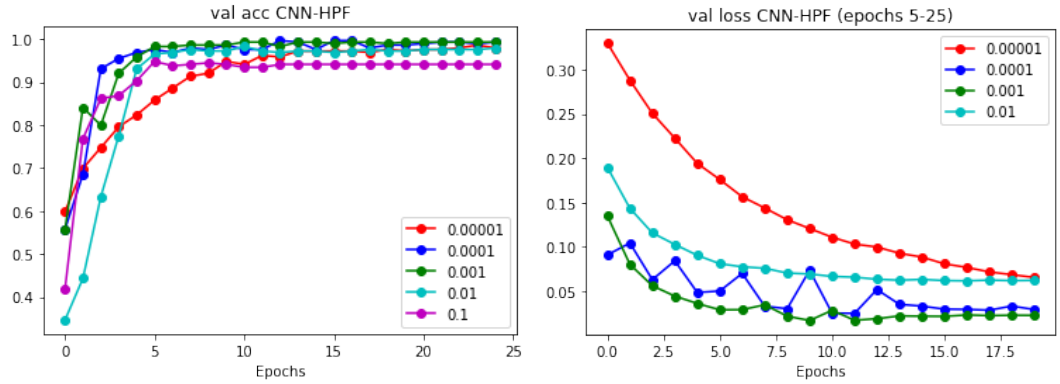
[10] Xin Wang, Hui Guo, Shu Hu, Ming-Ching Chang, and Siwei Lyu. Gan-generated faces detection: A survey and new perspectives (2022), 2022.

[11] Xiao Chen Xin Liu. A survey of gan-generated fake faces detection method based on deep learning. *Journal of Information Hiding and Privacy Protection*, 2(2):87–94, 2020.

A Appendix

A.1 Hyperparameter tuning on CNN-HPF

Learning rate



Batch size