Project Report

On

Deepfake Detection

Submitted
In partial fulfilment
For the award of the Degree of

PG-Diploma in Artificial Intelligence

(C-DAC, ACTS (Pune))

Guided By:

Dr. Krishnanjan B.

Submitted By:

Ashwini Chaudhari (230340128006)

Sanjay Jangid (230340128014)

Prathamesh Devkar (230340128018)

Saif Shaikh (220340128023)

Yash Dubey (220340128031)

Centre for Development of Advance Computing

(C-DAC), ACTS (Pune-411008)

ABSTRACT

The growing computation power has made the deep learning algorithms so powerful that creating an indistinguishable human synthesized video popularly called as deep fakes have become very simple. Scenarios where this realistic face swapped deep fakes are used to create political distress, fake terrorism events, revenge porn, blackmail peoples are easily envisioned. In this work, we describe a new deep learning-based method that can effectively distinguish AI-generated fake videos from real videos. Our method is capable of automatically detecting the replacement and reenactment deep fakes. We are trying to use Artificial Intelligence (AI) to fight Artificial Intelligence (AI). Our system uses a Multi-task cascaded convolution neural network(MTCNN) to extract the frame-level features and these features and further used to train the mesonet to classify whether the video is subject to any kind of manipulation or not, i.e whether the video is deep fake or real video. To emulate the real time scenarios and make the model perform better on real time data, we evaluate our method on large amount of balanced and mixed data-set prepared by mixing the various available data-set like Face-Forensic++. We also show how our system can achieve competitive result using very simple and robust approach.

Introduction

In the world of ever growing Social media platforms, Deepfakes are considered as the major threat of the AI. There are many Scenarios where these realistic face swapped deepfakes are used to create political distress, fake terrorism events, revenge porn, blackmail peoples are easily envisioned.

It becomes very important to spot the difference between the deepfake and pristine video. We are using AI to fight AI. Deepfakes are created using tools like FaceApp and Face Swap, which using pre-trained neural networks like GAN or Auto encoders for these deepfakes creation. Multi-task cascaded convolution neural network(MTCNN) extracts the frame-level features and these features are further used to train the Long mesonet to classify the video as Deepfake or real. To emulate the real time scenarios and make the model perform better on real time data, we trained our method with large amount of balanced and combination of various available dataset like FaceForensic++.

Further to make the ready to use for the customers, we have developed a front end application where the user the user will upload the video. The video will be processed by the model and the output will be rendered back to the user with the classification of the video as deepfake or real and confidence of the model.

Literature Survey

Face Warping Artifacts [15] used the approach to detect artifacts by comparing the generated face areas and their surrounding regions with a dedicated Convolutional Neural Network model. In this work there were two-fold of Face
Artifacts.
Their method is based on the observations that current deepfake algorithm can only generate images of limited resolutions, which are then needed to be further transformed to match the faces to be replaced in the source video. Their method has not considered the temporal analysis of the frames.

Detection by Eye Blinking [16] describes a new method for detecting the deepfakes by the eye blinking as a crucial parameter leading to classification of the videos as deepfake or pristine. The Long-term Recurrent Convolution Network (LRCN) was used for temporal analysis of the cropped frames of eye blinking. As today the deepfake generation algorithms have become so powerful that lack of eye blinking can not be the only clue for detection of the deepfakes. There must be certain other parameters must be considered for the detection of deepfakes like teeth enchantment, wrinkles on faces, wrong placement of eyebrows etc.

Capsule networks to detect forged images and videos [17] uses a method that uses a capsule network to detect forged, manipulated images and videos in different scenarios, like replay attack detection and computer-generated video detection.
In their method, they have used random noise in the training phase which is not a good option. Still the model performed beneficial in their dataset but may fail on real time data due to noise in training. Our method is proposed to be trained on noiseless and real time datasets.

Recurrent Neural Network [18] (RNN) for deepfake detection used the approach of using RNN for sequential processing of the frames along with

ImageNet pre-trained model. Their process used the HOHO [19] dataset consisting of just 600 videos.
Their dataset consists small number of videos and same type of videos, which may not perform very well on the real time data. We will be training out model on large number of Realtime data.

Synthetic Portrait Videos using Biological Signals [20] approach extract biological signals from facial regions on pristine and deepfake portrait video pairs. Applied transformations to compute the spatial coherence and temporal consistency, capture the signal characteristics in feature vector and photoplethysmography (PPG) maps, and further train a probabilistic Support Vector Machine (SVM) and a Convolutional Neural Network (CNN). Then, the average of authenticity probabilities is used to classify whether the video is a deepfake or a pristine.
Fake Catcher detects fake content with high accuracy, independent of the generator, content, resolution, and quality of the video. Due to lack of discriminator leading to the loss in their findings to preserve biological signals, formulating a differentiable loss function that follows the proposed signal processing steps is not straight forward process.
A mesonet, in the context of deepfake detection, refers to a multi-modal network that combines information from multiple sources or modalities (such as visual, audio, and textual) to improve the accuracy of deepfake detection algorithms. Deepfakes are highly realistic manipulated media created using deep learning techniques, and detecting them has become a critical concern in various fields, including journalism, entertainment, and security.
According to the https://doi.org/10.48550/arXiv.2308.04177
Mesonet has more accuracy than any other models.

TABLE V: Generalizability of detectors to unseen datasets

Rank	Detector	Original Accuracy (%)	Unseen Dataset					
			Accuracy (%)					Degradation (% points)
			FS	NT	DF	DFD	Average	
1	ShallowNet6	71.00	79.50	75.60	56.20	52.10	65.85	5.15
2	ShallowNet2	71.15	75.60	70.85	53.60	51.75	62.95	8.20
3	ShallowNet3	73.20	75.60	71.40	53.95	55.60	64.14	9.06
4	ShallowNet5	76.60	78.55	77.75	57.20	55.85	67.34	9.26
5	EfficientNet6	77.15	75.05	76.15	55.30	46.15	63.16	13.99
6	MesoNet3	85.50	75.50	76.10	65.95	61.45	69.75	15.75
7	EfficientNet3	79.05	67.70	66.35	63.05	50.35	61.86	17.19
8	MesoInception3	89.05	76.25	71.05	59.50	56.95	65.94	23.11
9	EfficientNet5	76.80	51.25	54.30	56.30	49.40	52.81	23.99
10	XceptionNet3	93.60	80.75	76.35	63.90	55.10	69.03	24.58
11	XceptionNet6	94.90	83.95	83.40	58.50	53.25	69.78	25.13
12	MesoNet6	94.65	77.80	76.90	60.95	57.95	68.40	26.25
13	EfficientNet4	78.50	49.10	53.90	55.85	48.80	51.91	26.59
14	MesoInception6	94.15	82.05	78.25	56.45	52.70	67.36	26.79
15	ShallowNet1	85.40	58.05	50.20	54.90	49.60	53.19	32.21
16	ShallowNet4	88.85	53.50	53.35	58.45	55.15	55.11	33.74
17	EfficientNet2	88.15	50.30	50.40	58.25	52.60	52.89	35.26
18	EfficientNet1	88.45	51.95	49.80	58.70	51.95	53.10	35.35
19	XceptionNet5	93.35	60.00	62.55	56.70	47.25	56.63	36.73
20	MesoNet5	96.20	64.55	62.40	56.75	53.55	59.31	36.89
21	MesoInception5	96.35	57.45	60.60	60.50	55.50	58.51	37.84
22	MesoNet2	96.85	57.55	58.35	60.80	55.65	58.09	38.76
23	MesoInception2	96.50	54.65	55.25	56.85	51.60	54.59	41.91
24	MesoNet1	97.60	51.15	52.30	61.80	55.75	55.25	42.35
25	XceptionNet4	96.15	48.80	55.75	57.30	50.80	53.16	42.99
26	MesoNet4	97.35	54.55	52.20	56.00	54.35	54.28	43.08
27	MesoInception4	97.60	50.00	53.35	59.10	53.95	54.10	43.50
28	MesoInception1	97.70	52.80	52.10	57.10	51.90	53.48	44.23
29	XceptionNet2	98.45	55.25	51.15	56.80	53.45	54.16	44.29
30	XceptionNet1	99.10	54.60	50.95	57.00	54.80	54.34	44.76

Goals and objectives

- Our project aims at discovering the distorted truth of the deep fakes.
- Our project will reduce the Abuses' and misleading of the common people on the world wide web.
- Our project will distinguish and classify the video as deepfake or pristine.
- Provide a easy to use system for used to upload the video and distinguish whether the video is real or fake.

Methodologies of Problem solving

- Solution Requirement
  We analysed the problem statement and found the feasibility of the solution of the problem. We read different research paper as mentioned in 3.3. After checking the feasibility of the problem statement. The next step is the dataset gathering and analysis. We analysed the data set in different approach of training like negatively or positively trained i.e training the model with only fake or real video's but found that it may lead to addition of extra bias in the model leading to inaccurate predictions. So after doing lot of research we found that the balanced training of the algorithm is the best way to avoid the bias and variance in the algorithm and get a good accuracy.
- Solution Constraints
  We analysed the solution in terms of cost, speed of processing, requirements, level of expertise, availability of equipment's.
- Parameter Identified
  1. Blinking of eyes
  2. Teeth enchantment
  3. Bigger distance for eyes

4. Moustaches
5. Double edges, eyes, ears, nose
6. Iris segmentation
7. Wrinkles on face
8. Inconsistent head pose
9. Face angle
10. Skin tone
11. Facial Expressions
12. Lighting
13. Different Pose
14. Double chins
15. Hairstyle
16. Higher cheek bones

Design

After research and analysis, we developed the system architecture of the solution as mentioned in the Chapter 6. We decided the baseline architecture of the Model which includes the different layers and their numbers.

Evaluation

We evaluated our model with a large number of real time dataset which include YouTube videos dataset. Confusion Matrix approach is used to evaluate the accuracy of the trained model.

Outcome

The outcome of the solution is trained deepfake detection models that will help the users to check if the new video is deepfake or real.

Applications

Web based application will be used by the user to upload the video and submit the video for processing. The model will pre-process the video and predict whether the uploaded video is a deepfake or real video.

Hardware Resources Required

In this project, a computer with sufficient processing power is needed. This project requires too much processing power, due to the image and video batch processing.

- Client-side Requirements: Browser: Any Compatible browser device

Table 4.1: Hardware Requirements

Sr. No.	Parameter	Minimum Requirement
1	Intel Xeon E5 2637	3.5 GHz
2	RAM	64 GB
3	Hard Disk	100 GB
4	Graphic card	NVIDIA GeForce GTX Titan (12 GB RAM)

Software Resources Required

1. Operating System: Windows 7+
2. Programming Language : Python 3+
3. Framework: PyTorch 1.4, Tensorflow
4. Libraries : OpenCV, MTCNN, streamlit, EfficientNet, Dlib

Activity Diagram:

Training Workflow:

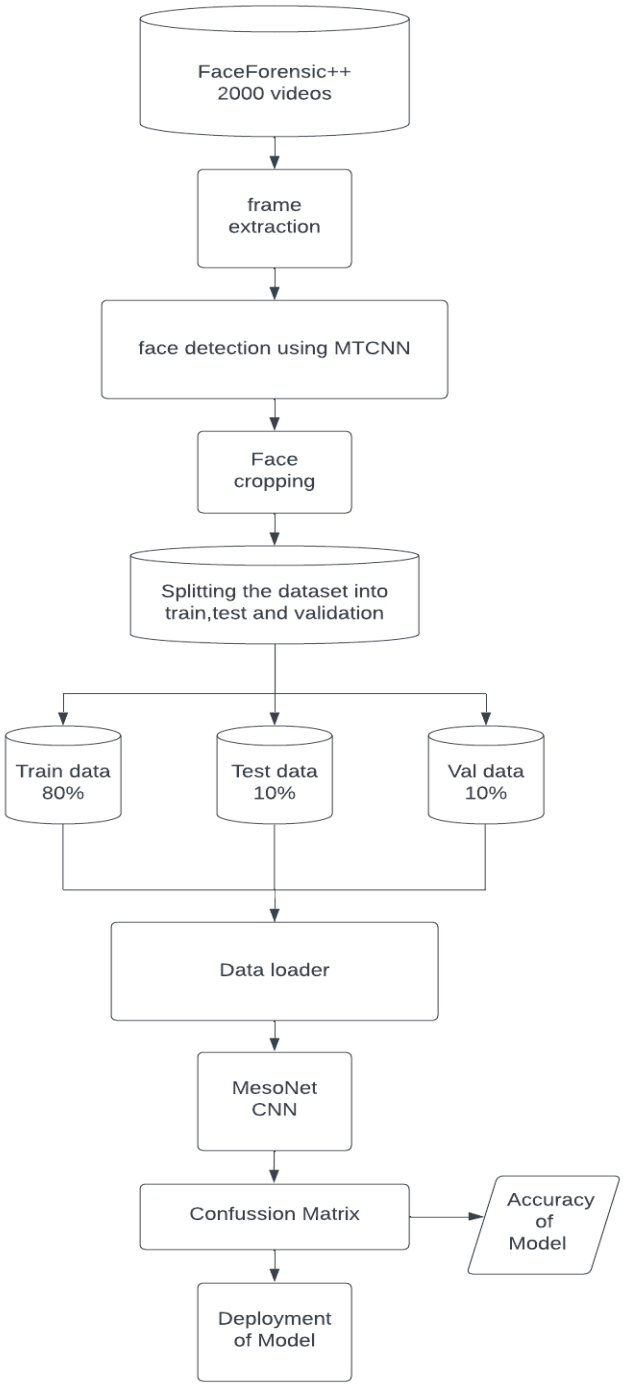


Figure 6.5: Training Workflow

Testing Workflow:

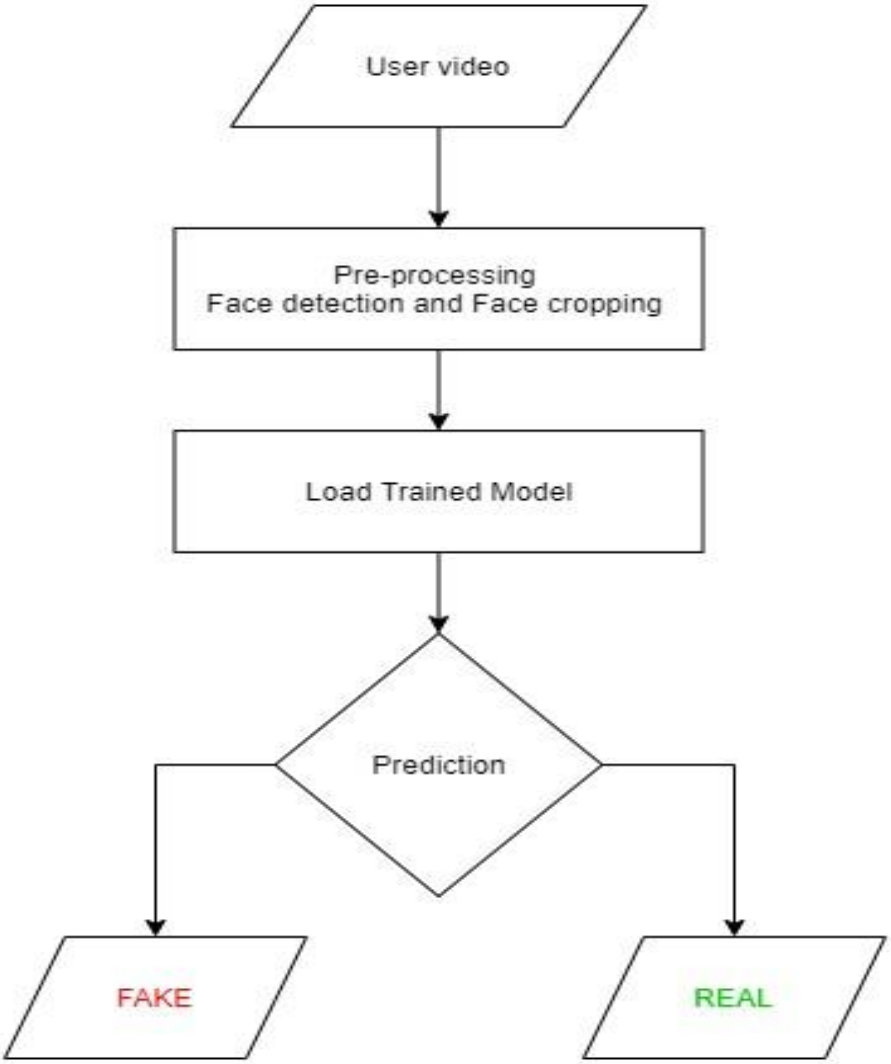


Figure 6.6: Testing Workflow

Detailed Design Document

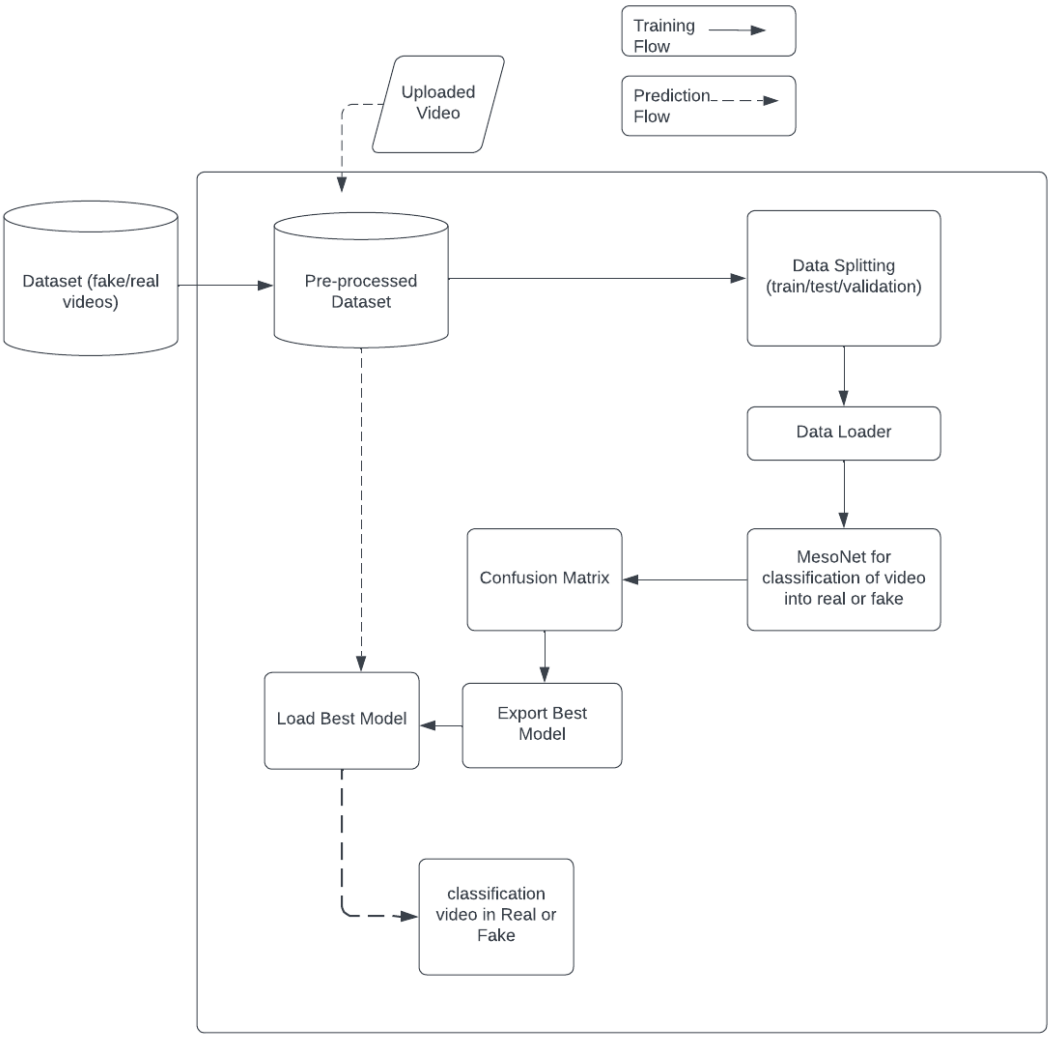


Figure 7.1: System Architecture

In this system, we have trained our deepfake detection model on equal number of real and fake videos in order to avoid the bias in the model. The system architecture of the model is showed in the figure. In the development phase, we have taken a dataset, pre-processed the dataset and created a new processed dataset which only includes the face cropped videos.

- Creating deepfake videos

To detect the deepfake videos it is very important to understand the creation process of the deepfake. Majority of the tools including the GAN and autoencoders takes a source image and target video as input. These tools split the video into frames, detect the face in the video and replace the source face with target face on each frame. Then the replaced frames are then combined using different pre-trained models. These models also enhance the quality of video my removing the left-over traces by the deepfake creation model. Which result in creation of a deepfake looks realistic in nature. We have also used the same approach to detect the deepfakes. Deepfakes created using the pretrained neural networks models are very realistic that it is almost impossible to spot the difference by the naked eyes. But in reality, the deepfakes creation tools leaves some of the traces or artifacts in the video which may not be noticeable by the naked eyes. The motive of this paper to identify these unnoticeable traces and distinguishable artifacts of these videos and classified it as deepfake or real video.

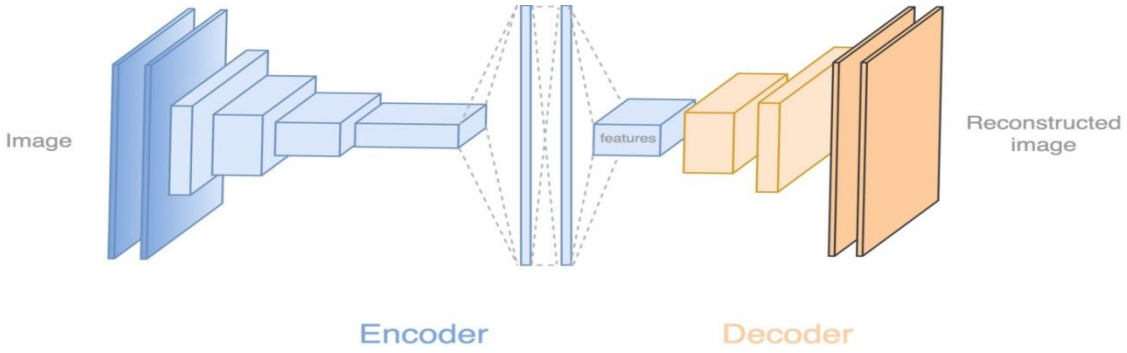


Figure 7.2: Deepfake generation

Figure 7.3: Face Swapped deepfake generation

Tools for deep fake creation.

1. Faceswap
2. Faceit
3. Deep Face Lab
4. Deepfake Capsule GAN
5. Large resolution face masked

Architectural Design

Module 1 : Data-set Gathering

For making the model efficient for real time prediction. We have gathered the data from different available data-sets like FaceForensic++(FF)[1]. Futher we have mixed the dataset the collected datasets and created our own new dataset, to accurate and real time detection on different kind of videos.
1000 Real and 1000 Fake videos from the FaceForensic++(FF) dataset.

Module 2 : Pre-processing

In this step, the videos are preprocessed and all the unrequired and noise is removed from videos. Only the required portion of the video i.e face is detected and cropped. The first steps in the preprocessing of the video is to split the video into frames. After splitting the video into frames the face is detected in each of the frame and the frame is cropped along the face. The frame that does not contain the face is ignored while preprocessing.

To maintain the uniformity of number of frames, we have selected a threshold value based on the mean of total frames count of each video. Another reason for selecting a threshold value is limited computation power. As a video of 10 second at 30 frames per second(fps) will have total 300 frames and it is computationally very difficult to process the 300 frames at a single time in the experimental environment. So, based on our Graphic Processing Unit (GPU) computational power in experimental environment we have selected 40-50 frames as the threshold value

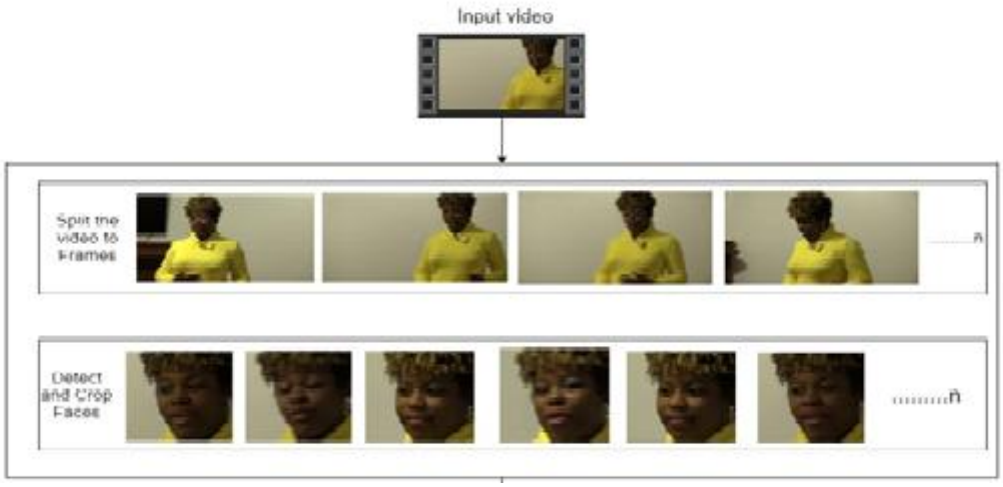


Figure 7.5: Pre-processing of video

Module 3: Data-set split

FaceForensic++ dataset contains 2000 videos. After preprocessing those videos into real and fake faces we split the face data in following manner:-

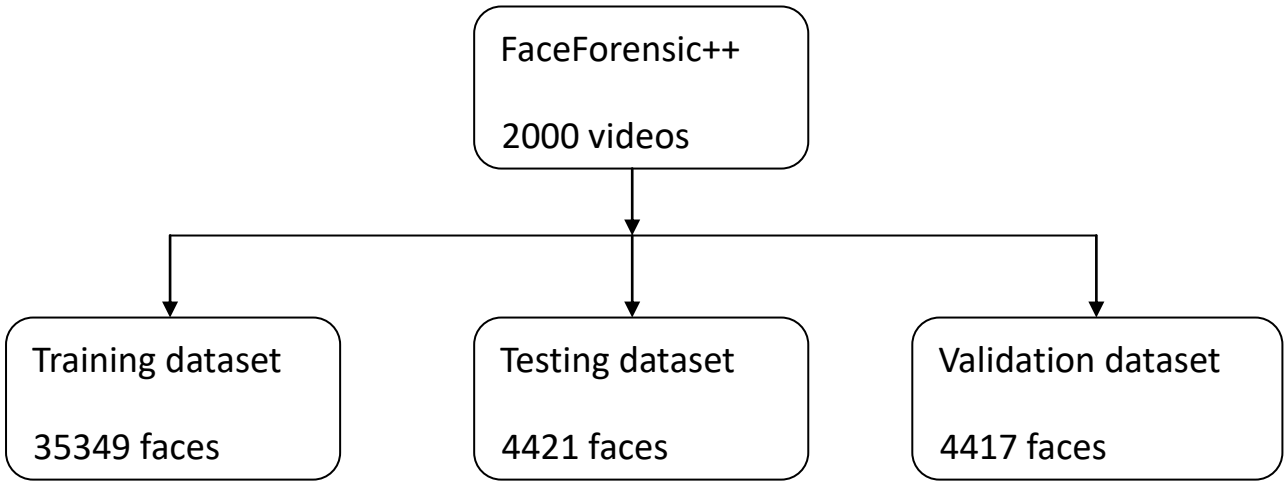


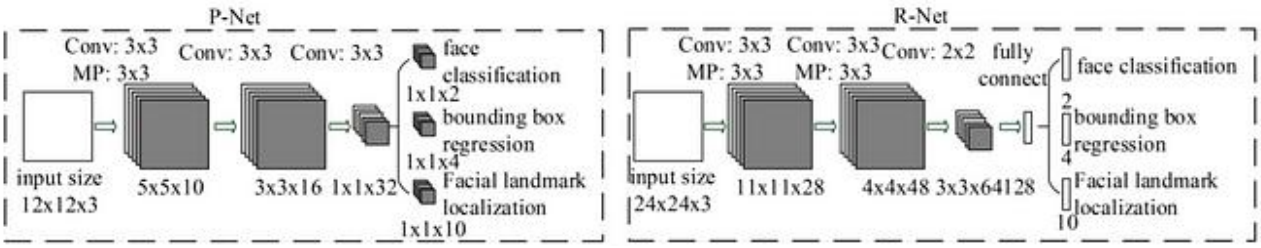
Figure 7.6: Train test split

7.2.4 Module 4: Model Architecture

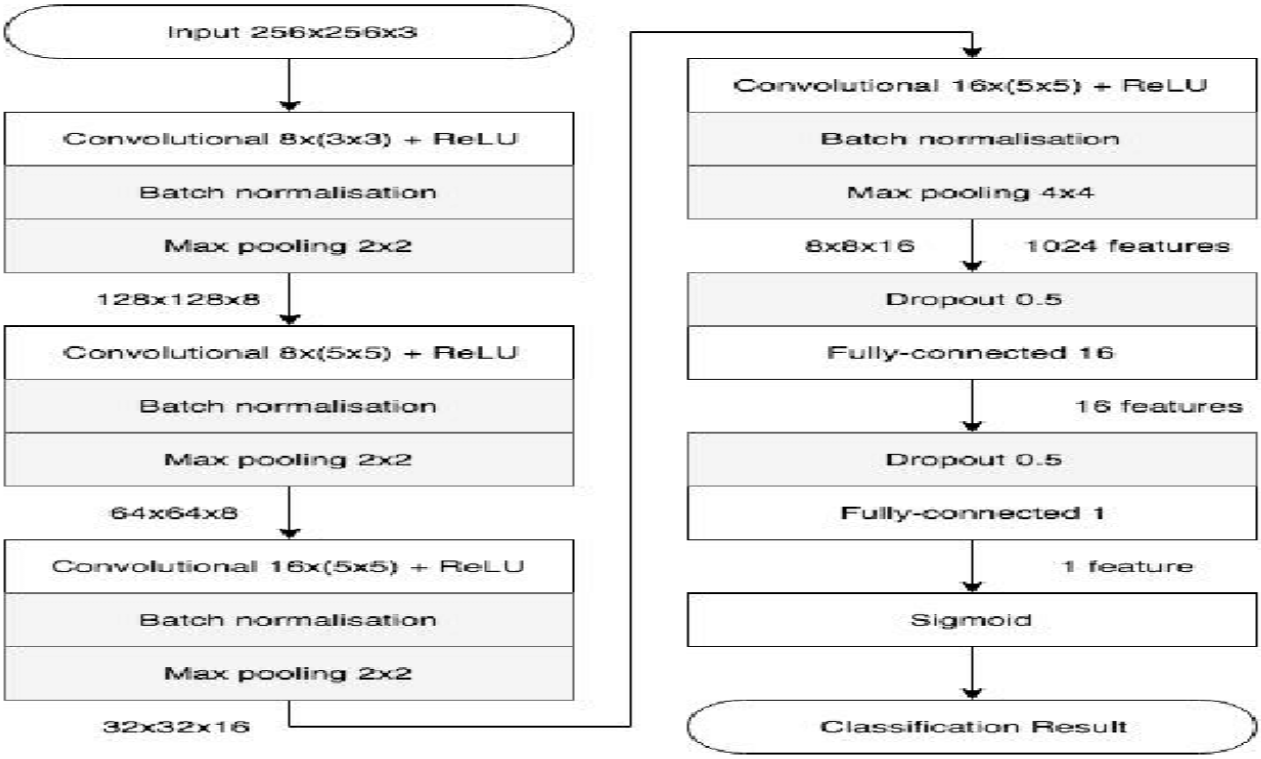
We have used the Pre- trained MTCNN model to extract the facial features at frame level and based on the extracted features a mesonet is trained to classify the video as Real or Fake.

MTCNN :

Instead of writing the code from scratch, we used the pre-trained model of MTCNN for facial feature extraction. MTCNN (Multi-task Cascaded Convolutional Networks) is a deep learning model that was originally designed for face detection and facial feature alignment in general images.



Mesonet :



MesoNet is a deep learning model that can be used to detect deepfakes. It is a relatively shallow convolutional neural network (CNN) that is trained to classify images into one of two classes: real or fake. MesoNet focuses on the mesoscopic properties of images, which are the intermediate-level features that are not captured by low-level features such as edges and textures, or high-level features such as objects and

faces. MesoNet is a promising deep learning model for deepfake detection. It is relatively simple and efficient, and it has been shown to be effective on a variety of deepfake datasets.

Figure 7.7: Overview of our model

7.2.5 Module 5: Hyper-parameter tuning

It is the process of choosing the perfect hyper-parameters for achieving the maximum accuracy. After reiterating many times on the model. The best hyper-parameters for our dataset are chosen. To enable the adaptive learning rate Adam optimizer with the model parameters is used. The learning rate is tuned to 1e-3(0.001) to achieve a better global minimum of gradient descent.

As this is a classification problem so to calculate the loss cross entropy approach is used. To use the available computation power properly the batch training is used. The batch size is taken of 64. Batch size of 64 is tested to be ideal size for training in our development environment.

The User Interface for the application is developed using Django framework. Django is used to enable the scalability of the application in the future.

The first page of the User interface i.e index.html contains a tab to browse and upload the video. The uploaded video is then passed to the model and prediction is made by the model. The model returns the output whether the video is real or fake along with the confidence of the model. The output is rendered in the predict.html on the face of the playing video.

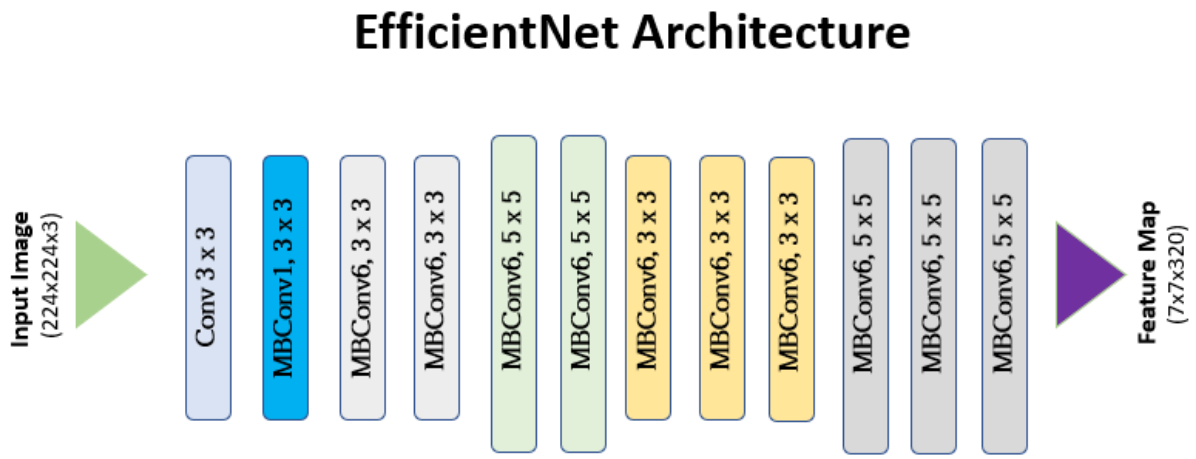
Project Implementation

8.3.2 Preprocessing Details

- Using glob we imported all the videos in the directory in a python list.
- cv2.VideoCapture is used to read the videos and get the mean number of frames in each video.
- To maintain uniformity, based on mean a value 40-50 frames are selected as ideal value for creating the new dataset.
- Then by using MTCNN on these frames face detection takes place and faces are cropped and saved in that particular video directory
- Then we used mesonet model to classify real and fake videos.
- Model evaluation is done with confusion matrix.
- Model deployment is done on streamlit.

8.3.3 Model Details

EfficientNet Architechture:-



Mesonet Architechture:-

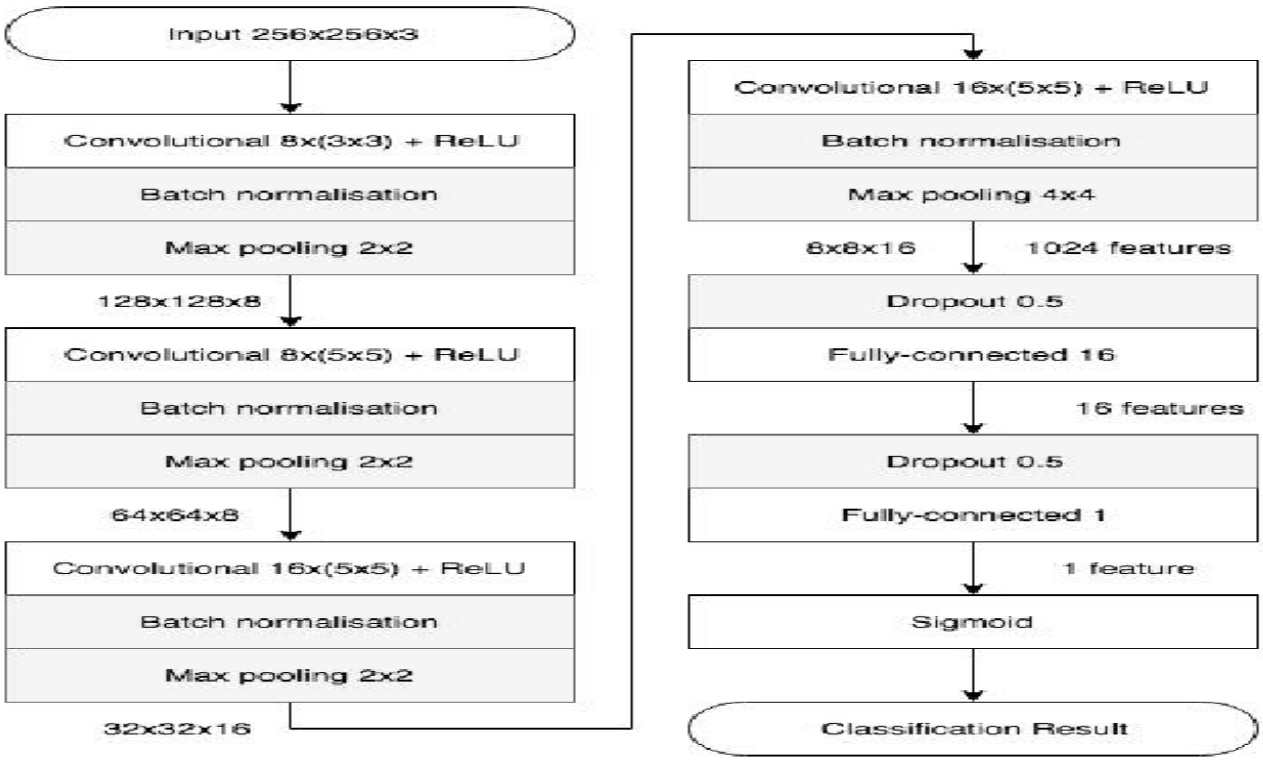
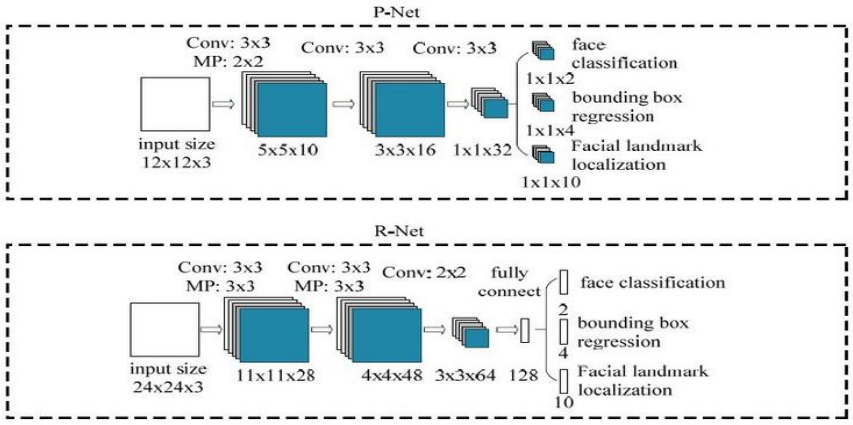


Figure 8.2: MesoNet Working

- MTCNN Architechture:-



- Sequential Layer : Sequential is a container of Modules that can be stacked together and run at the same time. Sequential layer is used to store feature vector returned by the EfficientNet model in a ordered way.

- ReLU:A Rectified Linear Unit is activation function that has output 0 if the input is less than 0, and raw output otherwise. That is, if the input is greater than 0, the output is equal to the input. The operation of ReLU is closer to

the way our biological neurons work. ReLU is non-linear and has the advantage of not having any backpropagation errors unlike the sigmoid function, also for larger Neural Networks, the speed of building models based off on ReLU is very fast.

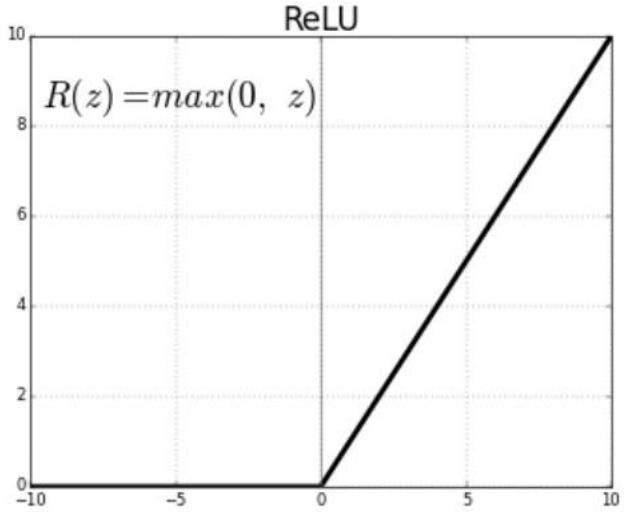


Figure 8.6: Relu Activation function

- Dropout Layer :Dropout layer with the value of 0.4 is used to avoid overfitting in the model and it can help a model generalize by randomly setting the output for a given neuron to 0. In setting the output to 0, the cost function becomes more sensitive to neighbouring neurons changing the way the weights will be updated during the process of backpropagation.

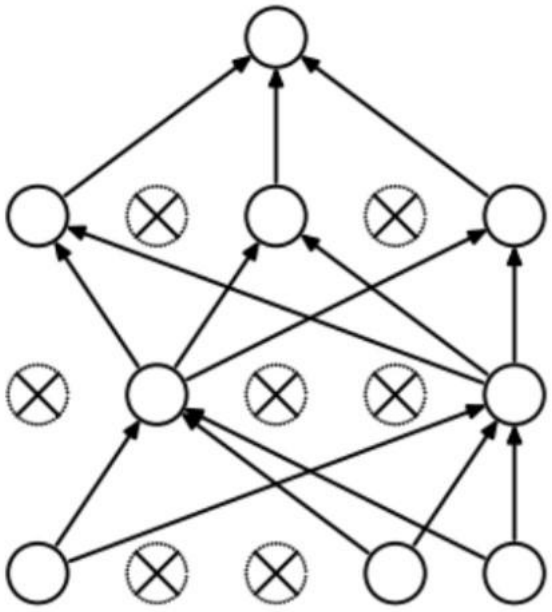


Figure 8.7: Dropout layer overview

- Adaptive Average Pooling Layer: It is used to reduce variance, reduce computation complexity and extract low level features from neighbourhood.2 dimensional Adaptive Average Pooling Layer is used in the model.

Model Training Details

- Train Test Split: The dataset is split into train and test dataset with a ratio of 80% train videos and 10% test videos,10% val data. The train and test split is a balanced split i.e 50% of the real and 50% of fake videos in each split.
- Data Loader: It is used to load the videos and their labels with a batch size of 64.
- Training: The training is done for 30 epochs with a learning rate of 1e-3 (0.001)
- Adam optimizer: To enable the adaptive learning rate Adam optimizer with the model parameters is used.
- Binary Cross Entropy: To calculate the loss function binary Cross Entropy approach is used because we are training a classification problem.
- Sigmoid Layer: A Sigmoid function is a type of squashing function. Squashing functions limit the output of the function into the range 0 to 1. This allows the output to be interpreted directly as a probability. Since the outputs of a sigmoid max function can be interpreted as a probability a sigmoid layer is typically the final layer used in neural network functions. It is important to note that a sigmoid layer must have the same number of nodes as the output later.

  In our case sigmoid layer has two output nodes i.e REAL or FAKE.

- Confusion Matrix: A confusion matrix is a summary of prediction results on a classification problem. It gives us insight not only into the errors being made by a classifier but more importantly the types of errors that are being made. Confusion matrix is used to evaluate our model and calculate the accuracy.
- Export Model: After the model is trained, we have exported the model. So that it can be used for prediction on real time data.

Model Prediction Details

- The model is loaded in the application
- The new video for prediction is preprocessed (refer 8.3.2, 7.2.2) and passed to the loaded model for prediction
- The trained model performs the prediction and return if the video is a real or fake along with the confidence of the prediction.

Results and Discussion

Screen shots

Figure 10.1: Home Page

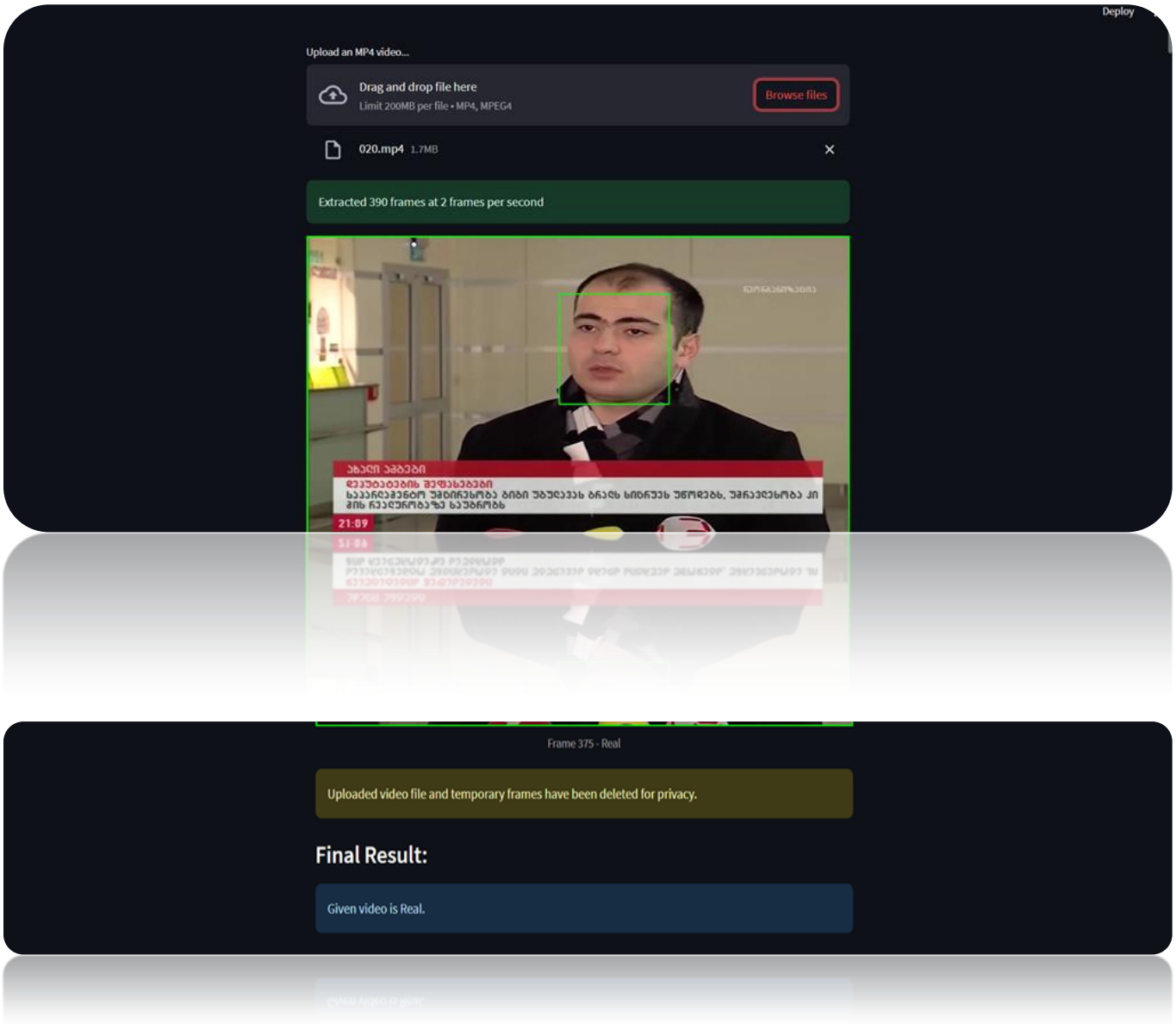


Figure 10.2: Real Video output

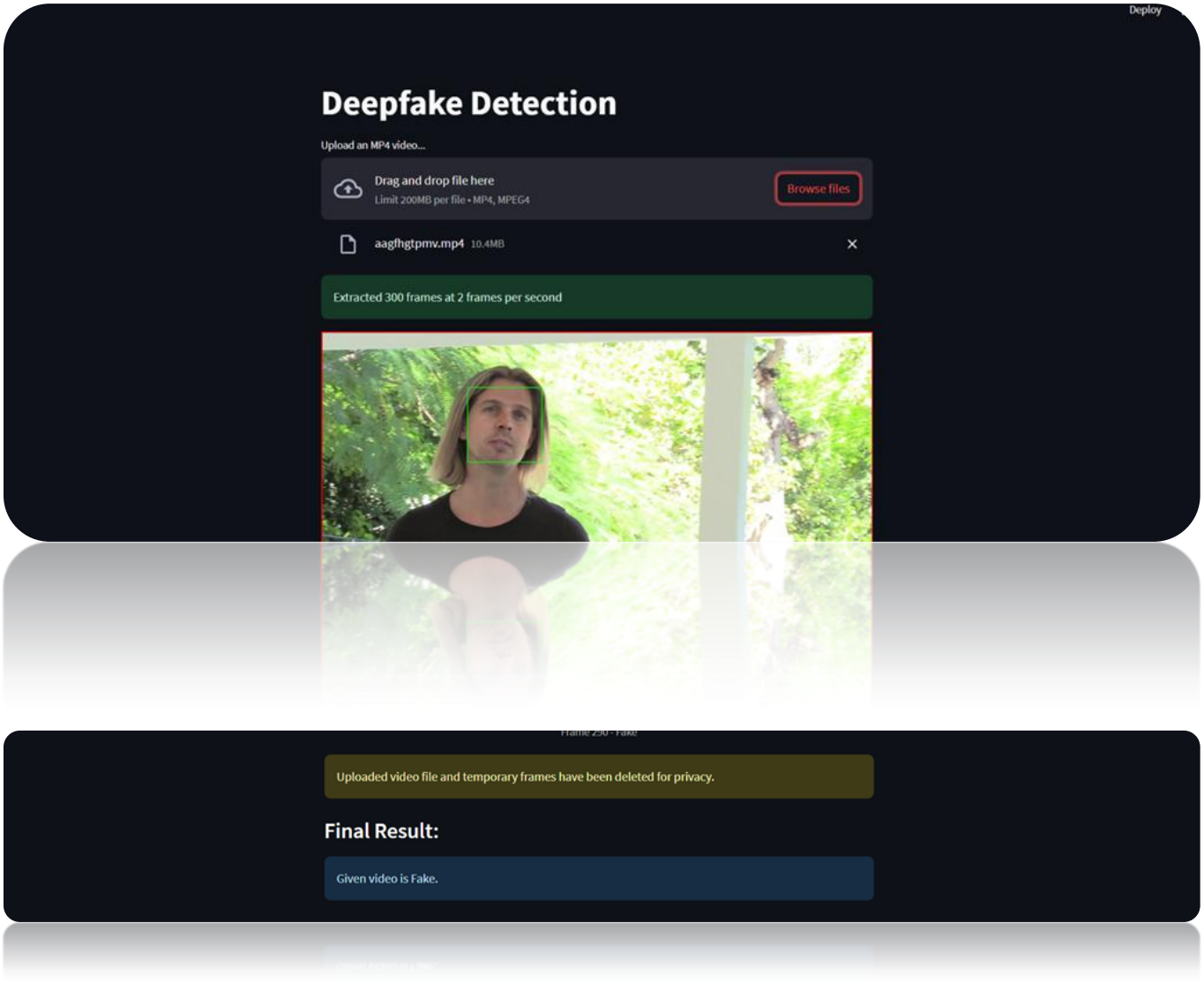
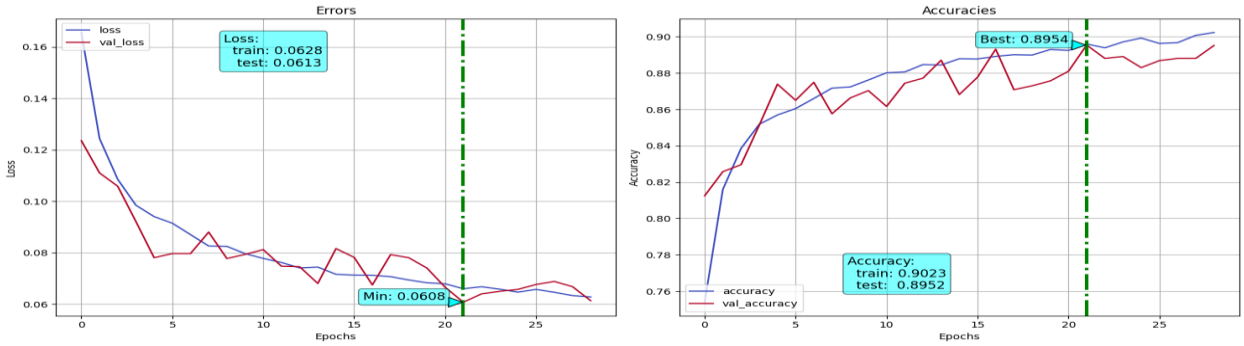


Figure 10.5: Fake video Output

Outputs

10.2.1 Model results

MESONET RESULT:-



EFFICIENTNET RESULT:-

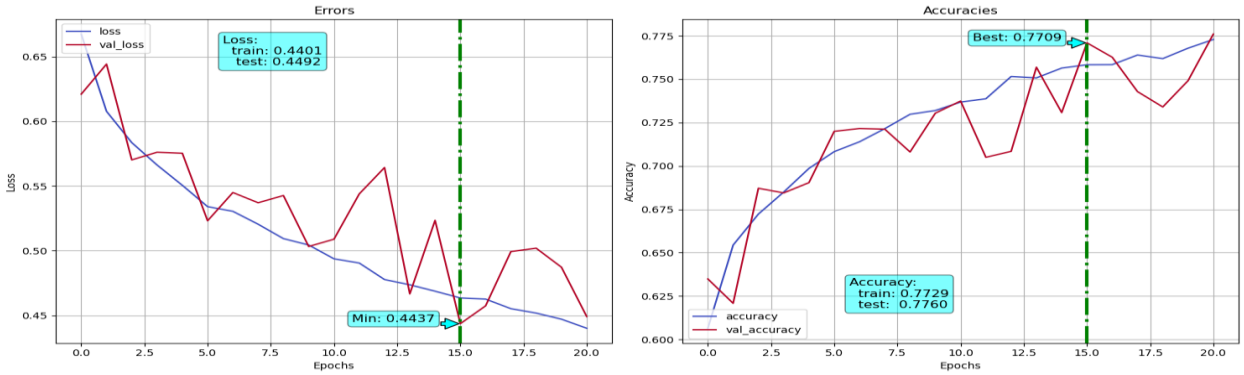


Table 10.1: Trained Model Results

Deployment

Following are the steps to be followed for the deployment of the application.

1. clone the repository using the below command.
   git clone https://github.com/abhijitjadhav1998/Deefake_detection_Django_app.git
   Note: As its a private repository only authorized users will be able see the code and do the process of deployment

2. pip install torch===1.4.0 torchvision===0.5.0 -f https://download.pytorch.org/whl/torch_stable.html
3. pip install -r requirement.txt
4. python manage.py migrate
5. Copy all the trained models into models folder.
6. python manage.py runserver 0.0.0.0:8000

Conclusion and Future Scope

12.1 Conclusion

We presented a neural network-based approach to classify the video as deep fake or real, along with the confidence of proposed model. Our method is capable of predicting the output by processing 1 second of video (10 frames per second) with a good accuracy. We implemented the model by using pre-trained MTCNN model for face detectionand mesonet for classification.

12.2 Future Scope

There is always a scope for enhancements in any developed system, especially when the project build using latest trending technology and has a good scope in future.

- Web based platform can be upscaled to a browser plugin for ease of access to the user.
- Currently only Face Deep Fakes are being detected by the algorithm, but the algorithm can be enhanced in detecting full body deep fakes.