**Detection Of Face Image Manipulation Techniques Using Transfer Learning With Convolutional Neural Networks**

**Nureni Ayofe Azeez1, Adjekpiyede Oghenerunor Obarol2**

**1Department of Computer Sciences, University of Lagos, Nigeria.**

**2Department of Computer Sciences, University of Lagos, Nigeria.**

nurayhn1@gmail.com1, kampkelly@live.com2

**Nureni Ayofe Azeez**

Nureni A. Azeez obtained his B.Tech. (Hons.) from the Federal University of Technology, Akure, Nigeria in 2005, MSc from the University of Ibadan, Oyo State, Nigeria in 2008, and Ph.D. from University of the Western Cape, South Africa in 2013, all in Computer Science. His areas of research include Security & Privacy, Access Control, Grid and Cloud Computing, Sensor Networks, E-Health, and ICT4D. He is a recipient of The Young Scientist Award at the 22nd International CODATA Conference that was held in Cape Town, South Africa in October 2010. He is currently a Senior Lecturer in the Department of Computer Sciences, University of Lagos, Nigeria.

**Adjekpiyede Oghenerunor Obaro**

Adjekpiyede Oghenerunor Obaro will be completing his Masters In Information Technology (MSc) degree programme in Computer Science from the University of Lagos, Nigeria in January 2022. He obtained his BSc (Hons.) degree in Microbiology from the University of Benin, Nigeria in 2015. His research interests include Machine Learning/Deep Learning and Blockchain.

**ABSTRACT**

Recent advances in machine learning and deep learning have brought great improvement in our ability to solve complex problems from computer-vision, speech recognition to self-driving cars in our advancement towards true AI. However, these advances in deep learning are also being used to create software that can endanger, cause threats to people and our society. Face image manipulation techniques can create fake images which are very much impossible to detect with the human eyes. Therefore, there is a need to be able to detect these fake images automatically.

This research presents how to detect fake images by making use of transfer learning with a generalized pre-trained convnet to detect fake images. Some of the most popular fake image manipulation techniques and attempts are looked at in this research to detect and classify them into their correct classes.

**Keywords:** computer-vision, convnet, deep learning, machine learning

**1.0 Introduction**

In modern times, there is a lot of big data available and a large proportion of this big data comes from our social networks such as Facebook, Instagram, Twitter, etc. A large proportion of people these days get their information and news from social media. One major characteristic of social networks these days is the massive reliance on pictures to improve the effectiveness and connectivity of these social networks. Likewise, humans take more pictures now than ever before and upload them online. Also, certain means of authentication of devices and systems now rely on the use of pictures though there are more advanced authentication mechanisms that require a live capture of the individual e.g Face ID.

This rise in social media has promoted a rise in the proliferation of maliciously falsified information. Due to the large amount of images being uploaded online, there has been a rise in the creation and manipulation of some of these images through the use of artificial intelligence. This results in the generation of deep fakes that can make a person appear to have done something they didn’t and can be used by the perpetrators to cover up their nefarious activities. Such altered pictures and videos can be used to blackmail, shame or even ruin one’s reputation.

The underlying mechanism for deepfake creation is deep learning models such as autoencoders and Generative Adversarial Networks (GANs) introduced in 2014 by Goodfellow et al., which have been applied widely in the computer vision domain. These models are used to examine facial expressions and movements of a person and synthesize facial images of another person making analogous expressions and movements. Usually, deepfake methods require a large number of images of an individual to train models to create deepfakes and because celebrities and politicians have a large amount of pictures available on the internet, they are usually frequent targets of these attacks. These can be particularly troubling for security reasons as world leaders can be “impersonated” and made to appear to say or do things that they did not do. Deepfakes therefore can be a threat affecting not only public ﬁgures but also ordinary people.

**2.0 LITERATURE REVIEW**

The first known attempt at trying to swap someone’s face, circa 1865, can be found in one of the iconic portraits of U.S. President Abraham Lincoln. The lithography, as seen in Figure 2.1, mixes Lincoln’s head with the body of Southern politician John Calhoun.



Figure 1: Deepfake Video Detection Using Recurrent Neural Networks

In this work, 4 different kinds of face manipulated images are experimented on which are:

**Deepfakes**

The term Deepfakes has widely become a synonym for face replacement based on deep learning, but it is also the name of a specific manipulation method that was spread via online forums (Andreas et al, 2019). There are various public implementations of DeepFakes available, most notably FakeApp (Fakeapp, 2018) and the faceswap github (Deepfakes, 2018). The FakeApp tool, for example, learns the facial structures of both the source and target faces to create high-quality face swaps. The FakeApp has been taken down but remains available through online forums (Saniat et al, 2019).

**Face2Face**

Face2Face is an approach for real-time facial reenactment of a monocular target video sequence (e.g., Youtube video) (Justus et al, 2018). It transfers the expressions of a source video to a target video while maintaining the identity of the target person. (Andreas et al, 2019). The goal is to animate the facial expressions of the target video by a source actor and re-render the manipulated output video in a photo-realistic fashion. (Justus et al, 2018).



Figure 2: Face2Face: live video editing of facial expressions (Visage Technologies, 2016)

**FaceSwap**

FaceSwap is a graphics-based approach to transfer the face region from a source video to a target video (Jessica and Jan, 2012). Simple techniques, such as the approach by Kowalski (Marek, 2019), use traditional computer graphics techniques.

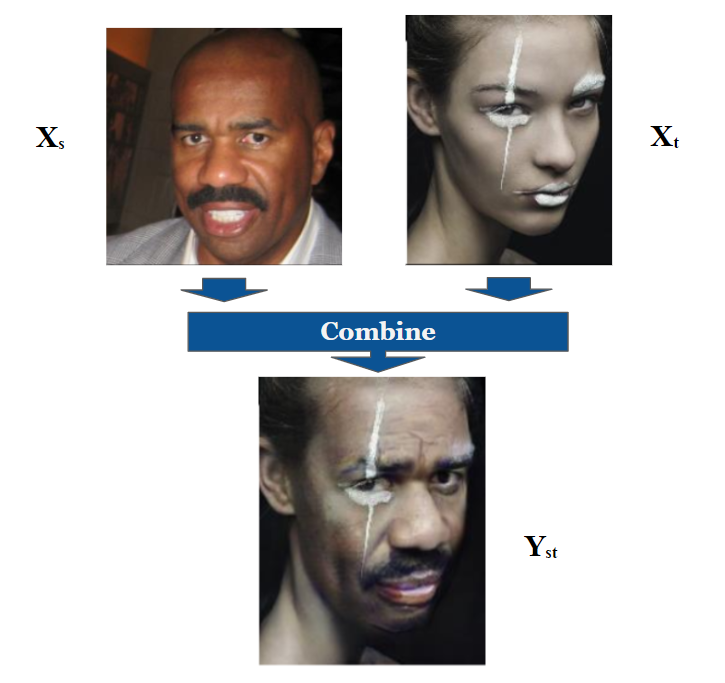


Figure 3: Understanding FaceShifter: a new face-swapping model (Ahmed Maher, 2020)

**Neural Style transfer:**

This is an image manipulation technique introduced by Leon Gatys et al in the summer of 2015. It consists of applying the style of a reference image to a target image while conserving the content of the target image (FranÇois Chollet, 2018)



Figure 4: Neural Style Transfer (FranÇois Chollet, 2018)

In this context, style essentially means textures, colors, and visual patterns in the image, at various spatial scales; and the content is the higher-level macrostructure of the image.

The key notion behind implementing style transfer is the same idea that’s central to all deep-learning algorithms: you define a loss function to specify what you want to achieve, and you minimize this loss. You know what you want to achieve: conserving the content of the original image while adopting the style of the reference image. (FranÇois Chollet, 2018)

Some examples of neural style transfer are:

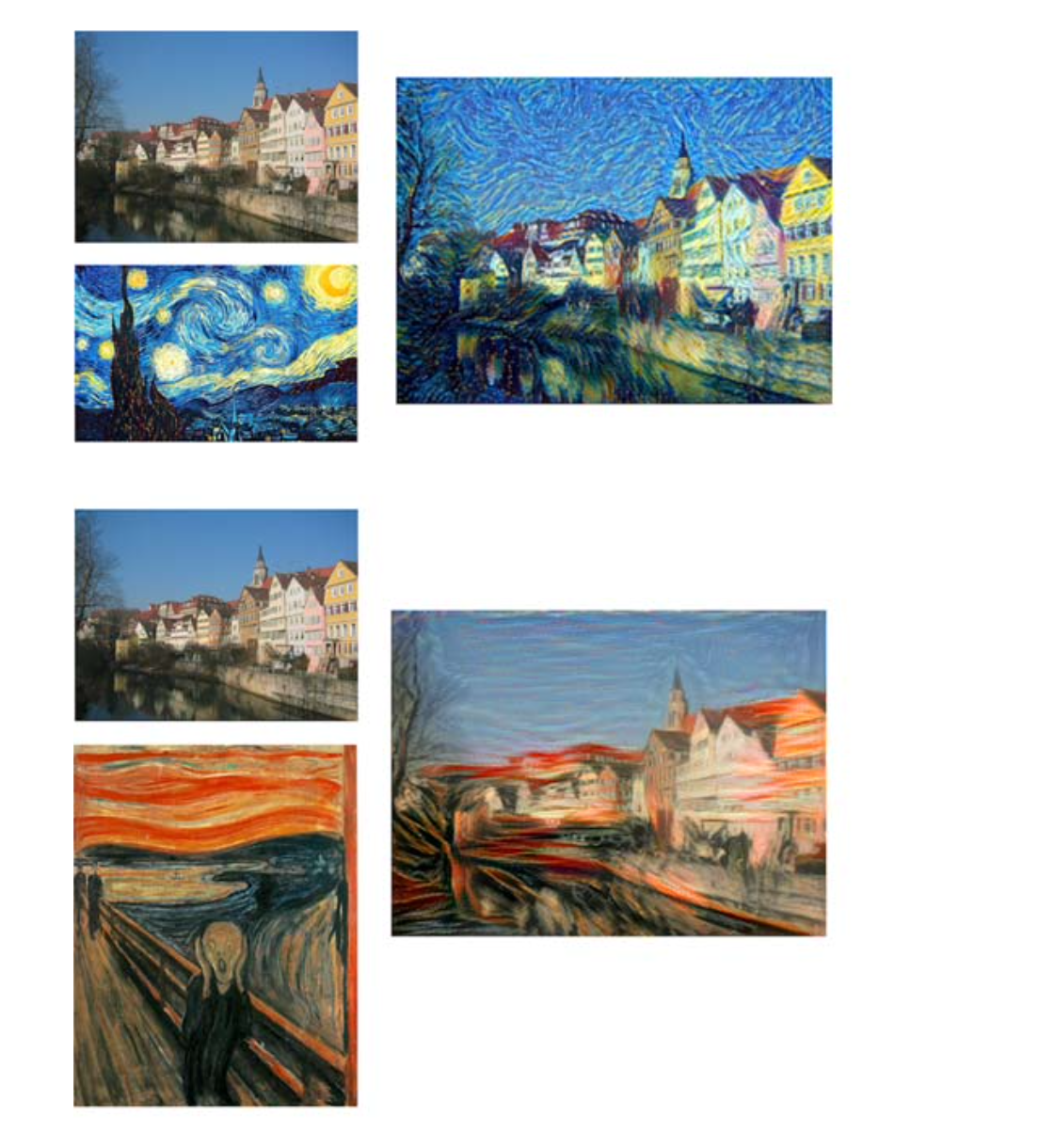


Figure 5: Neural Style Examples (FranÇois Chollet, 2018)

**2.1 Related Works**

There has been several work done on detecting deep fakes before now.

Most of the early detection mechanisms in the early 2018 included detection via biological signals. These works on biological signals involved looking for unusual behaviour in images such as blinking rate (lack of blinking), facial abnormalities, movement abnormalities.

While these may work on specific types of manipulations, they do not work well on generalized face image manipulation techniques.

There are also tools that can be used by journalists in their quest for information verification. As described by Brandtzaeg et al. [Petter, 2016; Petter, Asbjørn ad Maria et al 2018] tools that are frequently used by journalists for media verification are TinEye 1 , Google Reverse Image search and InVid Project.

In the last two decades, interest in virtual face manipulation has rapidly increased. A comprehensive state-of-the-art report has been published by Hyeongwoo et al. 2018. In particular, Christoph et al. 1997, presented an image-based approach called Video Rewrite to automatically create a new video of a person with generated mouth movements. With Video Face Replacement, Kevinn et al. 2011 presented one of the first automatic face swap methods. Using single-camera videos, they reconstruct a 3D model of both faces and exploit the corresponding 3D geometry to warp the source face to the target face. Garrido et al. 2014 presented a similar system that replaces the face of an actor while preserving the original expressions. VDub (Pablo et al, 2015) uses high-quality 3D face capturing techniques to photo-realistically alter the face of an actor to match the mouth movements of a dubber.

Thies et al. 2015 demonstrated the first real-time expression transfer for facial re-enactment. Based on a consumer level RGB-D camera, they reconstruct and track a 3D model of the source and the target actor. The tracked deformations of the source face are applied to the target face model. As a final step, they blend the altered face on top of the original target video. Face2Face, proposed by Justus et al. 2016, is an advanced real-time facial reenactment system, capable of altering facial movements in commodity video streams, e.g., videos from the internet. They combine 3D model reconstruction and image-based rendering techniques to generate their output. The same principle can be also applied in Virtual Reality in combination with eye-tracking and reenactment (Justus et al. 2018) or be extended to the full body.

Hyeongwoo et al. 2018 learn an image-to-image translation network to convert computer graphic renderings of faces to real images. Instead of a pure image-to-image translation network, NeuralTextures (Justus et al. 2019) optimizes a neural texture in conjunction with a rendering network to compute the reenactment result. In comparison to Deep Video Portraits (Kim et al. 2018), it shows sharper results, especially, in the mouth region. Supasorn et al, 2017) learned the mapping between audio and lip motions, while their compositing approach builds on similar techniques to Face2Face (Hadar et al. 2017). Hadar et al. 2017 present a reenactment method, Bringing Portraits to Life, which employs 2D warps to deform the image to match the expressions of a source actor. They also compare to the Face2Face technique and achieve similar quality. Recently, several face image synthesis approaches using deep learning techniques have been proposed. Zhihe et al. 2017 provide an overview.

Generative adversarial networks (GANs) are used to apply Face Aging (Justus et al. 2015), to generate new viewpoints (Rui et al. 2017), or to alter face attributes like skin color (Zhihe et al 2018). Deep Feature Interpolation (Paul et al 2017) shows impressive results on altering face attributes like age, mustache, smiling etc. Similar results of attribute interpolations are achieved by Fader Networks (Luca et al. 2018). Most of these deep learning based image synthesis techniques suffer from low image resolutions. Recently, Tero et al. 2018 have improved the image quality using progressive growing of GANs, producing high-quality synthesis of faces. (Andreas et al. 2019). There have also been a work on detecting mamipulation traces using a a specifical pre-processing module based on the convolution layers, namely AMTEN, to predict manipulation traces which also use a fake face detector, namely AMTENnet, to learn discriminative features from manipulation traces. (Zhiqing et al 2021).

There are also other efforts that have been used in an attempt to combat deep fakes which includes (1) legislation and regulation, (2) corporate policies and voluntary action, (3) education and training, and (4) anti-deepfake technology that includes deepfake detection, content authentication, and deepfake prevention (Mika, 2019).

**3. METHODOLOGY**

**3.0 Data Collection**

For this experiment, various data sets consisting of different categories of deep fakes and also real images were needed to run a multi-classification on them. With this set of data, a neural network model was trained to learn the manipulations and differences between fake images and real images. The model generated from this training was then used to detect images not previously trained by the model. Tensorflow which comes bundled with Keras, EfficientetB0 and Google Colab are also needed for this experiment.

All the data for this research were acquired from the popular data science competition platform: Kaggle. For the real images, low resolution and high-resolution images were gotten from the CelebA and CelebA-HQ datasets respectively. Also for real images, images from youtube were obtained from the Youtube dataset within Forensics++ on Kaggle. Thus, real face images under internet scenarios are simulated as real as possible.

The fake images were obtained from the FaceForensics++ dataset from Kaggle. FaceForensics++ contains manipulated video sequences which comprise Deepfakes, Faceswap, Face2Face for face expression manipulation, and NeuralTextures. FaceForensics++ also contained original video sequences of Youtube videos from which the manipulated sequences were generated. These Youtube videos will be added to the real images dataset.

For real images, the CelebA low-resolution dataset contains approximately 203k images of celebrities, the CelebA-HQ dataset contains high-resolution 30k images while the Youtube dataset contains 1000 video sequences of youtube videos.

For fake images, within the FaceForensics++ dataset, there are 1000 video sequences each of Deepfakes, Faceswap, Face2Face and NeuralTextures.

This set of data was chosen because it provides us with a good distribution of real images from low-resolution, high-resolution as well as images from youtube. Also, the fake images are from different sources of face image manipulation techniques thus covering a wide spectrum of the different types of fake images that can be generated.

**3.1 Featured Engineering**

From the datasets, it can be seen that there is a class imbalance mostly seen between the CelebA, CelebA-HQ datasets compared to the other classes

The data inForensics++ are in video format and so there is a need to extract images from them to use for this experiment. Since the CelebA and CelebA-HQ datasets are already in image format, there is no need to do any extraction on these.

For the video sequences in Forensics++, every 25th frame for each video was extracted and compiled category by category. To do this, a script was written to extract every 25th frame for each video in each of the categories. For example, for Deepfake sequences, (total no. of frames for 1000 video sequences) / 25 frames were gotten.

Resizing of these images was not done on either of these data during this process because they will be resized at a later step to 128 \* 128 before being used in training.

To extract the images from the video sequences, OpenCV and a state-of-the-art Face-Recognition model which has an accuracy of 99.38% on the Labeled Faces in the Wild benchmark. With these tools, a python script was created to extract the images. This python script generated the following number of images:

Deepfakes: 20,675

Face2Face: 20,883

Faceswap: 16,762

NeuralTextures: 16,759

Youtube: 20,885

In addition, the following number of images from CelebA and CelebA-HQ images are available

CelebA: 203k

CelebA-HQ: 30k

As see above, there is a class imbalance amongst the different classes. Some dataset like CelebA has a lot more data available than others. To prevent this class imbalance from affecting our training, some adjustment needs to be made to the samples to be taken for training for bot binary classification and multi-class classification.

For the binary classification into real and fake images, it was decided to combine the real images into one dataset and the fake images into one dataset.

A fixed number of 66,994 was chosen to use for both real and fake images. For the fake images category, this consisted of images from 4 different sources: Deepfakes, Face2Face, FaceSwap and NeuralTextures. For the real images, this consisted of images from CelebA, CelebA-HW and Youtube datasets in roughly equal proportions.

Another python script was created to distribute the images for each category (real, fake) into train, validation and test sets as shown in the table below:

**Table 1: Distribution of images dataset for binary classification**

| Category | Train (80%) | Validation (5%) | Test (10%) |
| --- | --- | --- | --- |
| Real | 53,595 | 3,350 | 10,049 |
| Fake | 53,595 | 3,350 | 10,049 |

For the multi-class classification into 7 categories, training set samples taken was 14,000 for each class, validation samples were 750 and test samples were 2000.

**Table 2: Distribution of images dataset for binary classification**

| Category | Train(83%) | Validation(5%) | Test (12%) |
| --- | --- | --- | --- |
| Each Category | 14,000 | 750 | 2000 |

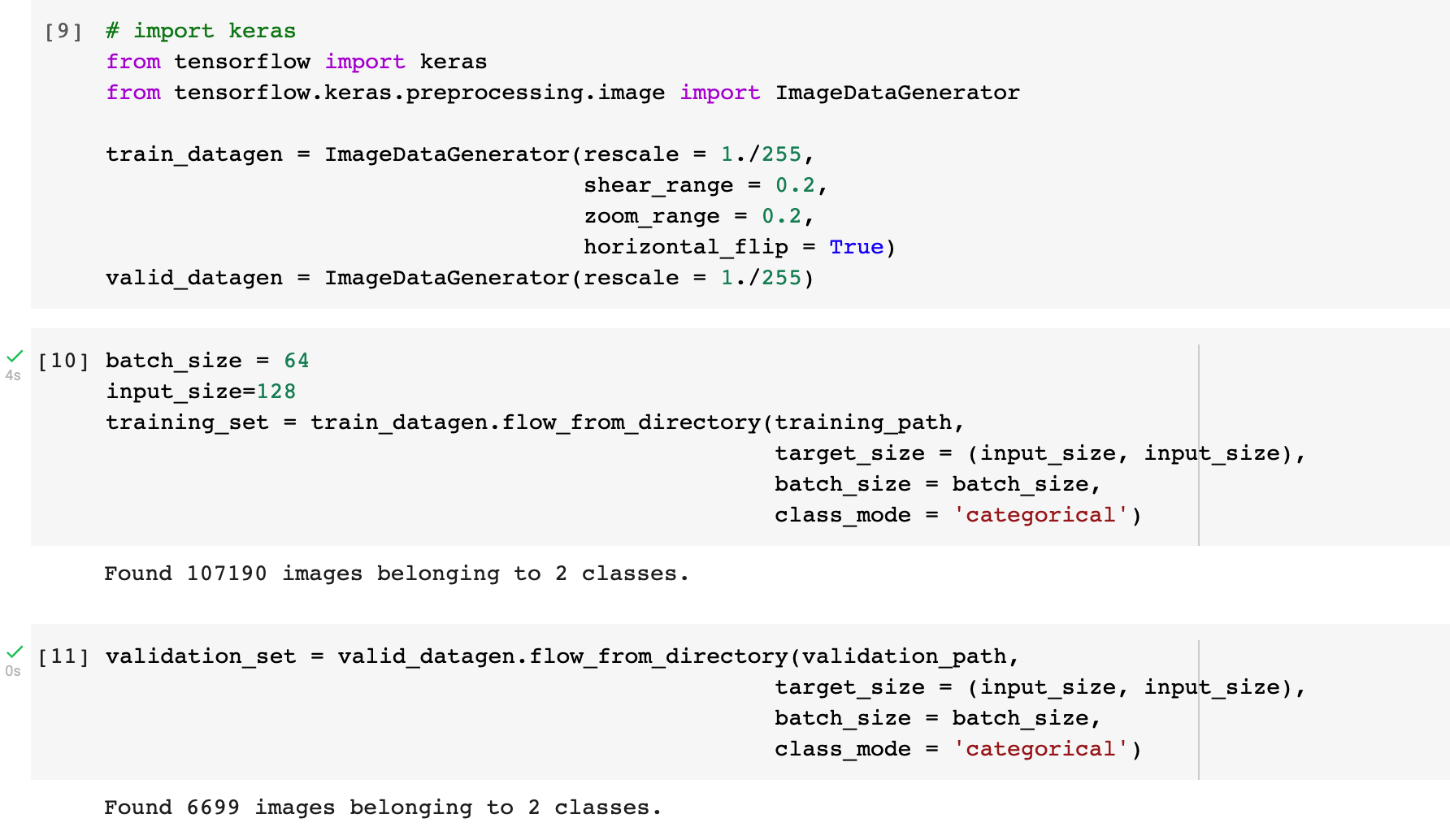
Both datasets were then zipped and uploaded to Google Drive.

**4. Experiments, Results And Discussion**

This experiment was conducted using the Tensorflow and Keras framework on Google Colab.

Before passing the data into our neural network for training, it was needed to do some transformations of the data into an acceptable format. The purpose of this transformation is to create varieties of the same image which would be passed to the network at each epoch.

The dataset was firstly imported from Google drive and unzipped. Using Keras preprocessing ImageDataGenerator, transformations were done on the images by applying a zoom range of 0.1, a shear range of 0.5 and randomly flipping the images horizontally with a rescaling factor of 1/255. was applied. The images were then resized to 128 \* 128 with a batch size of 64. The class mode used was `categorical`.



**4.1. Architecture of The Neural Network**

First, a baseline is needed to beat, so a base model was created to see how it performs and then compare with other network adaptations.

Throughout the experiment, Keras SeparableConv2D was used as the Conv2D layer.

**4.1.1 The architecture of the base model**

The base model used 3 convolutional layers and three densely connected layers.

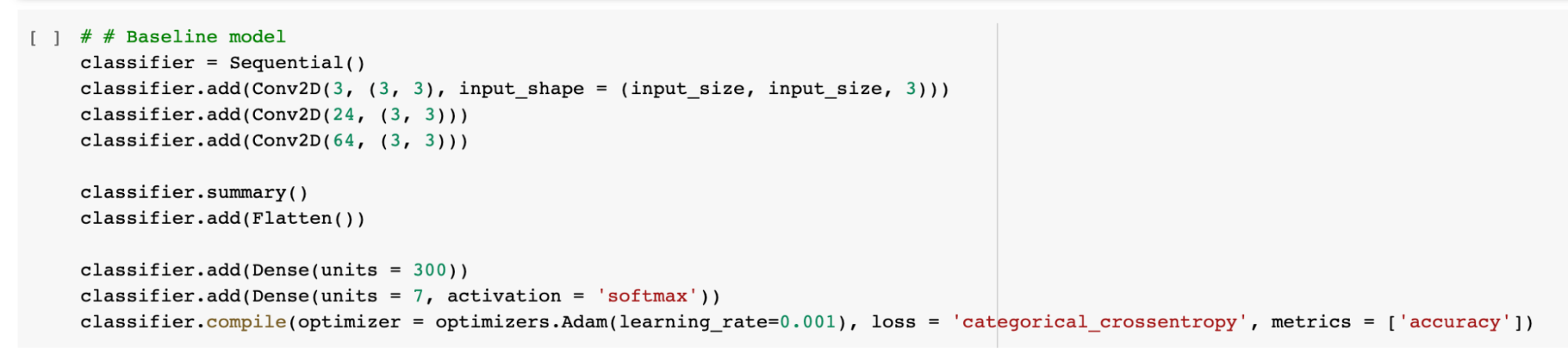
The first layer of the base model uses a 3 \* 3 kernel size, a filter size of 3, with a stride of 1 and `valid` padding.

The second and third layers use the same configuration as the first layer but the filter sizes are increased to 24 and 64 respectively. To fully exploit the features of the previous layer, the number of the convolutional kernels in the successive layer should not be less than the number of channels of the input feature map.

The network is then flattened before passing it to the densely connected layers. The first dense layer contains 300 neurons while the second dense layer contains 128 neurons.

The last dense layer has 2 neurons corresponding to the output classes with a Softmax activation function.

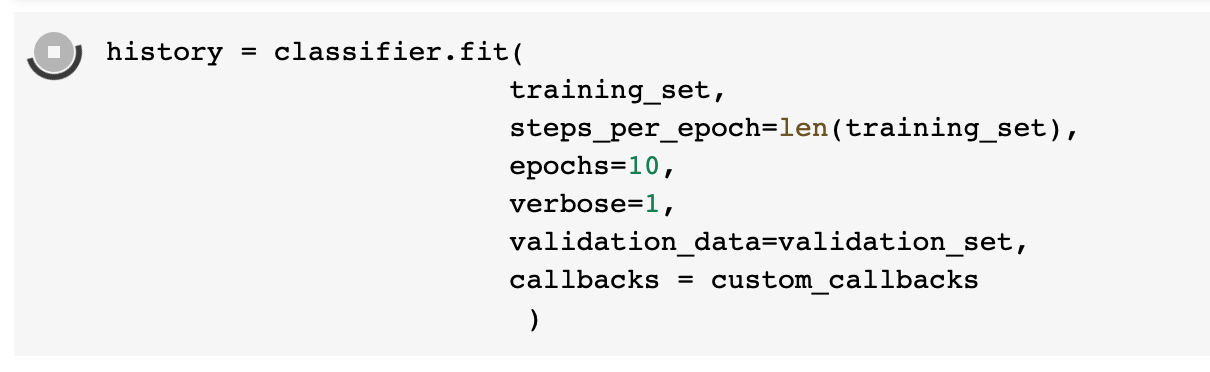
The model is then compiled using `Adam` optimizer with a learning rate of 0.001. `categorical\_crossentropy` was chosen for the loss while `accuracy` was the metric.



The base model is then called with the fit method for 10 epochs and the steps per epoch were set to be the length of the training set which can be calculated as (number of training data/batch\_size).

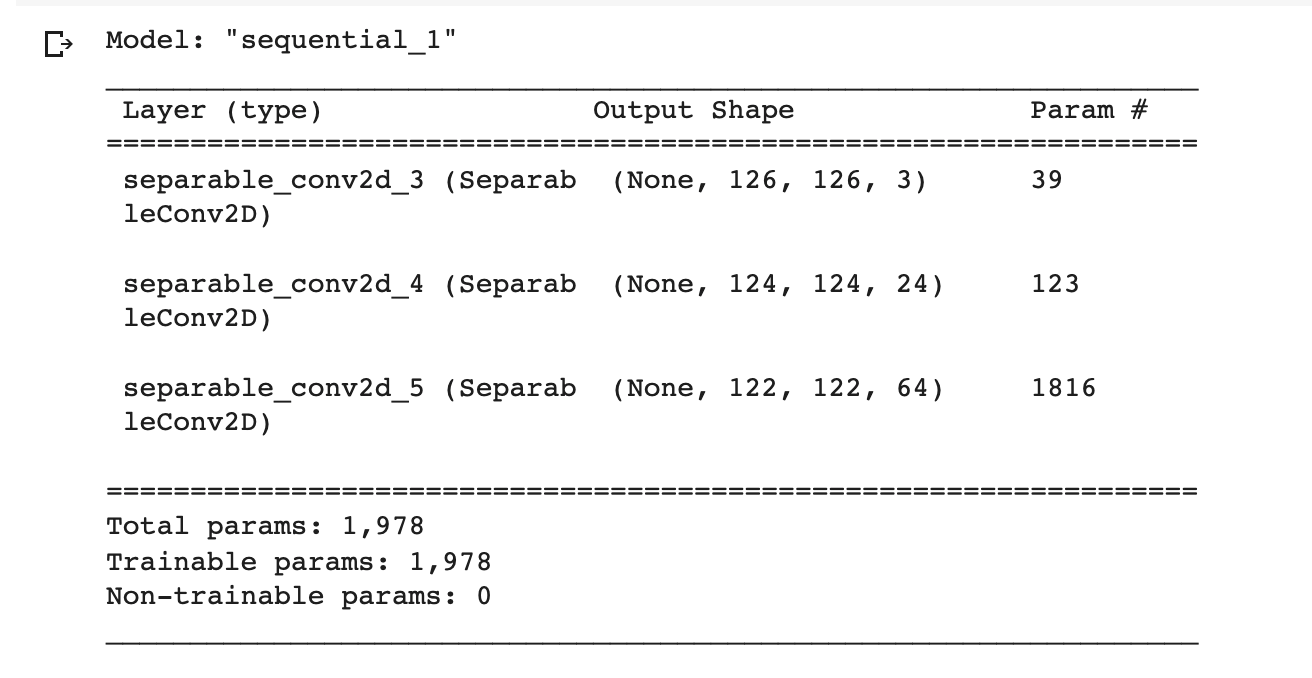
The performance of the model is monitored at every epoch by passing a Callback to the fit method. This Callback monitors the validation loss for minimization and stops training if it doesn’t decrease after 5 straight epochs. Also, another Keras Callback is added which saves the model at the end of every epoch once it makes a new improvement on its validation loss. These two callbacks can be accomplished with EarlyStopping and ModelCheckpoint from the Keras Callback Class.





**Figure 6: Fitting the model**

The figure below shows the summary of the base model network.



**Figure 7: Summary of the base model**

For multi-class classification with the base model, the change made was to use 7 as the number of neurons in the last dense layer with a Softmax activation function.

**4.1.2 The architecture of the transfer learning model (EfficientNetB0)**

To compare the base model, a test was also done with the same data on another network. For this network, transfer learning from the pre-trained state of the art network EfficientNet was used.

EfficientNet is a convolutional neural network architecture and scaling method from Google that uniformly scales all dimensions of depth/width/resolution using a compound coefficient. Unlike conventional practice that arbitrary scales these factors, the EfficientNet scaling method uniformly scales network width, depth, and resolution with a set of fixed scaling coefficients.

What is needed from Efficient Net is its convolutional base layers (i.e the learned features) so the top layers comprising the densely connected layers are not used. Tthe same input size of 128\*128 and max-pooling as the configuration for the base EfficientNet called EfficientNetB0 is used. The EfficientNetB0 is added to our `SeparableConv2D` model.

3 densely connected layers just like the base model was used with some variations.

The first two-layer uses 300 neurons each with a `relu` activation. Also, l1 and l2 regularization are added with values of 0.001 each. A dropout layer is also added after each dense layer with a value of 0.2. The purpose of the regularizers and dropout is to prevent overfitting to our training data.

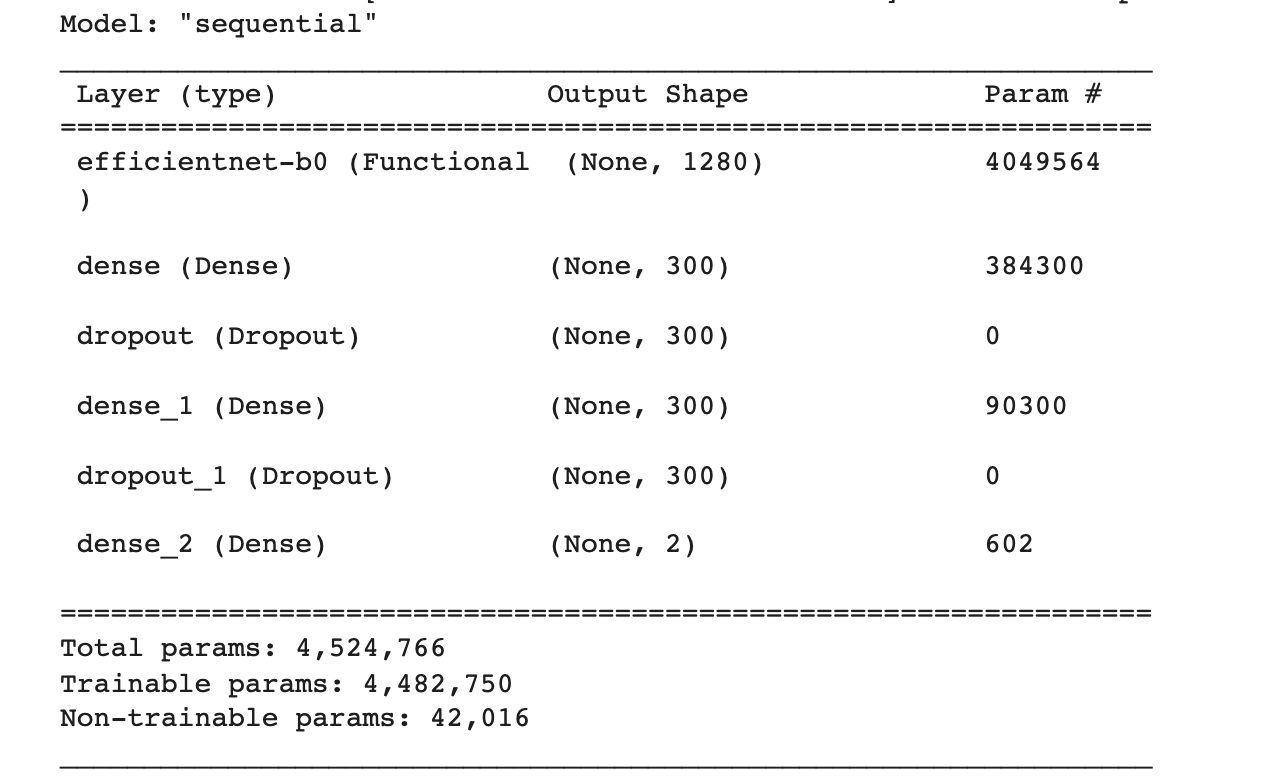
The last layer utilized 2 neurons with a Softmax activation function. The model is then compiled with Adam optimizer, categorical\_crossentropy as loss and accuracy as the metric.



**Figure 8: Structure of the EfficientNet network with transfer learning**

The model was fitted in the same pattern as the base model utilizing the same Callbacks for 10 epochs.

Below image is the summary of the model with Efficient Net.



**Figure 9: Summary of the EfficientNet model with transfer learning**

For multi-class classification with the pre-trained EfficientNetB0, the change made was to use 7 as the number of neurons in the last dense layer with a Softmax activation function.

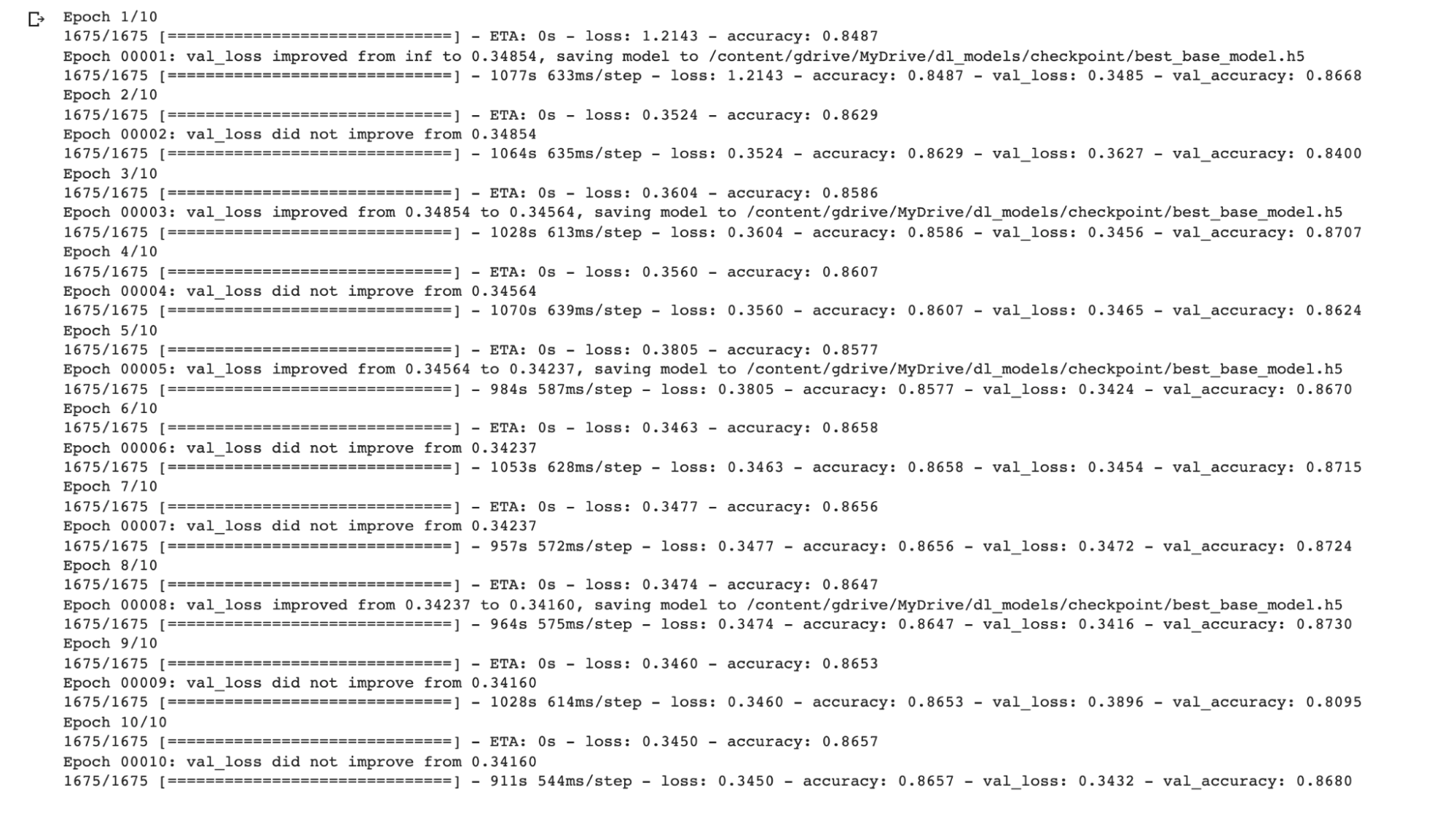
**RESULTS AND DISCUSSION**

Both networks were trained each for the binary classification for real and fake images and multi-class classification on the different categories of images.

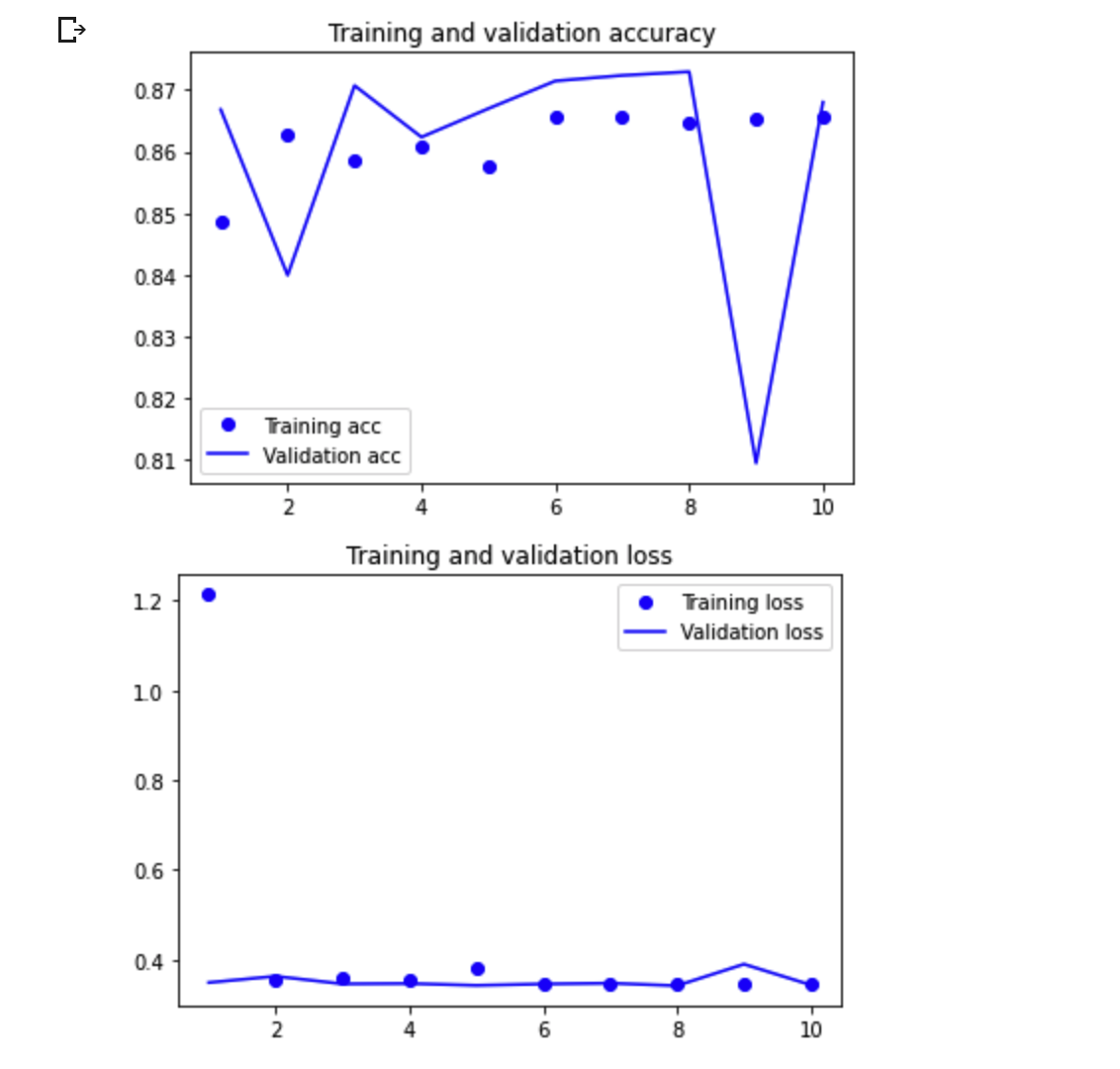
**4.2.1 Training Results On Binary Classification**

**Base Model**

For the base model, after training for 10 epochs on binary classification between real and fake images, the best model arrived at a training accuracy of 0.8647 and a validation accuracy of 0.8730. It can be seen that the base model does a fairly good job of predicting which classes the images belongs to. From the image below, it can be seen that the training accuracy was more stable than the validation accuracy.



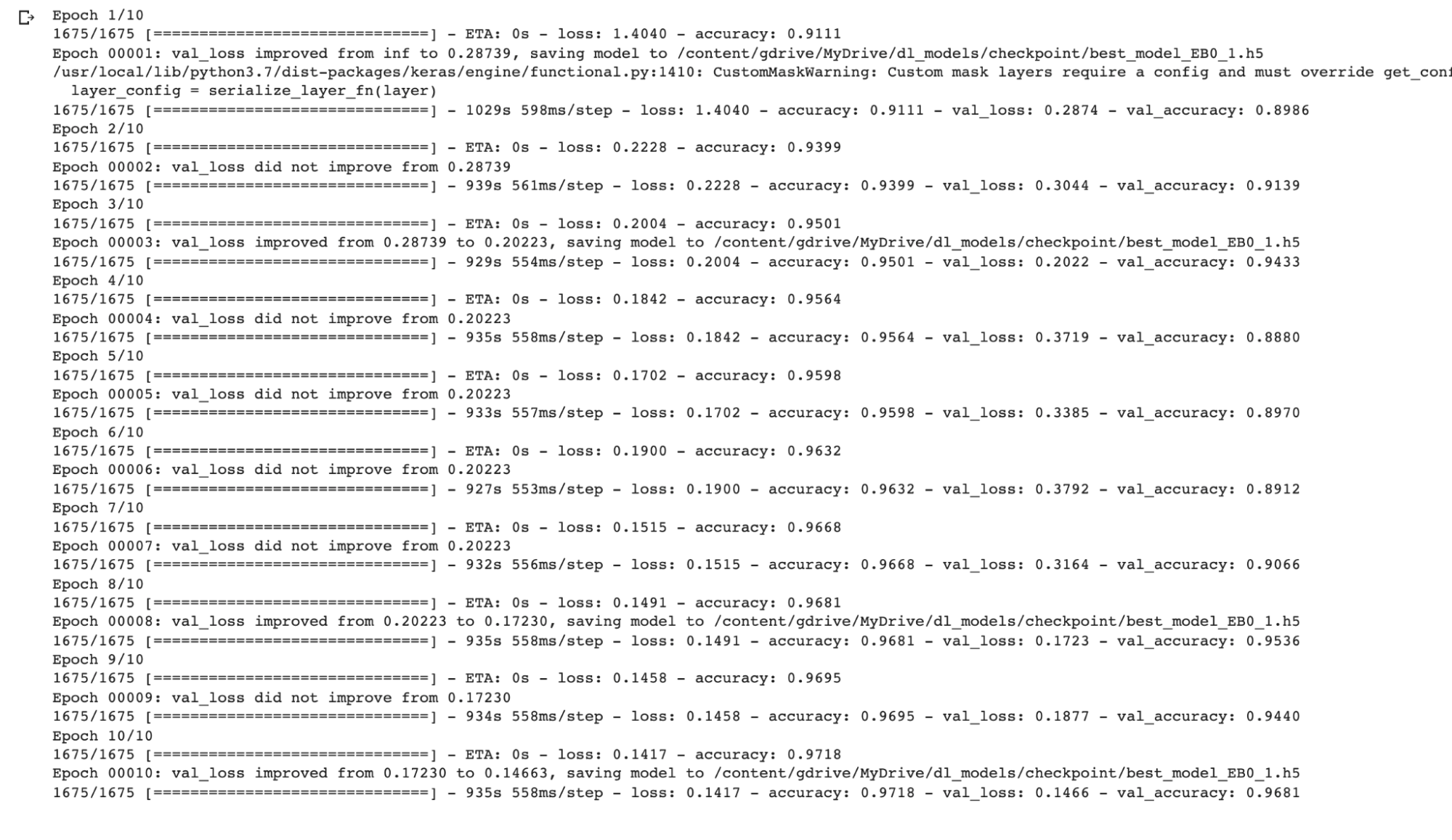
**Figure 10: Training results of base model**



**Figure 11: Training accuracy and validation loss for base model (binary classification)**

**EfficientNetB0**

In comparison, the model with transfer learning using the pre-trained EfficientNetB0 gave a training accuracy of 0.9718 and a validation accuracy of 0.9681. The model with Efficient Net gave better accuracy on both training and validation data.



**Figure 12: Training results for EfficientNet model**

**4.2.2 Testing On Binary Classification**

The model was then tested on the test data which was not used during training for binary classification. Before carrying out the prediction, the images were converted to the same format as the training data. For each image, it was convert to an array and rescale to 1/255. An extra dimension was also added because the predict method expects a batch. The sequence of input arrays is then stacked vertically to make a single array and feed the image to the predict method. The index of the maximum of the NumPy array is taken and this corresponds to the category the image belongs to with 0 equal to `fake` and q equal to `real`.

**Base model**

After carrying out testing, the base model gave a prediction of 99.5% for the fake images and prediction accuracy of 74.8% for real images.

**EfficientNetB0**

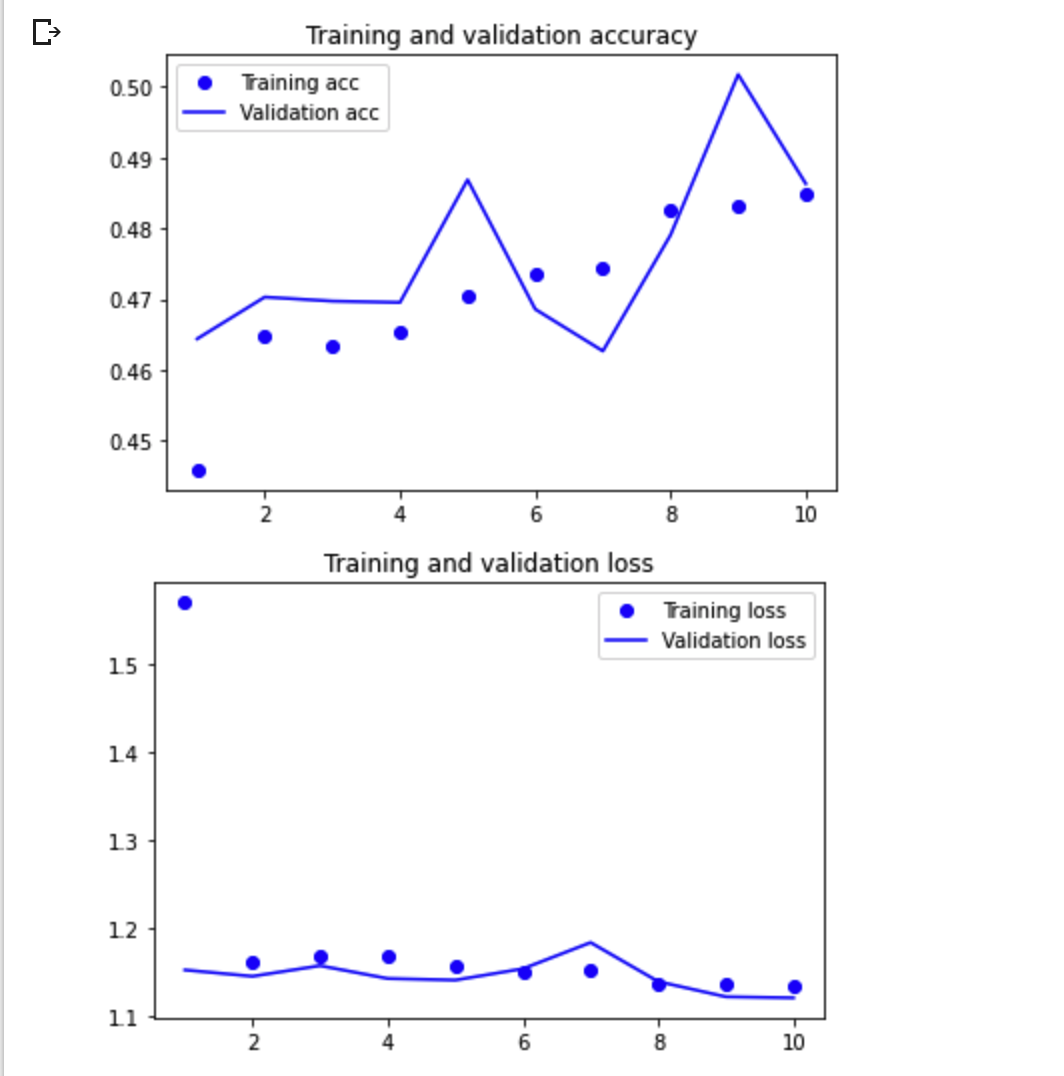
After testing, the model with transfer learning gave a prediction accuracy of 99% for fake images and a prediction accuracy of 94.5% for real images.

**4.2.3 Training Result On Multi-Class Classification**

Also, training was done using both networks on both binary and multi-class classification.

**Base Model**

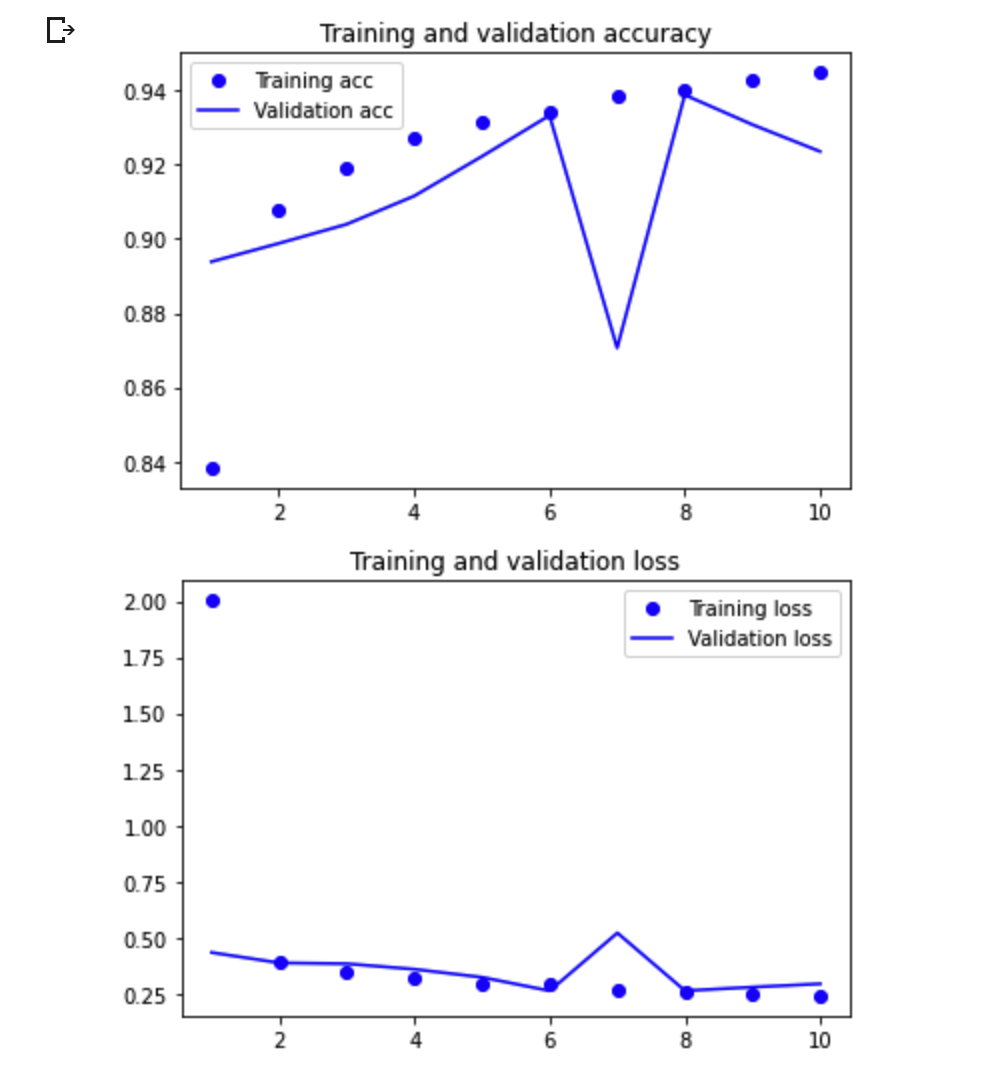
For the base model, after training for 10 epochs on multi-class classification amongst CelebA, CelebA-HQ, Youtube, Deepfakes, Face2Face, FaceSwap and Neural Textures, the best model with minimum validation loss arrived at a low training accuracy of 0.4847 and a low validation accuracy of 0.4863.



**Figure 13: Training accuracy and validation loss for base model (multi-classification)**

**EfficientNetB0**

In comparison, the best model on training with transfer learning using EfficientNetB0 for multi-classification gave a high training accuracy of 0.9343 and a high validation accuracy of 0.9333.



**Figure 14: Training accuracy and validation loss for EfficientNet model (multi-classification)**

**4.2.4 Testing On Multi-Class Classification**

Testing was then carried out on the test data which was not used during training for multi-class classification. Before carrying out the prediction, the same preprocessing tasks that were done on each image before binary-classification testing needs to be carried out. For each image, it was converted to an array and rescale to 1/255 and then an extra dimension was added because the predict method expects a batch. The sequence of input arrays is then stacked vertically to make a single array and feed the image to the predict method. The index of the maximum of the NumPy array is taken and this corresponds to the category the image belongs to as shown in the figure below.

Table 3: Index of prediction corresponding to image class

| **Category** | **Index** |
| --- | --- |
| CelebA | 0 |
| CelebA-HQ | 1 |
| Deepfakes | 2 |
| Face2Face | 3 |
| Faceswap | 4 |
| NeuralTextures | 5 |
| Youtube | 6 |

**Base model**

After carrying out testing, the base model gave the following results shown in percentage (%):

Table 4: Prediction result of multi-class classification with a simple base CNN model (where N/A represents no value)

|  | Predicted Class | CelebA | CelebA-HQ | Youtube | Deepfakes | Face2Face | Faceswap | NeuralTextures |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Actual Class |  |  |  |  |  |  |  |  |
| CelebA |  | 98.8 | 0.65 | 0.2 | 0.25 | 0.05 | 0.05 | N/A |
| CelebA-HQ |  | 0.35 | 99.4 | N/A | 0.05 | 0.15 | N/A | 0.05 |
| Youtube |  | N/A | 0.15 | 30.65 | 23.85 | 8.0 | 36.5 | 0.85 |
| Deepfakes |  | N/A | 0.1 | 18.65 | 55.35 | 6.35 | 18.8 | 0.55 |
| Face2Face |  | N/A | 0.1 | 20.46 | 27.66 | 26.11 | 25.01 | 0.65 |
| FaceSwap |  | N/A | 0.15 | 28.35 | 23.1 | 7.19 | 40.4 | 0.8 |
| NeuralTextures |  | N/A | 0.15 | 32.65 | 24.5 | 7.3 | 34.55 | 0.85 |

Table 5: Prediction result of multi-class classification with a simple base CNN model (where \* represents less than 10%)

|  | Predicted Class | CelebA | CelebA-HQ | Youtube | Deepfakes | Face2Face | Faceswap | NeuralTextures |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Actual Class |  |  |  |  |  |  |  |  |
| CelebA |  | 98.8 | \* | \* | \* | \* | \* | \* |
| CelebA-HQ |  | \* | 99.4 | \* | \* | \* | \* | \* |
| Youtube |  | \* | \* | 30.65 | 23.85 | \* | 36.5 | \* |
| Deepfakes |  | \* | \* | 18.65 | 55.35 | \* | 18.8 | \* |
| Face2Face |  | \* | \* | 20.46 | 27.66 | 26.11 | 25.01 | \* |
| FaceSwap |  | \* | \* | 28.35 | 23.1 | \* | 40.4 | \* |
| NeuralTextures |  | \* | \* | 32.65 | 24.5 | \* | 34.55 | \* |

**Efficient Net**

The model with transfer learning gave the following results in percentage (%):

Table 6: Prediction result of multi-class classification with pre-trained convnet efficient net model (where N/A represents no value)

|  | Predicted Class | CelebA | CelebA-HQ | Youtube | Deepfakes | Face2Face | Faceswap | NeuralTextures |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Actual Class |  |  |  |  |  |  |  |  |
| CelebA |  | 99.65 | 0.3 | 0.05 | N/A | N/A | N/A | N/A |
| CelebA-HQ |  | N/A | 100% | N/A | N/A | N/A | N/A | N/A |
| Youtube |  | N/A | 0.05 | 91.65 | 1.15 | 3.0 | 2.7 | 1.45 |
| Deepfakes |  | N/A | N/A | 4.6 | 94.1 | N/A | 1.2 | 0.1 |
| Face2Face |  | N/A | N/A | 4.45 | 0.1 | 95.25 | N/A | 0.2 |
| FaceSwap |  | N/A | N/A | 4.5 | 0.05 | 0.95 | 94.5 | N/A |
| NeuralTextures |  | N/A | N/A | 18.75 | 0.55 | 2.5 | 0.1 | 78.10 |

Table 7: Prediction result of multi-class classification with pre-trained convnet efficient net model (where \* represents less than 10%)

|  | Predicted Class | CelebA | CelebA-HQ | Youtube | Deepfakes | Face2Face | Faceswap | NeuralTextures |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Actual Class |  |  |  |  |  |  |  |  |
| CelebA |  | 99.65 | \* | \* | \* | \* | \* | \* |
| CelebA-HQ |  | \* | 100% | \* | \* | \* | \* | \* |
| Youtube |  | \* | \* | 91.65 | \* | \* | \* | \* |
| Deepfakes |  | \* | \* | \* | 94.1 | \* | \* | \* |
| Face2Face |  | \* | \* | \* | \* | 95.25 | \* | \* |
| FaceSwap |  | \* | \* | \* | \* | \* | 94.5 | \* |
| NeuralTextures |  | \* | \* | 18.75 | \* | \* | \* | 78.10 |

**4.3 DISCUSSION**

From the results obtained it can be seen that the base model did fairly well on binary-class detection between the two classes of fake and real images. It had a high accuracy of detecting fake images but poor accuracy of detecting real images. While it had a very poor result when it came to multi-class classification amongst the different categories with it having only good accuracy for CelebA and CelebA-HQ images.

In comparison, the pre-trained ConvNet with transfer learning gave high accuracy on binary classification as well as high accuracy for each of the categories in the multi-class classification predictions. However, it had only a fairly good accuracy for detecting Neural textures with an accuracy of 78.1%.

It is observed that the detection of NeuralTextures seems to be much harder for both networks as they both had the lowest accuracy in this category and the highest accuracies in detecting the low and high-resolution celebrity pictures.

From this experiment, it shows that previously learned features are very much effective at detecting face image manipulation techniques even though the previously learned features are from a general category of images. Even though the base model gave a higher accuracy for the fake images on testing in binary classification, it did considerably poorly once it came to the multi-class classification where it had to learn more specific features and representations for each FIM category.

To further improve the accuracy of the network with pre-trained ConvNet, using a much larger dataset is suggested as there will be more data to learn from especially for the neural textures.

**5.1 Conclusion**

In this research, it was shown how transfer learning with a generalized pre-trained convnet (EfficientNet) can detect face image manipulation techniques. From the experiment, it is observed that high accuracy in detection across different classes of FIMs was obtained as compared to a simple model which could only perform considerably well on just a broader classification between real and fake images. It can thus be concluded that the features learned from the general model are very much transferable to detection of deep fakes. In the future, the results achieved in this study can be improved upon by training on even larger datasets and applying more state of the art hyperparameter fine tuning.

# **REFERENCES**

Ahmed Maher: Towards Data Science - Understanding Face Shifter: a new face-swapping model (2020)

Andreas Rossler, Davide Cozzolino, Luisa Verdoliva, Christian Riess, Justus Thies, (2019). Matthias Nießner: FaceForensics++: Learning to Detect Manipulated Facial Images

Christoph Bregler, Michele Covell, and Malcolm Slaney. Video rewrite: Driving visual speech with audio. In 24th Annual Conference on Computer Graphics and Interactive Techniques, SIGGRAPH ’97, pages 353–360, 1997.

David Guera, Edward J. Delp: Deepfake Video Detection Using Recurrent Neural Networks, 2018

Deepfakes github. https://github.com/ deepfakes/faceswap. 2018

Fakeapp. https://www.fakeapp.com/. 2018

FranÇois Chollet: Deep Learning with Python (2018)

Guillaume Lample, Neil Zeghidour, Nicolas Usunier, Antoine Bordes, and Marc’Aurelio Ranzato Ludovic Denoyer. Fader networks: Manipulating images by sliding attributes. CoRR, abs/1706.00409, 2017.

Hadar Averbuch-Elor, Daniel Cohen-Or, Johannes Kopf, and Michael F. Cohen. Bringing portraits to life. ACM Transactions on Graphics (Proceeding of SIGGRAPH Asia 2017), 36(4):to appear, 2017

Hyeongwoo Kim, Pablo Garrido, Ayush Tewari, Weipeng Xu, Justus Thies, Matthias Nießner, Patrick Perez, Chris- ´ tian Richardt, Michael Zollhofer, and Christian Theobalt. ¨ Deep Video Portraits. ACM Transactions on Graphics 2018 (TOG), 2018. 3

Jessica Fridrich and Jan Kodovsky. Rich Models for Steganalysis of Digital Images. IEEE Transactions on Information Forensics and Security, 7(3):868–882, June 2012

Justus Thies, Michael Zollhofer, Matthias Nießner, Levi Val- ¨ gaerts, Marc Stamminger, and Christian Theobalt. Real-time expression transfer for facial reenactment. ACM Transactions on Graphics (TOG) - Proceedings of ACM SIGGRAPH Asia 2015, 34(6):Art. No. 183, 2015.

Justus Thies, Michael Zollhofer, and Matthias Nießner. De- ¨ ferred neural rendering: Image synthesis using neural textures. ACM Transactions on Graphics 2019 (TOG), 2019

Justus Thies, Michael Zollhofer, Marc Stamminger, Chris- ¨ tian Theobalt, and Matthias Nießner. Face2Face: Real-Time Face Capture and Reenactment of RGB Videos. In IEEE Conference on Computer Vision and Pattern Recognition, pages 2387–2395, June 2016.

Justus Thies, Michael Zollhofer, Marc Stamminger, Christian Theobalt, and Matthias Nießner. FaceVR: Real-Time Gaze-Aware Facial Reenactment in Virtual Reality. ACM Transactions on Graphics (TOG), 2018.

Justus Thies, Michael Zollhöfer, Marc Stamminger, Christian Theobalt, Matthias Nießner: Face2Face: Real-Time Face Capture and Reenactment of RGB Videos (2019)

Kevin Dale, Kalyan Sunkavalli, Micah K. Johnson, Daniel Vlasic, Wojciech Matusik, and Hanspeter Pfister. Video face replacement. ACM Trans. Graph., 30(6):130:1–130:10, Dec. 2011.

Leon A. Gatys, Alexander S. Ecker, Matthias Bethge. A Neural Algorithm of Artistic Style (2015)

Luca D’Amiano, Davide Cozzolino, Giovanni Poggi, and Luisa Verdoliva. A PatchMatch-based Dense-field Algorithm for Video Copy-Move Detection and Localization. IEEE Transactions on Circuits and Systems for Video Technology, in press, 2018.

Marek Kowalski. [n. d.]. faceswap. https://github.com/MarekKowalski/ FaceSwap (2019)

Michael Zollhofer, Justus Thies, Darek Bradley, Pablo ¨ Garrido, Thabo Beeler, Patrick Peerez, Marc Stamminger, ´ Matthias Nießner, and Christian Theobalt. State of the art on monocular 3d face reconstruction, tracking, and applications. Computer Graphics Forum, pp. 523–550, 2018

Mika Westerlund: The Emergence of Deepfake Technology: A Review (2019)

Pablo Garrido, Levi Valgaerts, Ole Rehmsen, Thorsten Thormaehlen, Patrick Perez, and Christian Theobalt. Automatic ´ face reenactment. In IEEE Conference on Computer Vision and Pattern Recognition, pp. 4217–4224, 2014.

Pablo Garrido, Levi Valgaerts, Hamid Sarmadi, Ingmar Steiner, Kiran Varanasi, Patrick Perez, and Christian ´ Theobalt. Vdub: Modifying face video of actors for plausible visual alignment to a dubbed audio track. Computer Graphics Forum, 34(2):193–204, 2015.

Paul Upchurch, Jacob Gardner, Geoff Pleiss, Robert Pless, Noah Snavely, Kavita Bala, and Kilian Weinberger. Deep feature interpolation for image content changes. In IEEE Conference on Computer Vision and Pattern Recognition, 2017.

Petter Bae Brandtzaeg, Asbjørn Følstad, and Maria Ángeles Chaparro Domínguez. 2018. How Journalists and Social Media Users Perceive Online Fact-checking and Verification Services. Journalism Practice (2018).

Petter Bae Brandtzaeg, Marika Lüders, Jochen Spangenberg, Linda Rath-Wiggins, and Asbjørn Følstad. 2016. Emerging Journalistic Verification Practices Concerning Social Media. Journalism Practice (2016).

Rui Huang, Shu Zhang, Tianyu Li, and Ran He. Beyond face rotation: Global and local perception GAN for photorealistic and identity preserving frontal view synthesis. In IEEE International Conference on Computer Vision, 2017.

Saniat Javid Sohrawardi, Sovantharith Seng, Akash Chintha, Bao Thai, Andrea Hickerson, Raymond Ptucha, Matthew Wright. DeFaking Deepfakes: Understanding Journalists’ Needs for Deepfake Detection (2019)

Supasorn Suwajanakorn, Steven M. Seitz, and Ira Kemelmacher-Shlizerman. Synthesizing Obama: learning lip sync from audio. ACM Transactions on Graphics (TOG), 36(4), 2017.

Tero Karras, Timo Aila, Samuli Laine, and Jaakko Lehtinen. Progressive Growing of GANs for Improved Quality, Stability, and Variation. In International Conference on Learning Representations, 2018

Visage Technologies: Face2Face: live video editing of facial expressions (2016)

Yongyi Lu, Yu-Wing Tai, and Chi-Keung Tang. Conditional cyclegan for attribute guided face image generation. In European Conference on Computer Vision, 2018.

Zhihe Lu, Zhihang Li, Jie Cao, Ran He, and Zhenan Sun. Recent progress of face image synthesis. In IAPR Asian Conference on Pattern Recognition, 2017.

Zhiqing Guo a, Gaobo Yang a,∗, Jiyou Chen a, Xingming Sun : Fake face detection via adaptive manipulation traces extraction network (2021)