A Survey on Deepfake Creation and Detection Using Deep Learning.

1. Abstract

Deepfake uses **Generative Adversarial Networks (GAN)**, a deep learning method to create high-quality fake images and videos, this is possible today due to advances of hardware and the availability of data. Contents created by Deepfake are hard to distinguish with real ones; therefore, it is a major problem in today's era of fake news. In this paper, we survey datasets used for detecting Deepfake contents. In addition to that, we will also perform Deepfake fake face images detection by using our custom CNN model, **MobileNet** model and pre-trained **MobileNet** model on **ImageNet** using **Transfer Learning**. We will try to find the answer whether a **MobileNet** model would produce enough accuracy to classify images generated from **GAN**.

2. Introduction

Deep learning is used in different forms to solve real-world problems. It is used to detect pedestrians for self-driven automatic vehicles, it is also being used to create a robot who could do day to day activities. Deepfake algorithms are used to create fake images or videos by clipping target subject data to source subject data so that the target subject would appear as an impersonation (speaking, facial expression etc.) of the source subject. It has become a serious problem for society because misinformation can be conveyed to masses by using these fake contents. Initially, Deepfake algorithms were used to generate fake videos for celebrity and by imposing their videos on pornographic videos but these days it is being used to create fake videos such as a **Synthesizing Obama** (Suwajanakorn, Seitz, & Kemelmacher-Shlizerman, 2017). Therefore, it is required to have solutions to detect fake contents so that countermeasures could be taken to stop defaming a person and circulation of fake news which may have severe consequences. With the advancement of deep learning technologies and abundance availability of data, it has accelerated the generation of Deepfake contents. The last decade has generated an enormous amount of data in different formats such as text, voice, images and videos from several social networking websites and most of the data is public and is used in academics for the purpose of research and analysis. We have also seen parallel exponential growth in hardware development and deep learning algorithms which have created a favourable ecosystem for the analysis of data in different forms such as classification of voices, images and videos; detection of objects in an image or video; creation of fake images from existing images by morphing different attributes of an image.

We have seen similar technologies in apps such as **Snapchat**; similarly, videos can be altered using tools like **Adobe Photoshop** or apps like **FaceSwap** (Marek, 2016/2020) and **ZAO** (Chinese Deepfake App ZAO Goes Viral, Privacy Of Millions). We reviewed several peer-reviewed papers on the course of data collection, how that data has been used for fake and real face classification, how images with fake face or real face could be created and detected using different flavours of CNNs on using different feature vectors with choosing some of traits associated with an image.

Dang *et al*. (Dang, Liu, Stehouwer, Liu, & Jain, 2020) suggested that there are four ways to create Deepfake contents (1) entire non-exist new face creation, (2) facial expression like smile or frown can be manipulated, (3) facial attributes such as gender, hair and glasses can be manipulated, (4) a target face swapped with source face, identity swapping. **Generative Adversarial Network (GAN)** (Goodfellow et al., 2014) uses the iterative process to generate new content from source content that results in realistic content. It has two

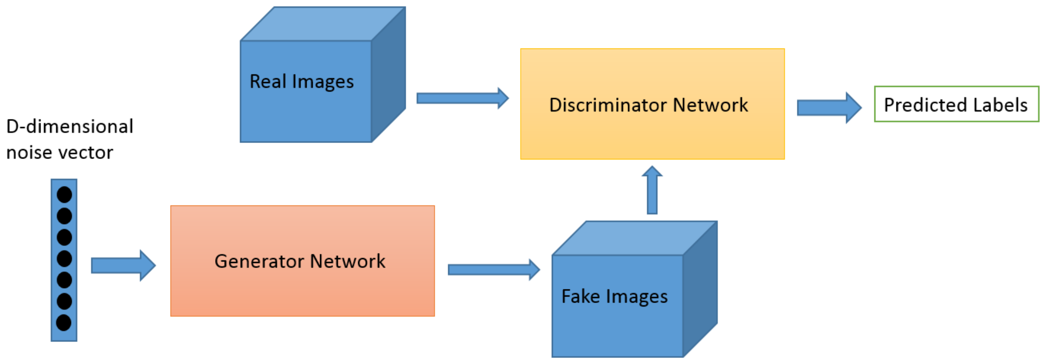


Fig 1. GAN architecture.

deep neuron networks, Generative and Discriminator, they compete with each other during the training of respective models and performs unsurprised learning.

In this paper, we have presented a detailed and systematic review of datasets that were used in state-of-the-art technologies for the creation and detection of Deepfake images and videos. We have also collected datasets to perform real face and fake face images classification and analysis by using own custom CNN model, **MobileNet** model and pre-trained **MobileNet** model on **ImageNet (Krizhevsky, Sutskever, & Hinton, 2012)** dataset and presented comparison results. In this paper, we tried to find answers to these research questions:

1) Which is a more suitable dataset for the creation of Deepfake images?
2) How much more efficient is our custom shallow CNN model?
3) **MobileNet** model, which is either pre-trained or trained from scratch, would produce enough accuracy to classify images generated from **GAN**?

3. Literature Review

Dong *et al.* (Yi, Lei, Liao, & Li, 2014) collected data such as name and face image from IMDB website by using web crawling, they named this database as **CASIA-WebFace** and it has 500,000 images for 10,000 people. They selected only those celebrities whose birth date lies between the range 1940-2014, after that, they cleaned images by using multi-factor face detector. Then they removed those people whose image counts is less than 15. Next, they annotated face in each of the images after locating the face in it. Then, they cleaned the database. They used 11-layers CNN model for the classification of face images into identiy. This network includes 10 layers of CNN, 5 pooling layers and 1 fully connected layer. The dimensions for the input layer is 100x100x1, first, four pooling layers uses max operator while the fifth layer uses avg operator. They have used filter of size 3x3 for convolution and ReLU neurons for all CNN layers. They performed classification with an accuracy of 92.24%.

Cao *et al.*(Cao, Shen, Xie, Parkhi, & Zisserman, 2018) collected 3.31 million images by web crawling through Google Image Seach. It has 9,131 entities with each entity has on an average 326.6 images. They named this database as **VGGFace2**. It has faces from different ethnicities which includes faces from Indian and Chinese ethnicity as well. It is evenly distributed across age, gender, background, pose and lighting. It has faces of celebrities from many professions such as politicians and sports. It has bounding boxes around faces those were verified by a human. It is also a gender-balanced database with 40.7% females. They used 8,631 classes for training and 500 for testing. They further used **ResNet-50** and **SENet** CNN models for the evaluation of the database. They used 3 variants of traning for the comparisons of results (i) **VGGFace** used for network learning, (ii) **MSIM** used for network learning and (iii) **VFFFace2** for network learning. After performing the experiment, they got Top-1 error (%) results in 10.6, 5.6 and 3.9 for **VGGFace**, **MSIM** and **VGGFace2** respectively.

Liu *et al.* (*Liu et al. - Deep Learning Face Attributes in the Wild.Pdf*) collected images from challenging face dataset **CalebFaces**. It has 10,000 identities with 20 images for each identity, therefore a total of 200,000 images. They named it as **CalebA**. Each image has been annotated with 40 attributes and five important areas of the face by experts. For their experiments they divided their database into three sets, 160,000 images were used for pre-training and fine-tuning of the **ANet** and **Lnet** model. 20,000 images of 1,000 identities were used to train SVM. Similarly, 20,000 images from 1,000 were used for testing. They used two CNN modes, **Lnet** for locating entire face region inside an image and **ANet** to extract features that could be used for identification of face attributes.

Tariq *et al.*(Tariq, Lee, Kim, Shin, & Woo, 2018) used **CalebA** as real face image database for the training of different CNN models to distinguish real and fake images. They used **ProGAN** and **Adobe Photoshop CS6** to generate fake face image database. They used images of various sizes: 64x64, 128x128, 256x256, 1024x1024. They achieved 94% AUC for GAN generated images and 74.9% AUC for human-generated images. They found **Xception** and **DenseNet** could not give good results, therefore, they proposed their own custom **Shallow Convolution Network (ShallowNet)**. They used three different versions of their **ShallowNet** for smaller size images. They used L2 regularizer for Conv2D layer, various values of batch normalization, dropout and Max Pooling. 200,000 images were used for training out of which 20% were used for validation and 18,000 images were used for testing.

Jain *et al.* (Jain, Singh, & Vatsa, 2018) used **CalebA** to generate attributes transferred fake faces using **StarGAN** (Choi et al., 2018), they used attributes such as gender, age, brown hair, blond hair and aging. 18,000 images were processed using **StarGAN** by using 9 attributes on 2000 images, 15,000 images of size 128x128x3 were used from **CalebA** database. They trained a CNN model with 32,000 images; 2,000 images were used for testing and 500 images were used for validation. They achieved an accuracy of 99.83%.

Yu *et al.* (Yu, Davis, & Fritz, 2019) used **CalebA** database for the human face and **LSUN** database for bedroom data, this results in a total of 20,000 images. They synthesized images by using different GAN models (**ProGAN** (Karras, Aila, Laine, & Lehtinen, 2018), **SNGAN** (Miyato, Kataoka, Koyama, & Yoshida, 2018). **CramerGAN** (Bellemare et al., 2017) and **MMDGAN** (Bińkowski, Sutherland, Arbel, & Gretton, 2018)) . They trained all models from starch using factory settings, but they changed epos value to 240 and generated images of size 128x128x3. They proposed that a GAN model generates a unique and immutable fingerprint that could be used as a feature to classify images into a group of their origin model. They proposed a simple attribution CNN model that had achieved an accuracy of 99.43%.

Natraj *et al.* (Nataraj et al., 2019) used **CycleGAN** database (Zhu, Park, Isola, & Efros, 2017) for collecting 36,302 images out of which 18,151 were GAN and 18,151 were non-GAN. **CycleGAN** transfers styles from unpaired images such as summer to winter, horses to zebras. In this database, there are GAN images such as apple to an orange, house to zebra, summer to winter, cityscapes, facades, map2sat etc. They added **StarGAN** generated 19,990 images, for this, they used real face 1,999 images from **CalebA** database and generated 17,991 fake face images from GAN model by using attributes such as black hair, blond hair, gender change. For evaluation, they used 50% of data for the training and 25% for the testing and 25% for the validation. They used multi-layered CNN models to detect tempted images. They obtained an accuracy of 99.49% on **StarGAN** database and 93.42% on **CycleGAN** database.

Dang *et al*. (Dang et al., 2020) used **CalebA** database to create **Diverse Fake Face Dataset** (DFFD). They explained four kind of face manipulation techniques to create DFFD: expression swap i.e. transferring expression from one entity to another, total face synthesized i.e. a completely new face constructed by using GAN models, identity swap i.e. entire face swapped to another body and attributes manipulation i.e. modification of face's attributes like hair, beard or glasses. Real face images for DFFD, they have used **CalebA**, **FFHQ** (Karras, Laine, & Aila) and source frames from **FaceForensics++** (Rossler et al., 2019). In their database has 52.3% females with an age range between 21-50 years with images of both high and low quality. For attributions, they have used **StarGAN** to generate fake face images using 2,000 faces of **CalebA** and 4,000 faces of **FFHQ**.

Faces-HQ (FFHQ): Karras *et al.* collected high quality (1024x 1024) images from Flickr by scrawling it, they have cleaned, aligned and cropped all images. They also removed paintings, photos of photos and statues mannualy that results in 70,000 images. They used it to test against their style based generator **StyleGAN** and overperformed non-style based generator, they achieved FID 4.40 on images generated from **FFHQ** database. They have also generated 100,000 fake imags using **StyleGAN.**

Wang *et al.* (Wang et al., 2020) took a novel approach in which they studied the behaviours of neurons while training data and used it to classify images using **SVM**, it is called as FakeSpotter. They used **CalebA** and **FFHQ** for real face database and for fake face images they used **StyleGAN**. They used 5,000 fake and 5,000 real face images for training. For evaluation and the testing they used 1,000 real and fake images with different identities.

4. Methodology

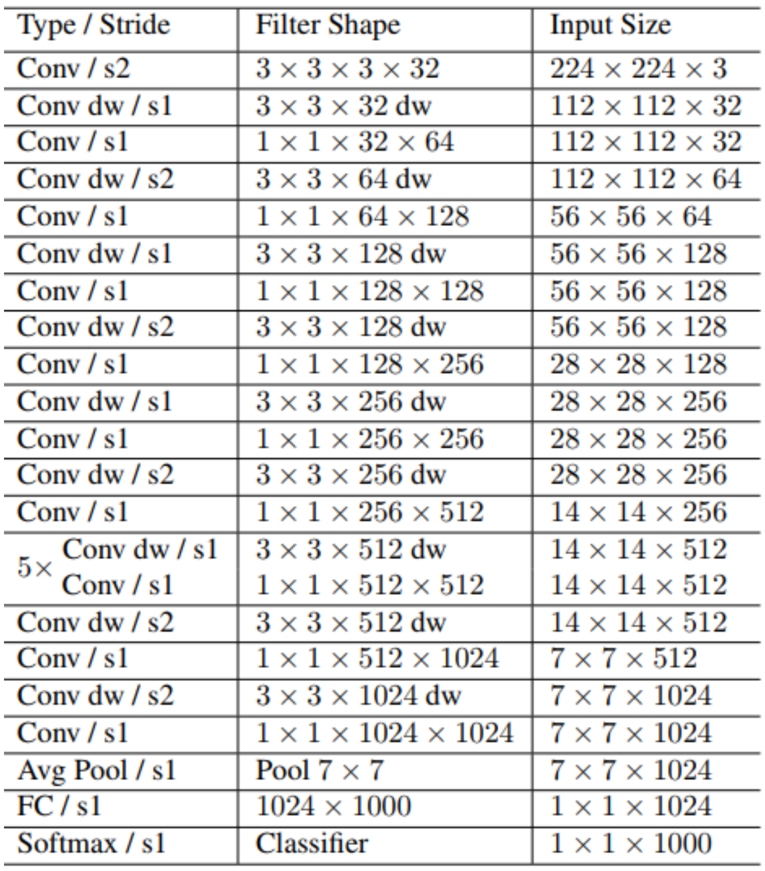
We used three CNNs models to classify images into fake and real, we then evaluated their results and compared their performance based on different matrices We first used a custom shallow convolutional neural network model to train and evaluate. It has 11 layers of which 3 are convolutional layers, 1 batch normalization layer, 1 activation layer, 1 max pool layer, 1 avg pool layer, 2 flatten layers, 1 dropout layer and 1 softmax layer. We used input layer of dimensions 224x224x3 and filter of size 7x7 for the first convolution layer and 3x3 for the second convolution layer for this network, it has **ReLU** (Agarap, 2019) as activation layer and dropout with value 0.8. A summary is presented in table 1 and the network implementation using Keras, please refer appendix 8.1in a file custom.py. We trained our custom model on our collected database then evaluate and compare results with other proposed models.

Layer(type)	**Output Shape**	**Parameters**
conv0 (Conv2D)	(None, 218, 218, 32)	4736
bn0 (BatchNormalization)	(None, 218, 218, 32)	128
activation (Activation)	(None, 218, 218, 32)	0
max_pool (MaxPooling2D)	(None, 109, 109, 32)	0
conv1 (Conv2D)	(None, 107, 107, 64)	18496
activation_1 (Activation)	(None, 107, 107, 64)	0
avg_pool (AveragePooling2D)	(None, 35, 35, 64)	0
flatten (Flatten)	(None, 78400)	0
rl (Dense)	(None, 500)	39200500
dropout (Dropout)	(None, 500)	0
sm (Dense)	(None, 2)	1002

Summary	Total params: 39,224,862 Trainable params: 39,224,798 Non-trainable params: 64

Table 1. Our custom shallow convolution neural network used for classificaiton.

Next, we used **MobileNet** (Howard et al., 2017) as a network to classify fake and real faces images. It has 30 layers with the lightweight network architecture, small, accurate and suitable for mobile and embedded devices, it's layers of the network are shown in table 2 and Keras implementation can be referred from appendix 8.1 in a file MobileNet.py. We trained **MobileNet** on our collected images database then evaluated and compared results with other proposed models. After that, we used **Transfer Learning** to train the second last layer of **MobileNet** network which is already trained on **ImageNet** database keeping all other layers froze on our collected database of images and then evaluated and compared results with other proposed models.



Type / Stride	Filter Shape	Input Size
Conv / s2	$3 \times 3 \times 3 \times 32$	$224 \times 224 \times 3$
Conv dw / s1	$3 \times 3 \times 32$ dw	$112 \times 112 \times 32$
Conv / s1	$1 \times 1 \times 32 \times 64$	$112 \times 112 \times 32$
Conv dw / s2	$3 \times 3 \times 64$ dw	$112 \times 112 \times 64$
Conv / s1	$1 \times 1 \times 64 \times 128$	$56 \times 56 \times 64$
Conv dw / s1	$3 \times 3 \times 128$ dw	$56 \times 56 \times 128$
Conv / s1	$1 \times 1 \times 128 \times 128$	$56 \times 56 \times 128$
Conv dw / s2	$3 \times 3 \times 128$ dw	$56 \times 56 \times 128$
Conv / s1	$1 \times 1 \times 128 \times 256$	$28 \times 28 \times 128$
Conv dw / s1	$3 \times 3 \times 256$ dw	$28 \times 28 \times 256$
Conv / s1	$1 \times 1 \times 256 \times 256$	$28 \times 28 \times 256$
Conv dw / s2	$3 \times 3 \times 256$ dw	$28 \times 28 \times 256$
Conv / s1	$1 \times 1 \times 256 \times 512$	$14 \times 14 \times 256$
5× Conv dw / s1	$3 \times 3 \times 512$ dw	$14 \times 14 \times 512$
Conv / s1	$1 \times 1 \times 512 \times 512$	$14 \times 14 \times 512$
Conv dw / s2	$3 \times 3 \times 512$ dw	$14 \times 14 \times 512$
Conv / s1	$1 \times 1 \times 512 \times 1024$	$7 \times 7 \times 512$
Conv dw / s2	$3 \times 3 \times 1024$ dw	$7 \times 7 \times 1024$
Conv / s1	$1 \times 1 \times 1024 \times 1024$	$7 \times 7 \times 1024$
Avg Pool / s1	Pool 7×7	$7 \times 7 \times 1024$
FC / s1	1024×1000	$1 \times 1 \times 1024$
Softmax / s1	Classifier	$1 \times 1 \times 1000$

Table 2 MobileNet network layers.

5. Experiments

We conducted our experiment on real face images collected from **CalebA** database as shown in fig. 1 and fake face images generated from **StarGAN** model as shown in fig 2.

Fig 1: Real face images were collected from **CalebA** database.

Fig 2: Fake face images were generated from **StyleGAN** model.

5.1. Datasets:

We collected a dataset of images from Kaggle's repository (140k Real and Fake Faces). It has 70,000 real face images from **CalebA** database and 70,000 fake face images generated from **StyleGAN** model, each of the image has a size of 256x256px. This database has a clear demarcation of images into the test, train and valid folders. It has CSVs files corresponding to each type of the folder that contains the metadata of each image those are present in respective folders.

5.2. Evaluation:

We performed several trails of the experiments to evaluate the performance of the proposed models by changing the values of the hyper-parameters. To see how well our proposed model works on differentiating real face images from fake face images, we used Keras library which was developed in python programing language. We used the test database to evaluate the performace of the proposed models. Each model takes an image as input in a form of 3-dimensional vectors and give output results as probabilities of being fake or real. We used **Receiver Operating Characteristic curve (ROC)** to measure the performance of the model. **The Area Under Curve (AUC)** of **ROC** measures the quality of the model, a greater value of AUC signifies a good model. We presented **AUC** values of proposed models into a table 3.

To evaluate our proposed models we took 10,000 real face images form **CalebA** database and 10,000 fake face images generated from **StarGAN model**, so we had total 20,000 images to train the model. Similarly to validate and evaluate the model we took 2,000 images out of which 1,000 were selected from real face images **CalebA** and 1,000 were fake face images generated by **SrarGAN** model. We took epochs as 5, Adam optimizer with learning rate 0.002, categorical_crossentropy as loss function, batch_size as 100.

We first pre-processed images and labels to make it compatible with CNN models, for that, we converted images to size 224x224 from the database images of size 256x256, after that, we converted to 3-dimensional normalised **RGB** representation of an image. We converted labels to 2-dimensional array using hot encoding.

First, we trained our custom shallow CNN model and its training loss, validation loss, training accuracy and validation accuracy curves are shown in fig. 2(a). A **ROC** curve is shown in fig. 2(b). Our custom CNN scored 75% **AUC** as shown in table 3.

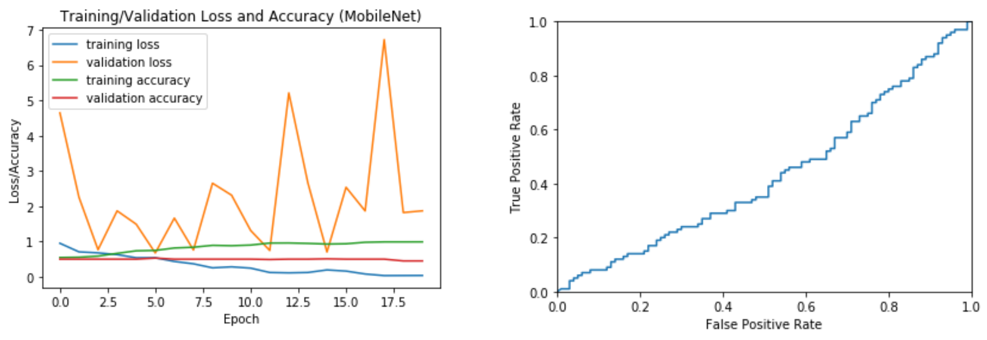


Fig 2. (a) Loss/Accurracy graph for training (b) ROC graph.

Similarly, we trained the **MobileNet** model on our dataset, we trained it from scratch. It took 16 hours to train on the dataset and its training loss, validation loss, trainning accuracy and validaton accuracy curves are shown in fig. 3(a). A **ROC** curve is shown in fig. 3(b). It's **AUC** is only 52%, a little higher than the random value. So, it's not a good detector to detect fake and real images for the Deepfake.

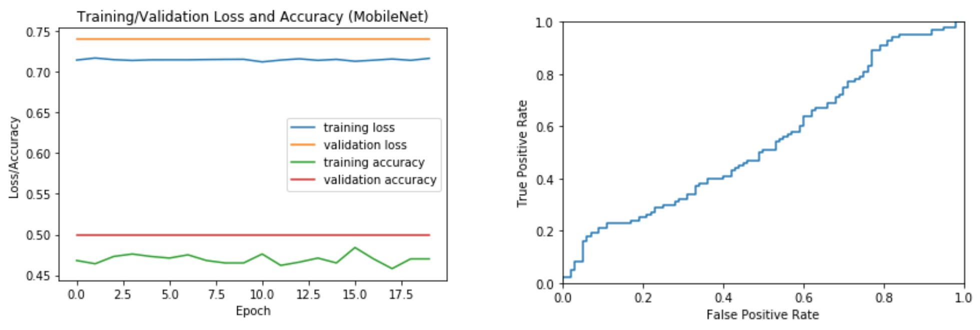


Fig 3. (a) Loss/Accurracy graph for training (b) ROC graph.

To train a **MobileNet** model from scratch is an expensive task in term of time and resources utilization, **MobileNet** took 16 hours to train on our proposed dataset. So it is required to take advantage of **Transfer Learning** to train a given model on a new dataset with existing learning from the prior dataset. We used **MobileNet** weights trained over **ImageNet** dataset, we fixed the first 25 layers of network and trained rest of layers. CNN detects different features of an image on each layer, first few layers detect colours and shapes, next few layers detect textures and objects like ears, eyes and nose and the last layer predicts a whole face wheater is it fake or real. Code can be referred from Appendix 8.1 in a file TransferLearningMobileNet.py.

After we trained a **MobileNet** model from **Transfer Learning**, its training loss, validation loss, training accuracy and validation accuracy curves are shown in fig. 4(a). A **ROC** curve is shown in fig. 4(b). **MobileNet** model took 10 hours to train on our dataset and it's **AUC** is only 56%, a little higher than random value. So, it is not a good detector to detect fake and real images for Deepfake.

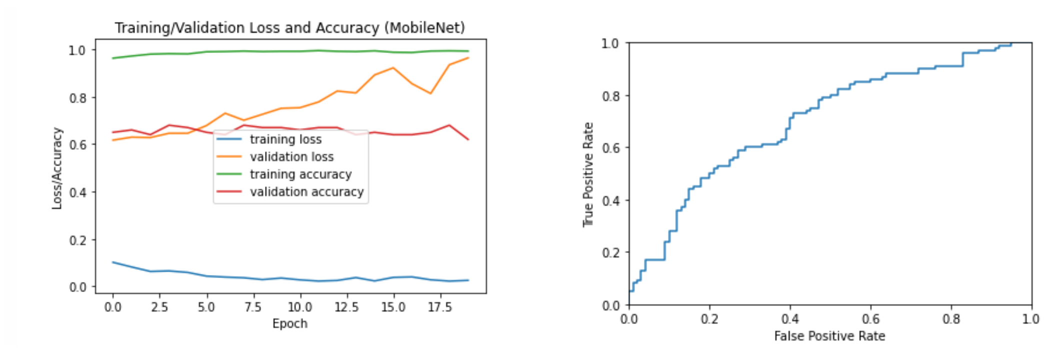


Fig 4. (a) Loss/Accurracy graph for training (b) ROC graph.

Model	Custom CNN	MobileNet	Transfer Learning MobileNet
AUC	74%	52%	56%

Table 3. AUC for different proposed networks

6. Conclusion

In this paper, we proposed three CNN models to distinguish real faces from fake faces synthesized by using **GAN**. For the dataset, we collected it from **CalebA** and fake face images generated by **StarGAN**. We found that **MobileNet** and **MobileNet** after **Transfer**

Learning did not perform well because of overfitting due to deep learning of the model. However, our shallow CNN model works well but its accuracy is not up to the mark. In the future, to improve the performance of the model, we would try to collect features based on GAN and features based on image synthesis such as texture difference, color difference, GAN fingerprint etc. Next, we would assimilate a larger dataset which would include variences in fake and fake face images. Most importantly, we would use GPU based machines to speed up the training process instead of our own laptops and Kaggle.

7. References

140k Real and Fake Faces. (n.d.). Retrieved June 20, 2020, from https://kaggle.com/xhlulu/140k-real-and-fake-faces

Agarap, A. F. (2019). Deep Learning using Rectified Linear Units (ReLU). *ArXiv:1803.08375 [Cs, Stat]*. Retrieved from http://arxiv.org/abs/1803.08375

Bellemare, M. G., Danihelka, I., Dabney, W., Mohamed, S., Lakshminarayanan, B., Hoyer, S., & Munos, R. (2017). The Cramer Distance as a Solution to Biased Wasserstein Gradients. *ArXiv:1705.10743 [Cs, Stat]*. Retrieved from http://arxiv.org/abs/1705.10743

Bińkowski, M., Sutherland, D. J., Arbel, M., & Gretton, A. (2018). Demystifying MMD GANs. *ArXiv:1801.01401 [Cs, Stat]*. Retrieved from http://arxiv.org/abs/1801.01401

Cao, Q., Shen, L., Xie, W., Parkhi, O. M., & Zisserman, A. (2018). VGGFace2: A Dataset for Recognising Faces across Pose and Age. *2018 13th IEEE International Conference on Automatic Face Gesture Recognition (FG 2018)*, 67–74. https://doi.org/10.1109/FG.2018.00020

Chinese Deepfake App ZAO Goes Viral, Privacy Of Millions “At Risk.” (n.d.). Retrieved June 22, 2020, from https://www.forbes.com/sites/zakdoffman/2019/09/02/chinese-best-ever-deepfake-app-zao-sparks-huge-faceapp-like-privacy-storm/#636dc4da8470

Choi, Y., Choi, M., Kim, M., Ha, J.-W., Kim, S., & Choo, J. (2018). StarGAN: Unified Generative Adversarial Networks for Multi-domain Image-to-Image Translation. *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 8789–8797. https://doi.org/10.1109/CVPR.2018.00916

Dang, H., Liu, F., Stehouwer, J., Liu, X., & Jain, A. (2020). On the Detection of Digital Face Manipulation. *ArXiv:1910.01717 [Cs]*. Retrieved from http://arxiv.org/abs/1910.01717

Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., … Bengio, Y. (2014). Generative Adversarial Nets. In Z. Ghahramani, M. Welling, C. Cortes, N. D. Lawrence, & K. Q. Weinberger (Eds.), *Advances in Neural Information Processing Systems 27* (pp. 2672–2680). Retrieved from http://papers.nips.cc/paper/5423-generative-adversarial-nets.pdf

Howard, A. G., Zhu, M., Chen, B., Kalenichenko, D., Wang, W., Weyand, T., … Adam, H. (2017). MobileNets: Efficient Convolutional Neural Networks for Mobile Vision Applications. *ArXiv:1704.04861 [Cs]*. Retrieved from http://arxiv.org/abs/1704.04861

Jain, A., Singh, R., & Vatsa, M. (2018). On Detecting GANs and Retouching based Synthetic Alterations. *2018 IEEE 9th International Conference on Biometrics Theory, Applications and Systems (BTAS)*, 1–7. https://doi.org/10.1109/BTAS.2018.8698545

Karras, T., Aila, T., Laine, S., & Lehtinen, J. (2018). Progressive Growing of GANs for Improved Quality, Stability, and Variation. *ArXiv:1710.10196 [Cs, Stat]*. Retrieved from http://arxiv.org/abs/1710.10196

Karras, T., Laine, S., & Aila, T. (n.d.). *A Style-Based Generator Architecture for Generative Adversarial Networks*. 10.

Krizhevsky, A., Sutskever, I., & Hinton, G. E. (2012). ImageNet Classification with Deep Convolutional Neural Networks. In F. Pereira, C. J. C. Burges, L. Bottou, & K. Q. Weinberger (Eds.), *Advances in Neural Information Processing Systems 25* (pp. 1097–1105). Retrieved from http://papers.nips.cc/paper/4824-imagenet-classification-with-deep-convolutional-neural-networks.pdf

Liu et al. - Deep Learning Face Attributes in the Wild.pdf. (n.d.). Retrieved from http://openaccess.thecvf.com/content_iccv_2015/papers/Liu_Deep_Learning_Face_ICCV_2015_paper.pdf

Marek. (2020). *MarekKowalski/FaceSwap* [Python]. Retrieved from https://github.com/MarekKowalski/FaceSwap (Original work published 2016)

Miyato, T., Kataoka, T., Koyama, M., & Yoshida, Y. (2018). Spectral Normalization for Generative Adversarial Networks. *ArXiv:1802.05957 [Cs, Stat]*. Retrieved from http://arxiv.org/abs/1802.05957

Nataraj, L., Mohammed, T. M., Manjunath, B. S., Chandrasekaran, S., Flenner, A., Bappy, J. H., & Roy-Chowdhury, A. K. (2019). Detecting GAN generated Fake Images using Co-occurrence Matrices. *Electronic Imaging*, *2019*(5), 532-1-532–537. https://doi.org/10.2352/ISSN.2470-1173.2019.5.MWSF-532

Rossler, A., Cozzolino, D., Verdoliva, L., Riess, C., Thies, J., & Niessner, M. (2019). FaceForensics++: Learning to Detect Manipulated Facial Images. *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, 1–11. https://doi.org/10.1109/ICCV.2019.00009

Suwajanakorn, S., Seitz, S. M., & Kemelmacher-Shlizerman, I. (2017). Synthesizing Obama: Learning lip sync from audio. *ACM Transactions on Graphics*, *36*(4), 1–13. https://doi.org/10.1145/3072959.3073640

Tariq, S., Lee, S., Kim, H., Shin, Y., & Woo, S. S. (2018). Detecting Both Machine and Human Created Fake Face Images In the Wild. *Proceedings of the 2nd International Workshop on Multimedia Privacy and Security - MPS '18*, 81–87. https://doi.org/10.1145/3267357.3267367

Wang, R., Juefei-Xu, F., Ma, L., Xie, X., Huang, Y., Wang, J., & Liu, Y. (2020). FakeSpotter: A Simple yet Robust Baseline for Spotting AI-Synthesized Fake Faces. *ArXiv:1909.06122 [Cs]*. Retrieved from http://arxiv.org/abs/1909.06122

Yi, D., Lei, Z., Liao, S., & Li, S. Z. (2014). Learning Face Representation from Scratch. *ArXiv:1411.7923 [Cs]*. Retrieved from http://arxiv.org/abs/1411.7923

Yu, N., Davis, L., & Fritz, M. (2019). Attributing Fake Images to GANs: Learning and Analyzing GAN Fingerprints. *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, 7555–7565. https://doi.org/10.1109/ICCV.2019.00765

Zhu, J.-Y., Park, T., Isola, P., & Efros, A. A. (2017). Unpaired Image-to-Image Translation Using Cycle-Consistent Adversarial Networks. *2017 IEEE International Conference on Computer Vision (ICCV)*, 2242–2251. https://doi.org/10.1109/ICCV.2017.244

8. Appendix

8.1. https://github.com/mukeshyadav-cdac/Custom-Convolution-Neural-Network-to-Detect-Fake-and-Real-faces-from-CalebA-and-StarGAN-Database .