DeepFake Image Detection
A Comparative Study of Three Different Convolutional Neural Networks

May 4, 2020

Abstract

With the advent of Generative Adversarial Network (GAN) and other deep learning based DeepFake techniques, the immediate challenge we face as a community is how to assess the validity of online material be it machine learning derived images or videos. We are faced with an unprecedented potential for an extreme violation of basic human rights along with a fundamental unavoidable change in how humans interact socially. We have already seen evidence of maligning and manipulation of news headlines, medical (dis)information along with abuse of individual privacy. The goal of this proposed project is to use an online image database to effectively detect DeepFake images. This paper focuses on use of convolutional neural networks for classification of true versus fake images obtained from a large online database. We aimed to compare three different convolutional neural networks: 1)VGGFace, 2) DenseNet and 3) Custom CNN Architecture. Future work would include the use of unsupervised clustering methods / auto-encoders explore if true versus fake images cluster separately and also to add transparency and interpretability to our models by use of CNN visualization methods.

1 Introduction

DeepFake, an unheard-of concept to the general masses until 2017 now dominates the social media world. DeepFake utilizes both supervised machine learning (ML) methods such as convolutional neural nets (CNNs) and generative adversarial networks (GANs) along with unsupervised ML methods such as autoencoders to create fabricated images, videos, and audio. Recent additions to the basic DeepFake framework have involved the use of generative adversarial networks (GAN) [1] which are generative models that learn the distribution of the data without any supervision. Given these innovations in the DeepFake images and videos now being synthesized, there is now a serious forensics problem in how we can distinguish between fake images from real ones. This can lead to many ethical quandaries with serious societal impact. This can include the creation of disturbing fake pornography using celebrity data, propagation of fake news along with deception of security systems which depend on face recognition systems. This project addresses the critical need for the application and assessment of existing ML methods along with development of new ones to forensically ascertain the fake versus truth data in the onslaught of DeepFake data. We used the "140k Real and Fake Faces" kaggle data set [2] which consists of 70,000 real faces (from Flickr) and 70,000 fake faces (StyleGAN-generated). We also used the "Real and Fake Face Detection" kaggle data set [3] for additional samples. To implement and solve the problem at hand, We used two existing CNN frameworks (VGGFace and DenseNet) along with a custom CNN with the primary objective of distinguishing between real and fake data.

2 Literature Review

2.1 What is a DeepFake and how is one created?

DeepFake, a combination of "deep learning" and "fake" is a machine learning method that can layer images or videos of a target person to that of a another person to create a new "unreal" image/video of the target person doing or saying things the source person does. The construction of a DeepFake image or video in essence requires auto-encoders, which consists of an encoder and decoder. The process of creating a DeepFake first involves reducing the image to a decreased dimension space (compressed image), which retains critical image information by the encoder. The next step involves the decoder which then reconstructs this image.

2.2 Recent innovations in DeepFake architectures

Recent additions to the basic DeepFake framework have involved the use of generative adversarial networks (GAN) [4] which are generative models that learn the distribution of the data without any supervision. GANs were conceived by Goodfellow et al. and consist of an updated framework for estimating generative

models via an adversarial process, in which two models are trained simultaneously. A generative model G that captures the data distribution, and a discriminative model D that estimates the probability that a sample came from the training data rather than G. The training procedure for G is to maximize the probability of D making a mistake. The application of GANs to DeepFakes involves the decoder consisting of training of G and D in an adversarial relationship [4]. Since the generator creates new images from the latent representation of the source material and is constantly corrected as the discriminator attempts to determine whether or not the image is generated. This results in a decoder which is immensely efficient in retaining key image information in the latent space. This results in a generator which creates images that are extremely close to real data along with a process in which any defects would be caught by the discriminator. Initially, when GANs were applied to to DeepFake models, initially these had an architecture which included a single GAN. Since then, there have been several innovations with the goal to improve the quality of the fake data. These include the use of Cycle-GANs to address the issue of the need for paired training images during the training period by many training models [5]. However, While GAN images improve in their realism over time, DeepFake methods were still challenged by the lack of control in their output, i.e. changing specific features such pose, face shape and hair style in an image of a face. This has been addressed by NVIDIA by a Style-Based Generator Architecture for GANs (StyleGAN) method [6]. StyleGANs generate the fake image gradually, starting from a very low resolution and continuing to a high resolution (1024×1024). By modifying the input of each level separately, it controls the visual features that are expressed in that level, from coarse features (pose, face shape) to fine details (hair color), without affecting other levels. The ”140k Real and Fake Faces” Kaggle data set used in this project consists of 70,000 fake faces which have been StyleGAN-generated.

2.3 Moral and Ethical quandaries created by the DeepFake era

The negative impacts of DeepFake have been widespread and have led to various scandals. The increase in use of DeepFake can be attributed to it ease of accessibility with the prominence of applications that promote the technique such as FakeApp [7]. This is app which was released on Reddit and is essentially a guided user interface wrapped around the Deepfake algorithm, allowing users with limited knowledge of programming and machine learning to create Deepfakes. Several other versions have followed such as OpenFaceSwap [8]. Such applications are a serious threat as they allow the average person without an extensive background or understanding of deep learning algorithms to create images or videos that can have harmful implications on ones reputation and privacy. A vast majority of the reported targets of DeepFake are celebrity figures and politicians. One such example is a viral DeepFake video of the creator of Facebook, Mark Zuckerberg publicly making a statement that he has “total control of billions of people’s stolen data.” DeepFakes such as FakeApp, OpenFaceSwap and MyFakeApp have also been used to create fake celebrity pornographic videos, revenge porn, fake news and malicious hoaxes. Such viral videos and pornographic images have caused quite the uproar from the public, and so has required the government to take stern action, with the House of Senate holding a special hearing in June 2019 on DeepFake to discuss its implication on national security and public misinformation.

2.4 Published approaches used for forensic evaluation of DeepFakes

Initial attempts at creating DeepFake detection systems focused on the weaknesses in DeepFake generation methods then which included models which had not been trained on footage of people with their eyes closed. This meant that their generated videos featured unnatural blinking patterns [9]. However, this detection method was soon overcome by the next generation of DeepFake models which the included blinking in their training data.

Detection methods have also exploited the fact that current DeepFake algorithm could only generate images of limited resolutions, which need to be further warped to match the original faces in the source video. Such transforms leave distinctive artifacts in the resulting DeepFake videos. Li et al. showed that these artifacts can be effectively captured by CNNs and subsequently used as a method of distinguishing between real and fake data [10]

Another area of focus has been the use of head poses as a way of detecting inconsistencies in DeepFake images [11]. Yang et al. developed a system based on the observations that DeepFakes are created by splicing synthesized face region into the original image, and in doing so, introduce errors that can be revealed when 3D head poses are estimated from the face images. The authors report an SVM classifier which was evaluated using a set of real face images and Deep Fakes after features were selected which cue specific features of head poses.

DeepFake detection techniques have also focused on a specific image domain such as that of world leaders [12]. This consisted of a forensic technique that models facial expressions and movements that typify an individual’s speaking pattern. This algorithm tracked small facial movements unique to each individual, markers which are known as “soft biometrics” such how a person smiles, purses their lips or raises their eyebrows. This paper reported 92% accurate at classifying several different types of DeepFakes.

2.5 Rationale behind our forensic approach

Given the need for forensic evaluation due the challenges posed by increasingly real DeepFake images, we used two existing CNN frameworks (VGGFace and DenseNet) along with a custom CNN with the primary objective of distinguishing between real and fake data. Outlined below are details of the three methods used.

DenseNet, an extension of a widely known Residual CNN architecture (ResNet) and has shown considerable promise in the field of facial recognition. Unlike its ResNet counterpart and other convolution neural networks, DenseNet differentiates itself by establishing a direct connection between each layer to all of the subsequent layers in the network through the use of dense blocks [13]. While doing this, DenseNet utilizes a transition layer made of convolution, average pooling, and batch normalization between each dense block to concatenate feature maps. By concatenating feature maps, there is collective knowledge shared by all the layers, leading to better flow of information as the learned features are non-redundant and each layer has access to gradients from the initial input and loss function. This allows for deep supervision to occur, making DenseNet a powerful network. Additionally, using DenseNet poses additional benefits such as it requires fewer parameter, reduces number of feature maps needed, and is not computationally expensive compared to other CNNs. To test the capabilities of DenseNet, we compared its performance with another neural network architecture known as VGGFace.

VGGFace is an image recognition model which has achieved state-of-the-art results on standard face recognition data sets developed by researchers at the Visual Geometry Group at Oxford [14]. This model allows us to develop a very large training data set whilst requiring only a limited amount of person-power for annotation. VGGFace uses a method for collecting face data using knowledge sources available on the web and deploys this procedure to build a data set with over two million faces. This data set is then used as the basis for developing deep CNNs for face recognition tasks such as face identification and verification. Specifically, models are trained on the very large data set, then evaluated on benchmark face recognition data sets, demonstrating that the model is effective at generating generalized features from faces. This includes a process of training a face classifier first that uses a sigmoid activation function in the output layer to classify faces as people. This layer is then removed so that the output of the network is a vector feature representation of the face, called a face embedding. The model is then further trained, via fine-tuning, in order that the Euclidean distance between vectors generated for the same identity are made smaller and the vectors generated for different identities is made larger. This is achieved using a triplet loss function which aims at learning score vectors that perform well in the final application, i.e. identity verification by comparing face descriptors in Euclidean space. A deep convolutional neural network architecture is used in the VGGFace style, with blocks of convolutional layers with small kernels and ReLU activations followed by maxpooling layers, and the use of fully connected layers in the classifier end of the network.

The Custom CNN was used primarily to evaluate the effectiveness of the two popularly known frameworks described above. The results obtained from this model will help to decide if the other two models are as good as they claim. Additionally, this model will incorporate techniques such as dropout and padding, not found in the other models. We can investigate if such techniques improve the performance of a CNN.

3 Data

To conduct our analysis on deep fake image detection, we obtained a data set from kaggle [3]. This data set comprises of 960 expert-generated photo-shop images of the human face. These high-quality images contain a different face in each image and can have manipulations to the whole face or areas of the face such as the eyes, nose, or mouth. For training and comparison purposes, along with the fake images, the data set also includes 1081 real images with no manipulations made. Initially, we thought this data set would be sufficient for solving our problem. However, with only around 2,000 images to work with, it was not feasible to only use this data set.

Consequently, we obtained another data set from kaggle [2]. This data set consisted of 140,000 images of the human face split into train, validation and test set files. There were 100,000 training images, 20,000 validation images, and 20,000 testing images, respectively. In each set, there was an equal ratio of real images to fake images. A label of 1 was assigned to real images and a label of 0 was assigned to fake images. The real images in this data set were collected from an image and video hosting service, Flickr. On the other hand, the fake images were produced using Generative Adversarial Networks (GANs). To incorporate data from the previous data set, we added the 2,000 images from that data set to the test set. This gave us approximately 22,000 images to use in the test set. From inspecting the data, it should be noted, that the samples for training are diverse as there are images of people of different races, ages, and genders.

4 Methods/Analysis

For this comparative study on the uses of CNN networks for DeepFake image detection, we trained 5 models. Three of the models were trained using DenseNet, one model was trained using VGGFace, and one model was trained using a custom CNN. In the subsequent sections, we discuss each of the models in detail.

4.1 Model 1: Custom Model

Based on theory reviewed in our SYS 6016 coursework, for the custom model, we used a total of 6 convolutional layers each paired with batch normalization layer and maxpooling layer. The activation function used for all the convolutional layers was Rectified Linear Unit (ReLU). We also applied dropout for each convolutional layer to minimize over fitting. In addition to applying a dropout, we also used padding in order to allow the kernel to have more space to inspect the image, leading to higher accuracy. After the convolutional layers, we also added a dense layer at the end with sigmoid activation as this was a binary classification problem.

4.2 Model 2: Custom Model with Augmented Data

For this model, we decided to run the same framework as Model 1 but integrated data augmentation in an effort to observe changes in prediction accuracy. We added a horizontal flip to the images and also included a zoom range of 0.2, which allows for the image to be zoomed in. Apart from this, we also specified a shear range of 0.2. It should also necessary to point out that we used a rescaling factor to ensure that all number is the RGB matrix are within the range [0,1]. This safeguards against image quality being a factor in causing a difference in classification between images as not all images have the same pixels. These data augmentation parameters were used to help with the generalizability of our findings.

4.3 Model 3: VGGFace

For the model we implemented the VGGFace architecture proposed by Nhu et al. in [14]. This model consisted of five layer blocks with each block made up of convolutional and maxpooling layers. The first and second block each comprised of two 3 x 3 convolutional layers followed by a maxpooling layer. The third, fourth, and fifth blocks each comprise of three 3 x 3 convolutional layers followed by a maxpooling layer. All convolutional layers used ReLU activation function. Since, VGGFace uses pre-trained weights, we had to fine tune to the needs of our . Fine-tuning was done by adding a dense layers after the five layer blocks that gave us the facial features. Finally, we also added a dense layer as the output layer with sigmoid activation.

4.4 Model 4: DenseNet

For this model, we wanted to observe the accuracy of the DenseNet-121 model found in Keras with a slight modification with the use of a dense layer as the last layer. This model contained 4 dense block with closely connected layers such as batch normalization (BN), ReLU activation, and 3 x 3 convolution. In addition to this, between each dense block, the model also contained a transition layer which included a 2 x 2 average pooling layer and a 1 x 1 convolution. After the last dense block, we added the custom dense layer with sigmoid activation. For training we used 100,000 images in the training set and 20,000 images in the validation set.

4.5 Model 5: DenseNet with Data Augmentation

To add to the pre-built DenseNet model, we decided to experiment with data augmentation as this would allow us to diversify the data being used to train the model. We used all the same layers as Model 4. For this specific model, we investigated the impacts of rotating images. Specifically, we used a horizontal flip as well as a rotation range of 20. Furthermore, we also included the rescaling parameter in this model like we did in the custom model with data augmentation.

4.6 Model 6: DenseNet with Gray Scale

The main premise of this model was to gauge how much importance color has on the classification of fake versus real image. Color images and gray scale images differ in pixels and resolution, hence this can impact the classification of fake images. Therefore, we decided to investigate this. For this model, we used the pre-built DenseNet but changed all the images to gray scale using the color mode parameter for the image data found in flow_from_directory. The layers used stayed the same in accordance with the previous models.

4.7 Analysis of Vector Representations

For all the models we extracted the output of the last layer before classification to see if the vectors are representative of the images. The vectors have dimensions of 512, 2048, 1024 for custom, VGGFace and DenseNet models. This was entirely too large, therefore, we used principal component analysis (PCA) to keep points that contributed the most in terms of variability. By running PCA, we were able to retain 50

principal components. We then used support vector machine (SVM) with polynomial kernel to classify the retained components into the two classes (real or fake).

5 Results

Through this comparative study, we learned that neural networks are very efficient in detecting and classifying GAN generated images.

Table 1: Results

Model	Neural Network			SVM after PCA		
Metrics	Accuracy	Precision	Recall	Accuracy	Precision	Recall
Custom Model	0.931	0.89	0.89	0.974	0.972	0.976
Custom Model with Augmented Data	0.907	0.84	0.84	0.91	0.906	0.916
VGGFace	**0.96**	**0.95**	**0.95**	**0.995**	**0.995**	**0.995**
DenseNet	**0.956**	**0.94**	**0.93**	**0.984**	**0.987**	**0.981**
DenseNet with Augmented Data	0.871	0.79	0.73	0.863	0.861	0.864
DenseNet with Gray Scale Images	0.959	0.95	0.94	0.503	0.50	0.516

The VGGFace architecture proposed by Nhu et al. performed the best with an accuracy of 96%. However, this model is computationally very expensive, and requires much sophisticated processors to enable us to train on augmented data. The DenseNet architecture provided similar results as can be seen in Table. 1, but is much more efficient and concurs with the claims made in [13]. Introducing data augmentation reduced the accuracy of the DenseNet model by around 8-10%, which is expected as data augmentation makes the training samples harder. Training the model for larger number of epochs can probably provide even better results. The performance of the DenseNet architecture on gray scale images shows that color does not have any impact on the model's ability to classify GAN generated images.

The Custom Model also gave decent results. Its performance in terms of computational efficiency lied in between VGGFace and DenseNet. The Custom Model provided better results on augmented data when compared to DenseNet. The relative performance of all models can be observed via the ROC plot shown in Fig. 1

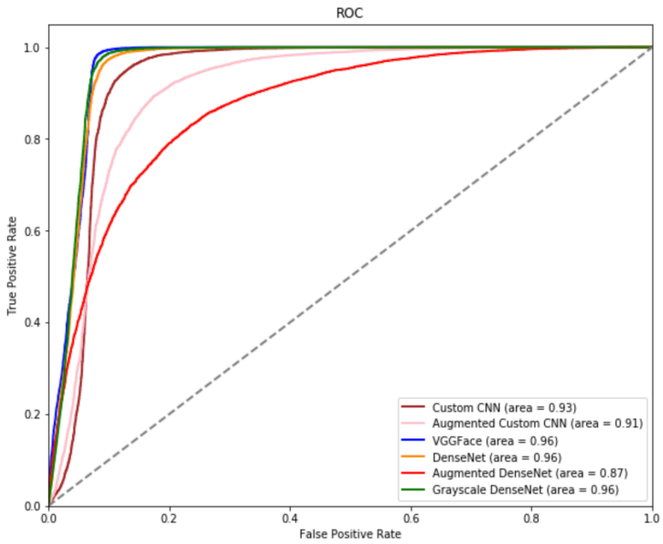

Figure 1: ROC Curve

The scatter plot of the principle components for each model except Model 6 (DenseNet with Gray Scale) labelled based on the image type show that the vectors are representative of type of the image. PC0 and PC1 components of VGGFace in particular, form very distinctive clusters for real and fake images. From the scatter plots we can observe that the DenseNet with Gray Scale does not give accurate representations for the images and therefore, classification using these vectors in SVM shows very poor performance as we can see in Table. 1.

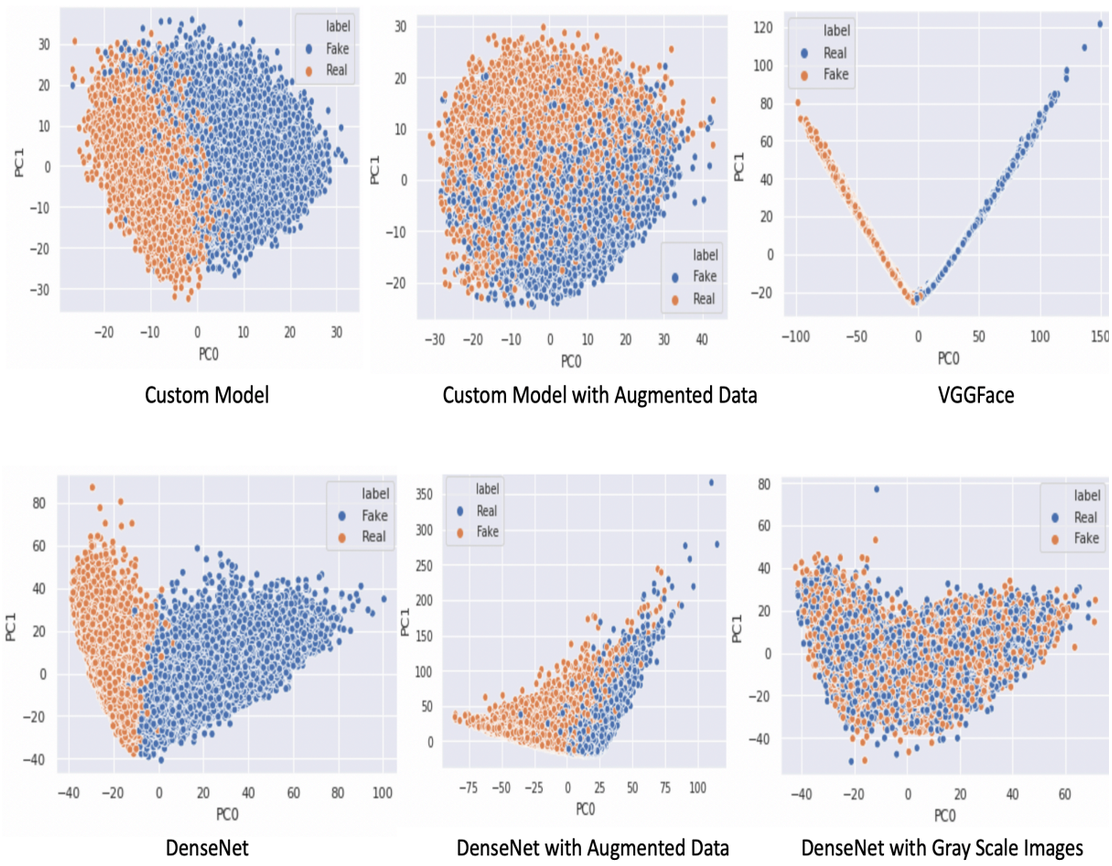

Figure 2: Principle Component Plots

6 Conclusion

DeepFakes are here to stay and in doing so have changed our perception of reality forever. An immense challenge in developing forensic methods to detect real versus fake images and videos is that once papers are published on new innovative approaches or methods are shared via open access, these flaws are immediately incorporated in the next iteration of DeepFake generation methods. Even with models with accuracy as high as 97% are not enough. Similar to the medical domain, it is the ones that are missed that represent the larger problem - i.e., 3% of billions of images on Google or Facebook platforms would represent an immense loss of trust from users of these interfaces.

Our results show that state-of-art CNNs are now able to distinguish with minimal mis-classification inaccuracies between fake and real data. However, the detection of these minimal inaccuracies remain a critical area of research. Recent efforts have focused on improving the algorithms that create DeepFakes by adding specially designed noise to digital photographs or videos that are not visible to human eyes and can fool the face detection algorithms [15]. Ultimately, this is a battle now between human ingenuity and the ubiquitous pervasive presence of machines which have qualities which allow them to become iteratively intelligent. Future work would include the use of unsupervised clustering methods such as auto-encoders to explore if true versus fake images cluster separately and also to add transparency and interpretability to our models by use of CNN visualization methods.

References

[1] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In *Advances in neural information processing systems*, pages 2672–2680, 2014.

[2] https://www.kaggle.com/xhlulu/140k-real-and-fake-faces, .

[3] https://www.kaggle.com/ciplab/real-and-fake-face-detection, .

[4] T. T. Nguyen, C. M. Nguyen, D. T. Nguyen, D. T. Nguyen, and S. Nahavandi. Deep learning for deepfakes creation and detection. *arXiv:1909.11573*, 2019.

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

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

[7] https://www.reddit.com/r/fakeappreddit/.

[8] https://www.reddit.com/r/sfwdeepfakes/comments/8a2sj1/openfaceswap_an_actual_deepfakes_gui/.

[9] Yuezun Li, Ming-Ching Chang, and Siwei Lyu. In ictu oculi: Exposing ai generated fake face videos by detecting eye blinking, 2018.

[10] Yuezun Li and Siwei Lyu. Exposing deepfake videos by detecting face warping artifacts, 2018.

[11] Xin Yang, Yuezun Li, and Siwei Lyu. Exposing deep fakes using inconsistent head poses, 2018.

[12] Shruti Agarwal, Hany Farid, Yuming Gu, Mingming He, Koki Nagano, and Hao Li. Protecting World Leaders Against Deep Fakes. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*. IEEE, June 2019.

[13] Gao Huang, Zhuang Liu, Laurens van der Maaten, and Kilian Q. Weinberger. Densely connected convolutional networks, 2016.

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

[15] Yuezun Li, Xin Yang, Baoyuan Wu, and Siwei Lyu. Hiding faces in plain sight: Disrupting ai face synthesis with adversarial perturbations, 2019.