DEEPFAKE VIDEO DETECTION

Project report submitted in partial fulfillment of the requirement for the degree of

Bachelor of Technology

in

Computer Science and Engineering

By

Navdeep Singh Hada (181405)

Under the supervision of

(Dr. Rakesh Kanji)

to

Department of Computer Science & Engineering and Information Technology
Jaypee University of Information Technology Waknaghat, Solan-173234,
Himachal Pradesh

DECLARATION

I hereby declare that, this project has been done by me under the supervision of Dr. Rakesh Kanji, Department of Computer Science & Engineering and Information Technology, Jaypee University of Information Technology. I also declare that neither this project nor any part of this project has been submitted elsewhere for award of any degree or diploma.

navdeep Singh

Supervised by:

Dr. Rakesh Kanji

Assistant Professor (SG)

Department of Computer Science & Engineering and Information Technology

Jaypee University of Information Technology

Rakesh Kanji

Submitted by:

Navdeep Singh Hada

181405

Computer Science & Engineering Department

Jaypee University of Information Technology

CERTIFICATE

I hereby declare that the work presented in this report entitled **"Deepfake Video Detection"** in partial fulfillment of the requirements for the award of the degree of **Bachelor of Technology** in **Computer Science and Engineering/Information Technology** submitted in the department of Computer Science & Engineering and Information Technology, Jaypee University of Information Technology Waknaghat is an authentic record of my own work carried out over a period from January 2022 to May 2022 under the supervision of **(Dr. Rakesh Kanji)** (Assistant Professor, Department of Computer Science & Engineering and Information Technology).

The matter embodied in the report has not been submitted for the award of any other degree or diploma.

navdeep Singh

Navdeep Singh Hada, 181405

This is to certify that the above statement made by the candidate is true to the best of my knowledge.

Rakesh Kanji

Dr. Rakesh Kanji
Assistant Professor (SG)
Computer Science & Engineering and Information Technology
Dated: 30th May 2022

AKCNOWLEDGEMENT

Firstly, I express my heartiest thanks and gratefulness to almighty God for His divine blessing makes us possible to complete the project work successfully.

I really grateful and wish my profound my indebtedness to Supervisor Dr. Rakesh Kanji, Assistant Professor, Department of CSE/IT Jaypee University of Information Technology, Wakhnaghat. Deep Knowledge & keen interest of my supervisor in the field of “Computer Vision & Pattern Recognition” to carry out this project. His endless patience, scholarly guidance, continual encouragement, constant and energetic supervision, constructive criticism, valuable advice, reading many inferior drafts and correcting them at all stage have made it possible to complete this project.

I would like to express my heartiest gratitude to Dr. Rakesh Kanji, Department of CSE, for his kind help to finish my project.

I would also generously welcome each one of those individuals who have helped me straight forwardly or in a roundabout way in making this project a win. In this unique situation, I might want to thank the various staff individuals, both educating and non-instructing, which have developed their convenient help and facilitated my undertaking.

Finally, I must acknowledge with due respect the constant support and patients of my parents.

navdeep Singh

Navdeep Singh Hada

Computer Science & Engineering

181405

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

ABSTRACT

With the advancement of artificial intelligence (AI) and cloud computing, audio, video, and image manipulation techniques have grown faster and more sophisticated. This type of media content is known as deepfakes. It is becoming increasingly possible for computers to control media in increasingly convincing ways, for instance, by having a duplicate of a public individual's voice or superimposing one person's face onto another. Media confirmation, media provenance, and deepfake discovery are the three types of countermeasures used to counteract deepfakes. Multi-modal detection techniques are used in deepfake detection solutions in order to detect whether target media has been altered or synthesized. There are two types of detection techniques in use today: manual and algorithmic. Techniques utilizing human media analysts with access to software include traditional manual techniques. AI-based algorithms are used in algorithmic detection to detect manipulated media.

We aim to build a Deep Learning Binary Classifier to help detect deepfake videos which can be applied to solve several real world problems caused by deepfakes ranging from distortion of democratic discourse; manipulation of elections; Weakening the credibility of institutions; weakening journalism; causing social divisions; sabotaging public well-being; and causing long-lasting damage to prominent people, including chose authorities and possibility for office. In this work, we present our profound convolutional neural organization put together model approved with respect to Deepfake Detection Challenge dataset for crucial AI research in deepfake discovery. Here we train the dataset on various convolution neural network models like ResNet50 + LSTM, MesoNet, DenseNet-121 and custom models. Our best model of ResNet50 + LSTM has an accuracy score of 94.63%. By implementing this project, we hope to automate the detection of deepfake videos. The model has demonstrated that it is possible to use these type of models and apply it in real world for curbing deepfake videos.

1. Chapter-1: INTRODUCTION

1.1 Introduction

Technologies for editing images, videos, and audio are advancing rapidly. There has been an increase in innovation in the areas of changing images, sound recordings, and recordings. A wide range of methods for making and controlling advanced substances is likewise available. Today, it is possible to generate hyper-reasonable advanced pictures with a little asset and a simple how-to guide available on the web. A deepfake is a technique that replaces the essence of a particular person in a video with the essence of another. Basically, a single face image gets merged with an integrated portrait. As well as addressing the last result of a promotion reasonable video, the term is also used to refer to the final product. In addition to creating extraordinary Computer Generated Imagery (CGI), Virtual Reality(VR), and Augmented Reality(A), Deepfakes can be used for educational purposes, animation, art, and cinema.

As smartphones have become more sophisticated and good internet access has become ubiquitous, an increasing number of social media and media sharing sites have boosted the ability to create and transmit digital videos. Since low-cost computing power has been growing steadily, deep learning has become more powerful than ever before. Advances of this magnitude have inevitably brought new challenges. DeepFake is a manipulation of video and audio using deep generative adversarial models. There are many instances where DF is spread over social media platforms causing spamming and providing wrong information. Such DFs are terrible and can lead to people's being threatened by, or misled by, them.

The essences of A are reproduced by an auto-encoder EA based on the dataset of facial pictures of A, and an auto-encoder EB is used to reproduce the essences of B from the dataset of facial pictures of B. In order to make Deepfake pictures, you have to assemble adjusted appearances of two changed individuals An and B, then prepare an auto-encoder EA to recreate the essences of A from the dataset of facial pictures of A and another auto-encoder EB. Shared encoding loads of EA and EB are used, while keeping the decoding parts of each encoder separate. By improving this coder, any picture with a face of A can be decoded with the decoder of EB, rather than this common encoder. This principle is outlined in Figure 1 and 2.

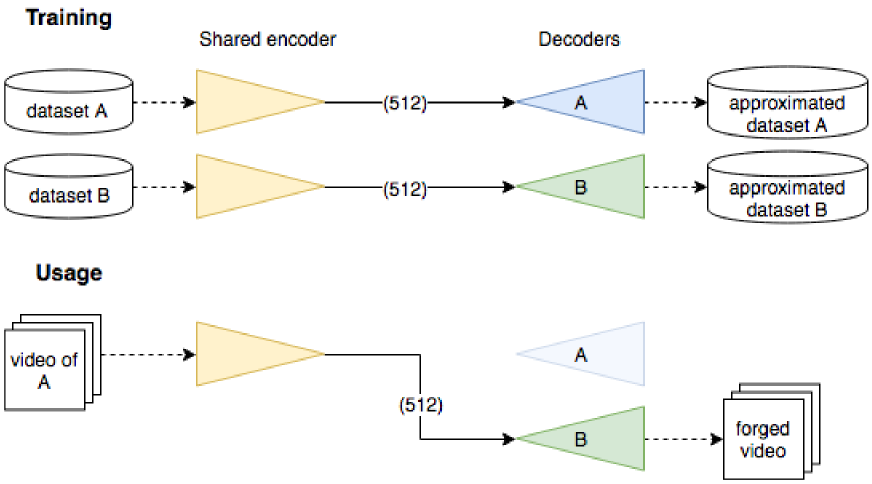


Figure 1. *Deepfake principle*.

According to this methodology, an encoder is engaged that concatenates general information of brightening, position, and appearance of the face while a devoted decoder places the detail of each face and recreates steady traits and detail. Using this method, the relevant information can be separated from the morphological information. By and by, the results are great, which explains why the procedure is so significant. The last step consists of taking the objective video, taking the objective face from each frame, adjusting it so that there is the same illumination and expression, then using the modified auto-encoder to produce yet another face, and subsequently merging it with the objective face.

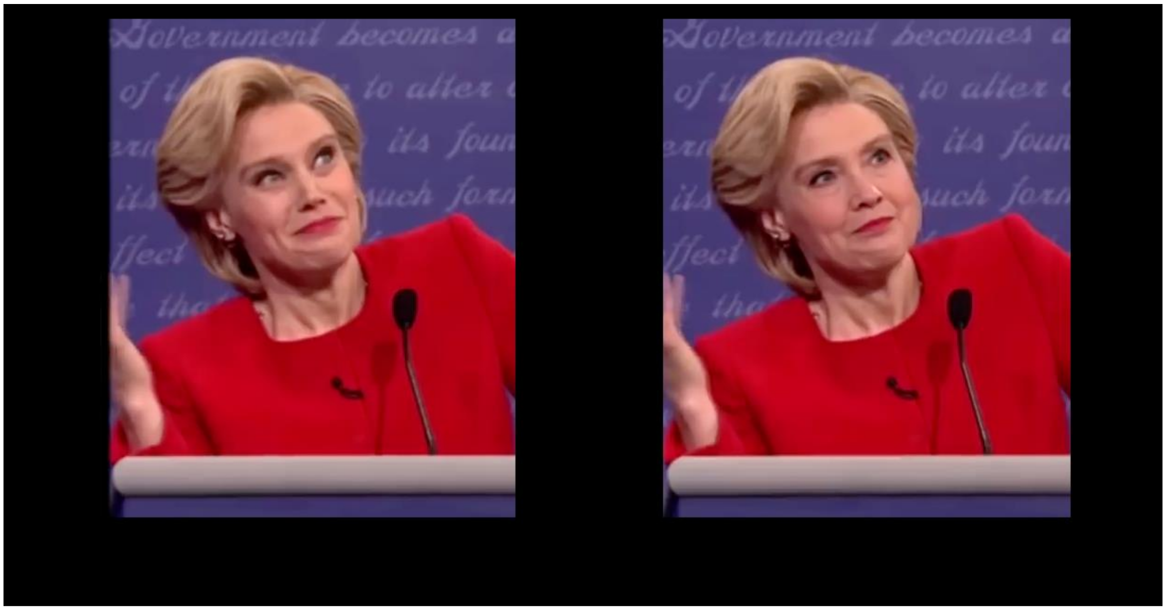
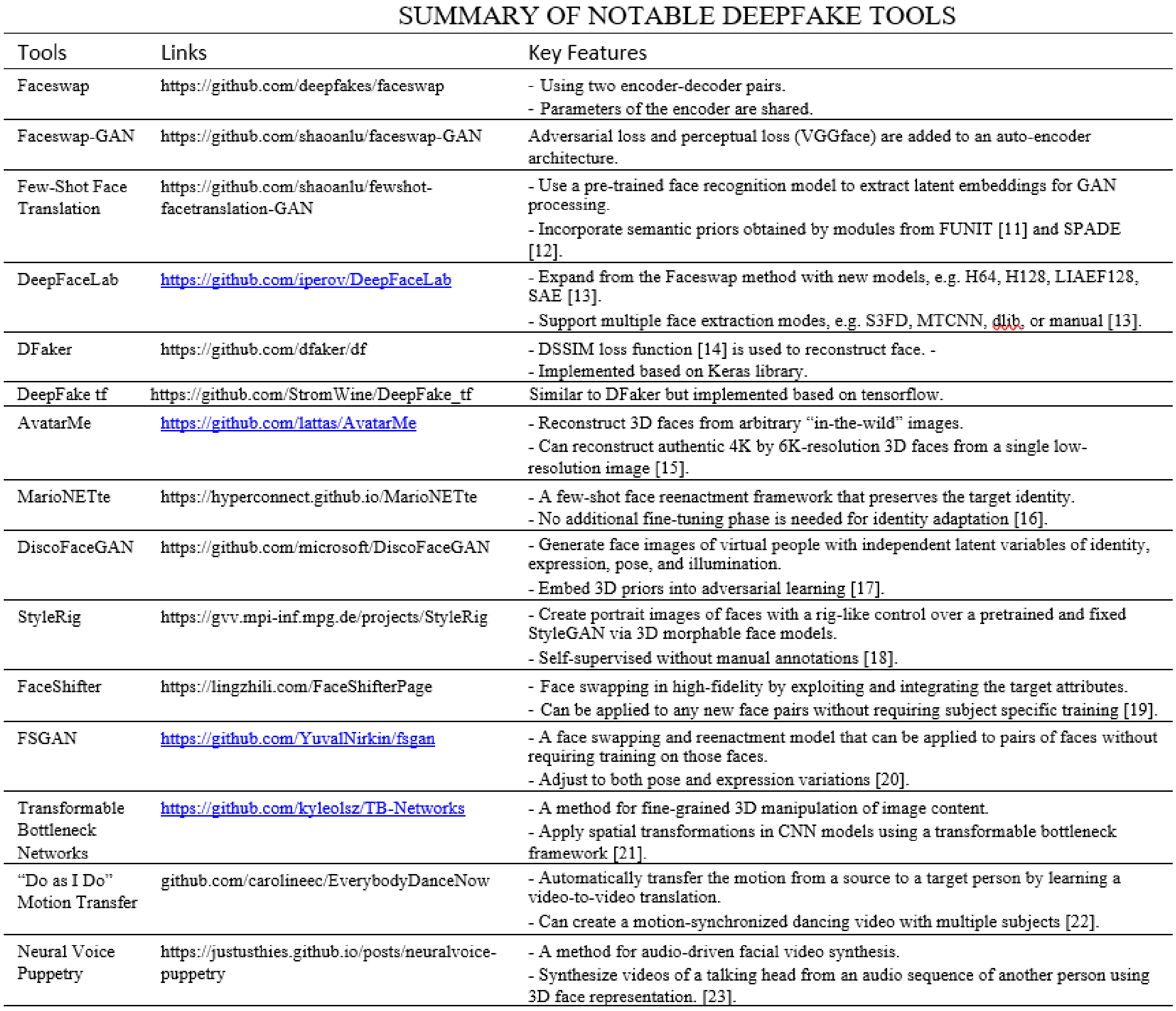


Figure 2. Illustration of a picture (left) being manufactured (right) utilizing the Deepfake procedure. Note that the manufactured face comes up short on the expressiveness of the first.

Since the Deepfake peculiarity, different creators have proposed various systems to separate genuine recordings from counterfeit ones. Despite the fact that each proposed system has its solidarity, current discovery techniques need generalizability. As pointed by [1], despite the fact that each proposed system has its solidarity, current discovery techniques need generalizability. The creators noticed that current existing models center around the Deepfake creation apparatuses to handle by concentrating on their alleged practices. For instance, Yuezun *et al*. [2] and TackHyun *et al*. [3] used inconsistencies in eye blinking to detect Deepfakes. However, using the work of Konstantinos *et al*. [4] and Hai *et al*. [5], it is now possible to mimic eye blinking. A system they presented in [4] uses natural facial expressions such as blinking eyes to create videos of talking heads. The authors in [5] proposed a model that can generate facial expression from a portrait. Their framework can blend a still picture toexpress feelings, including a mind flight of eye-flickering movements. Such progressions inDeepfake innovation make it hard to recognize the controlled recordings. To conquer such asituation, DF recognition is vital.

Thus, we portray another profound learning-based strategy that can viably recognize AI-created fakevideos (DF Videos) from genuine recordings. To provide a CNN model to accurately identify and label deepfake videos. Using semi-supervised learning, we present a methodology that is even more accurate than CNN based on the accuracy metric. A subset of DeepFake Detection Challenge data was used to evaluate the proposed ResNet50 + LSTM model. It was impossible to train on the original dataset since it had many features. So a subset of the dataset has been taken but the same data and splits as the overall dataset has been maintained.

Table 1. Some other methods for deepfake creation tool



SUMMARY OF NOTABLE DEEPFAKE TOOLS

Tools	Links	Key Features
Faceswap	https://github.com/deepfakes/faceswap	- Using two encoder-decoder pairs. - Parameters of the encoder are shared.
Faceswap-GAN	https://github.com/shaoanlu/faceswap-GAN	Adversarial loss and perceptual loss (VGGface) are added to an auto-encoder architecture.
Few-Shot Face Translation	https://github.com/shaoanlu/fewshot-facetranslation-GAN	- Use a pre-trained face recognition model to extract latent embeddings for GAN processing. - Incorporate semantic priors obtained by modules from FUNIT [11] and SPADE [12].
DeepFaceLab	https://github.com/iperov/DeepFaceLab	- Expand from the Faceswap method with new models, e.g. H64, H128, LIAEF128, SAE [13]. - Support multiple face extraction modes, e.g. S3FD, MTCNN, dlib, or manual [13].
DFaker	https://github.com/dfaker/df	- DSSIM loss function [14] is used to reconstruct face. - - Implemented based on Keras library.
DeepFake tf	https://github.com/StromWine/DeepFake_tf	Similar to DFaker but implemented based on tensorflow.
AvatarMe	https://github.com/lattas/AvatarMe	- Reconstruct 3D faces from arbitrary "in-the-wild" images. - Can reconstruct authentic 4K by 6K-resolution 3D faces from a single low-resolution image [15].
MarioNETte	https://hyperconnect.github.io/MarioNETte	- A few-shot face reenactment framework that preserves the target identity. - No additional fine-tuning phase is needed for identity adaptation [16].
DiscoFaceGAN	https://github.com/microsoft/DiscoFaceGAN	- Generate face images of virtual people with independent latent variables of identity, expression, pose, and illumination. - Embed 3D priors into adversarial learning [17].
StyleRig	https://gvv.mpi-inf.mpg.de/projects/StyleRig	- Create portrait images of faces with a rig-like control over a pretrained and fixed StyleGAN via 3D morphable face models. - Self-supervised without manual annotations [18].
FaceShifter	https://lingzhili.com/FaceShifterPage	- Face swapping in high-fidelity by exploiting and integrating the target attributes. - Can be applied to any new face pairs without requiring subject specific training [19].
FSGAN	https://github.com/YuvalNirkin/fsgan	- A face swapping and reenactment model that can be applied to pairs of faces without requiring training on those faces. - Adjust to both pose and expression variations [20].
Transformable Bottleneck Networks	https://github.com/kyleolsz/TB-Networks	- A method for fine-grained 3D manipulation of image content. - Apply spatial transformations in CNN models using a transformable bottleneck framework [21].
"Do as I Do" Motion Transfer	github.com/carolineec/EverybodyDanceNow	- Automatically transfer the motion from a source to a target person by learning a video-to-video translation. - Can create a motion-synchronized dancing video with multiple subjects [22].
Neural Voice Puppetry	https://justusthies.github.io/posts/neuralvoice-puppetry	- A method for audio-driven facial video synthesis. - Synthesize videos of a talking head from an audio sequence of another person using 3D face representation. [23].

1.2 Problem Statement

The task is to design and develop a deep learning algorithm to classify the video as deepfake or pristine. We will predict the probability that the video is fake or not with DF detection, which is usually a binary classification where the input is a video .mp4 and the output is a label L E ["REAL", "FAKE"], and we will be analyzing the input video.

An image classification problem to detect deepfakes can be thought of as a binary image classification task, with input being a video of 150 frames with a resolution of 1920x1080 pixels (or 1080x2040 if recorded vertically). The output can be described as a binary label V * [0, 1] that indicates the presence or absence of DF videos.

On every sample of the training set, the binary cross entropy loss is optimized.

$$\mathrm{B}\,L\,(\mathrm{V},\mathrm{l}) = -1 * \log p\,(\mathrm{Y} = 1|\mathrm{V}) - (1-\mathrm{l}) * \log p\,(\mathrm{Y} = 0|\mathrm{V})$$

The probability that i will be labeled by the network is given by p(Y = i|V).

1.3 Objectives

Project objectives include the development of a CNN model that accurately detects and labels deepfakes. As a result of this work, we demonstrate that the performance of CNN is outperformed by a semisupervised learning approach. Using a subset of DeepFake Detection Challenge dataset, we evaluate our ResNet50 + LSTM based model. Due to the large data set, it was not possible to train the original dataset. We have taken a subset of the dataset, but kept the same splits and data as the whole dataset.

1.4 Methodology

Many tools are available to create Deepfake(DF), but there are few tools available to detect it. We anticipate that our approach for detecting DFs will greatly contribute to the prevention of DF percolation over the world wide web. The method we have developed can detect all types of DF, including replacement DF, retrenchment DF, and interpersonal DF.

Our proposed technique (Figure 3) removes visual and fleeting elements from faces by utilizing a blend of a CNN with a RNN. Since all visual controls are situated inside face areas, and countenances are normally present in a little district of the frames, utilizing an organization that concentrates highlights from the whole frames isn't great. All things considered, we center around removing highlights just in locales where a face is available. We split the video into frames. Followed by the face discovery and editing the frames with identified face. Then, at that point, consolidating the new edited face to make the new video. Then, at that point, we use ResNet50 CNN model to separate the highlights from the frames of the video, trailed by a LSTM layer for succession handling. Then, at that point, test-time augmentation is performed and predictions are made. Our methodology is portrayed exhaustively in the accompanying subsections, including a helping and test augmnetation approach we remembered for our DFDC accommodation.

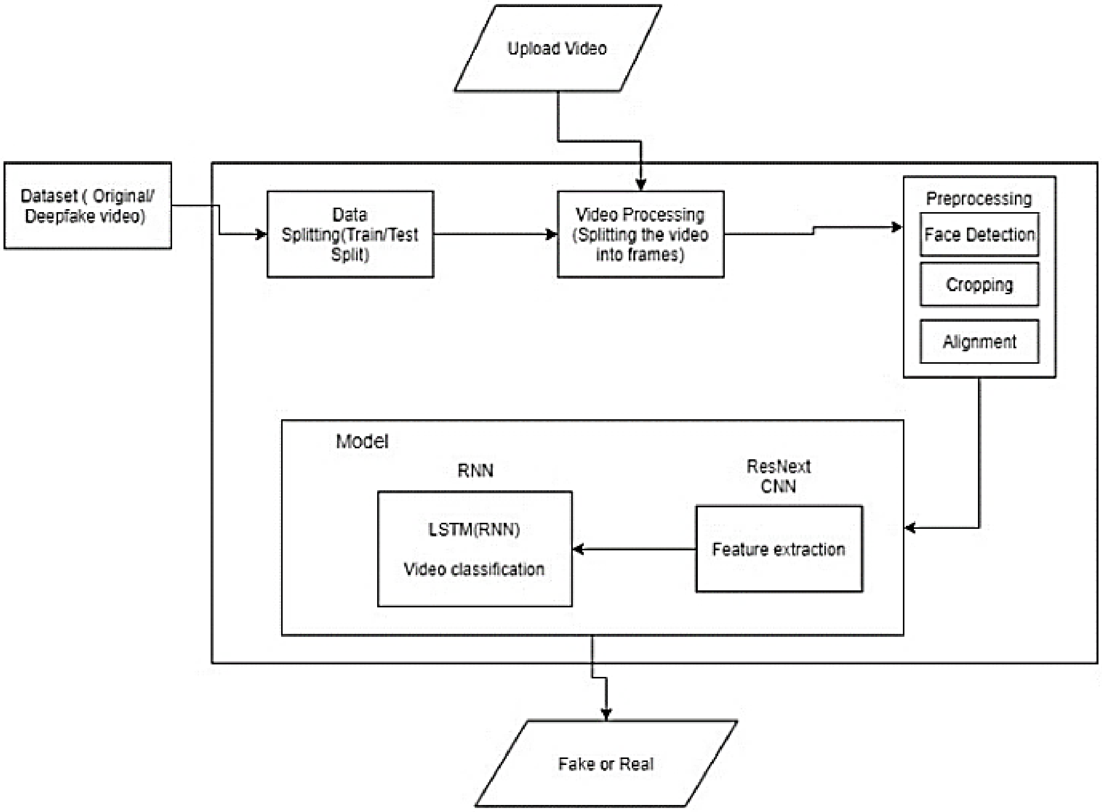


Figure 3. System Architecture

1.5 Organization

This Project is completely based on computer vision. As part of this project, the following methods, technologies and tools are used:

Methods Used:

- Data Augmentation
- Data Visualization
- Convolutional Neural Networks
- Transfer Learning
- ResNet50
- LSTM
- Test Time Augmentation

Technologies Used:

- Python, Matplotlib
- Pandas, Jupyter Notebook
- SKlearn
- Keras
- Tensorflow

Using any of these technologies is free and do not require any complex technical skills. Because of the time limitations of product development, these technologies are easier to use and faster. Therefore, the project can be implemented with ease.

2. Chapter-2: LITERATURE SURVEY

Deep fake video is becoming more and more prevalent, posing a serious threat to democracy, justice, and public trust. This has prompted an increase in the demand for video analysis, detection and intervention. Listed below are a few examples:

- By using a dedicated Convolutional Neural Network model for comparing generated faces with their surrounding regions, Exposing DF Videos by Detecting Face Warping Artifacts [6] was able to detect the artifacts. In this work, a pair of facial artifacts are featured. In their method, they observe that current implementations of the DF algorithm can only generate images with limited resolution, which will later need to be transformed further in order to match the faces in the video source.

- Uncovering AI Created Fake Videos by Detecting Eye Blinking [7] depicts another technique to uncover counterfeit face recordings produced with profound neural organization models. The technique depends on discovery of eye flickering in the recordings, which is a physiological sign that isn't top notch in the orchestrated fake recordings. This technique has been tested over benchmark datasets assessing eye-blinking techniques and demonstrates promising results when applied to recordings created using Deep Neural Network-based programming. Their technique just uses the absence of flickering as a hint for discovery. Anyway certain different parameters should be considered for recognition of the profound fake like teeth charm, wrinkles on faces and so forth Our strategy is proposed to think about this multitude of parameters.

- Capsule network is another method for detecting manipulated or forged video and image data. Through this method, manipulated and forged videos and images can be detected in various situations, like replay attacks and computer-generated videos. There has been random noise used in their method which is an undesirable practice. Although the model demonstrated positive performance in their dataset, it may not survive on real-time data due to noise in training. For training our method, we propose a noiseless and real-time dataset.

- Based on biological signals extracted from genuine and fake portrait video pairs, Detection of Synthetic Portrait Videos [9] detects and identifies fake portrait videos that contain biological signals. To train a probabilistic SVM and a CNN, change the highlight sets and PPG guides to catch the sign attributes, and ensure spatial soundness and transient consistency. Having determined the likelihood of the video being fake or credible, then a new verification is performed. This program determines whether a product contains counterfeit components with high accuracy, without regard to the generator, the content, the goal, or the nature of the video. It is not convenient to formulate a differentiable loss function based on the proposed signal processing steps when there is a lack of discriminator, resulting in the loss in their findings.

3. Chapter-3: SYSTEM DEVELOPMENT

Security, government majority rule, and protection are progressively threatened by deepfakes. Several convincing applications for face trading exist in video compositing, representation changes, as well as character verification as it allows replaced faces in photos with those chosen from a variety of stock photos. But on the other hand, it is additionally a tactic that digital attackers use to obtain ill-conceived access by infiltration of distinguishing verification frameworks. Forensic modeling can get more challenging using profound learning programs like CNN and GAN because the remaining photos can preserve the pose, facial expression, and lighting of the originals.

GAN images are probably the most difficult to detect among deep learning-generated images since they are highly realistic and high quality thanks to GAN's ability to automatically learn input distributions and generate outputs that have the same distribution as the inputs.

3.1 ***Design:***

The threat of deepfakes has prompted the development of strategies for identifying them. Initially, video synthesis varied in look and complexity to match antique and mechanical elements from antiques. To recognize deepfakes, continuing techniques also applied innovatively-developed concepts and strategies to eliminate distinguishable and striking highlights. Location deepfakes are normally classified as binary classification problems, where classification logic is applied to distinguish legitimate recordings from altered ones.

There are three types of deepfake detection methods. Identifying physical and psychological behaviors in videos is the goal of methodologies in the first category. The goal is to track head movements and eye blinking. We will discuss GAN finger impressions and organic signs found on pictures, such as blood streams that are recognizable in pictures. In the third category, we consider visual artifacts. For training, visual artifact-focused methods depend on large amounts of data. It is in this third category that we propose to place the presented model. The degradation of frame data after video compression prevents many image detection methods from being used for videos. Aside from that, recordings have worldly attributes that do not adapt among one set of frames, so it is possible to construct strategies that can distinguish just still photographs from fake ones.

3.2 ***Algorithm:***

- Start
- Upload Deepfake Detection Challenge Image dataset
- Importing all necessary Libraries
- Data Preprocessing
- Build and Train a ResNet50 + LSTM Model
- ResNet50 extracts features
- LSTM for sequence processing
- Perform Test Time Augmentation
- Predict on Test Dataset
- End

3.3 ***Model Development:***

Dataset: -

Learning from data is at the core of Deep Learning. To achieve good learning quality and accurate predictions, careful dataset preparation is vital. MTCN, YouTube, and Deep fake detection challenge datasets are being used in equal quantities for our mixed dataset. The nature of these recordings commonly incorporates standing or sitting individuals, either confronting the camera or not, with a wide scope of foundations, light conditions, and video quality. The preparation recordings have a goal of 1920 × 1080 pixels, or 1080 × 1920 pixels whenever recorded in vertical mode. This dataset is made by an all-out of 119,146 recordings with an exceptional name (genuine or counterfeit) in a preparation set, 400 recordings on the validation set without names and 4000 private recordings in a testing set. The 4000 recordings of the test set cannot be examined yet models can be assessed on it through the Kaggle framework. The proportion of manipulated: real recordings is 1:0.28. Since just the 119,245 preparing recordings contain names, we utilize the entirety of that dataset to prepare and approve our strategy. The gave preparing recordings are isolated into 50 numbered sections. Our preparation process uses 30 sections, our validating process uses 10 sections, and our testing process uses 10 sections.

The private set utilized for testing assesses submitted strategies inside the Kaggle framework and reports a log-likelihood loss. Log-likelihood loss radically punishes being both sure and wrong. In the most pessimistic scenario, an expectation that a video is bona fide when it is really controlled, or the opposite way around, will add boundlessness to your An interesting name is allotted to every video determining whether or not it contains a control. Notwithstanding, it isn't indicated which sort of control is performed: face, sound, or both. As our technique just uses video data, controlled recordings with just sound controls will prompt boisterous marks as the video will be named as fake however faces will be genuine. Moreover, more than one individual may be available in the video, with face controls performed on just one of them. blunder score. Practically speaking, assuming this most pessimistic scenario occurs, the loss is cut to an extremely huge worth. This assessment framework represents an additional a test, as techniques with great execution in measurements like exactness, could have extremely high log-likelihood blunders. Our recently pre-arranged dataset contains half of the first video and half of the controlled deepfake recordings. The dataset is parted into 80% train and 20% test set.

Classification Setup: -

In the following model, X refers to the input set, Y refers to the output set, and f is the prediction function of the classifier that takes values in X as input to the action set A, the random variable pair (X, Y) that is assumed to take values in X * Y. The chosen classification task is to minimize the error E(f) = E[l(f(X), Y)], with l(a, y) = 1/2 (a − y) ^2 .

PyTorch and Keras 2.1.5 have been used to implement both networks using Python 3.5. The network's weights are optimized using successive batches of frames of size 224 × 224 × 3 using the default parameters for ADAM (β1 = 0.9 and β2 = 0.999).

Preprocessing: -

The video is divided into frames as part of the pre-processing. Then, face recognition is performed, followed by trimming of the frames that contain the detected faces. We determine the mean of the dataset video and make a new handled face trimmed dataset containing the frames equivalent to the mean in order to maintain consistency in the quantity of frames. In pre-processing, frames without faces are ignored. Since all visual controls are situated inside face areas, and countenances are ordinarily present in a little locale of the casing, utilizing an organization that concentrates highlights from the whole frames isn't great. All things being equal, we centre around extricating highlights just in districts where a face is available. As handling the 10 second video at 30 frames each second for example all out 300 frames will require a ton of computational power. So we propose to use only 150 frames for training the model for experimental purposes.

Model: -

In deepfake videos, there are inconsistencies between frames as well as within frames. LSTM and CNN are used to identify deepfake recordings using the temporal-aware pipeline technique. CNN is utilized to eliminate frame-level elements, which are subsequently incorporated into LSTM to make a succession descriptor. The model comprises of ResNet50 followed by one LSTM layer. A fully connected network is at last utilized for grouping doctored recordings from genuine ones dependent on the succession descriptor. The Data Loader stacks the pre-handled face trimmed recordings and split the recordings into train and test set. Further the frames from the handled recordings are passed to the model for preparing and testing in mini batches. The recognition network comprising of fully connected layers is utilized to accept the grouping descriptor as information and work out probabilities of the frames arrangement having a place with either bona fide or deepfake class.

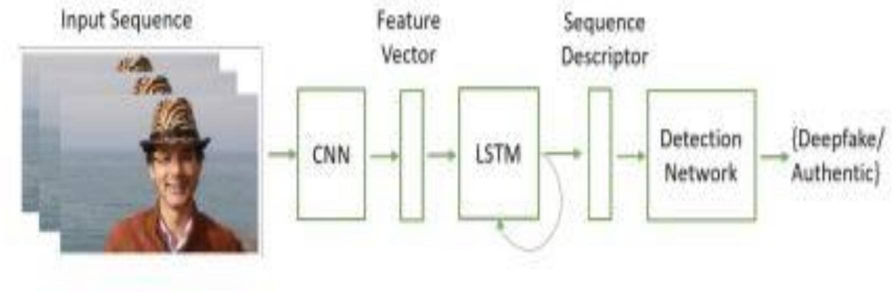


Figure 4. Input Sequence through model

ResNet CNN for Feature Extraction: -

By using the ResNet CNN classifier, we are proposing to efficiently extract the features and create an accurate frame level classifier instead of rewrite it. To properly converge the gradient descent of the model, we will add extra layers and choose a proper learning rate to fine-tune the network. After the last pooling layer, the 2048-dimensional feature vectors are used as the sequential LSTM input.

LSTM for Sequence Processing: -

Data from all tested casings should be used to perform the video-level forecast. In addition to the programmed face weighting, we incorporate a Recurrent Neural Network (RNN) underneath to combine the highlights of all face districts and frames. In order to get an accurate measurement, we combine the features, logits, and all face districts with the Long Short Term Memory (LSTM). We expect a progression of ResNet CNN data frames graphs as information and a two-hub neural organization, with chances for the combo to be imperative for an extensive fake video or an untampered video. Developing a model to recursively process sequences in a meaningful manner is the key challenge to be addressed. As a solution, we propose the use of a 2048 LSTM unit with 0.4 dropout probability, in order to accomplish our objectives.

By comparing the frame at 't' second with the frame at 't-n' second, the time of the video can be analyzed by using LSTMs. In other words, n is the number of frames before t. The LSTM consists of three stacked bidirectional layers and one unidirectional layer with aspect 2048. To determine the likelihood of the video being manipulated, the direct layer and the Sigmoid function are applied to the result of the LSTM.

Training Process: -

Training, validation, and testing are essential components of Deepfake detection. Training forms the core of the proposed model. This is where learning takes place. For DL models to fit specific domains of problems, designs and fine-tuning are necessary. We must find parameters that are optimal for training our dataset. The training and validation components are also similar. During the validation process, we fine-tune our model. The validation component tracks progress in training and accuracy in detecting DeepFakes. A specific video is classified and determined by the testing component by determining the class of the faces extracted. The testing component contributes to the research objectives.

Feature Learning (FL) is one component of the proposed model, while classification is the other. FL extracts learnable features from face images by analyzing them. As input, the FL is converted into a sequence of pixels for the final detection process through the Classification process. A feature learning (FL) method involves convolutional operations that are stacked on top of each other. A ResNet50-inspired architecture underlies the feature learning component. As opposed to the ResNet50 architecture, the FL component does not have the fully connected layer, and its purpose is not to classify faces but rather to extract features from face images for the Classification component. As a result, with a FL component, you do not have the fully connected layer of a CNN.

The ResNet50 is initialized with pre-trained weights. We introduce the LSTM and Fully Connected layers with an arbitrary weight system. The network is prepared start to finish with the parallel cross-entropy loss (BCE) work with the LSTM expectation. The BCE loss is processed with trimmed countenances from casings of a haphazardly chosen video. Note that this loss depends on the result probabilities of recordings being controlled (video level forecast). The BCE applied to refreshes the loads. The BCE applied to refreshes all weights of the outfit (barring ResNet50).

$$LogLoss = -\frac{1}{n}\sum_{i=1}^{n}[y_i log(\hat{y}_i) + log(1 - y_i)log(1 - \hat{y}_i)]$$

While we train the total group start to finish, we start the training process with a discretionary introductory advance comprising of 2000 batches of arbitrary harvests to get an underlying arrangement of parameters of the model. While this didn't present any expansion in discovery precision during our trials, it gave a quicker union and a more steady preparing process.

Because of computational limitations of GPUs, the size of the network, and the quantity of info frames, just a single video can be handled at an at once. Be that as it may, the network parameters are refreshed in the wake of handling each 64 recordings (for the binary cross-entropy loss). With a learning rate of 0.001, Adam is used as the optimization technique. Our method for reducing calculation costs uses Relu (activation function) and LeakyRelu.

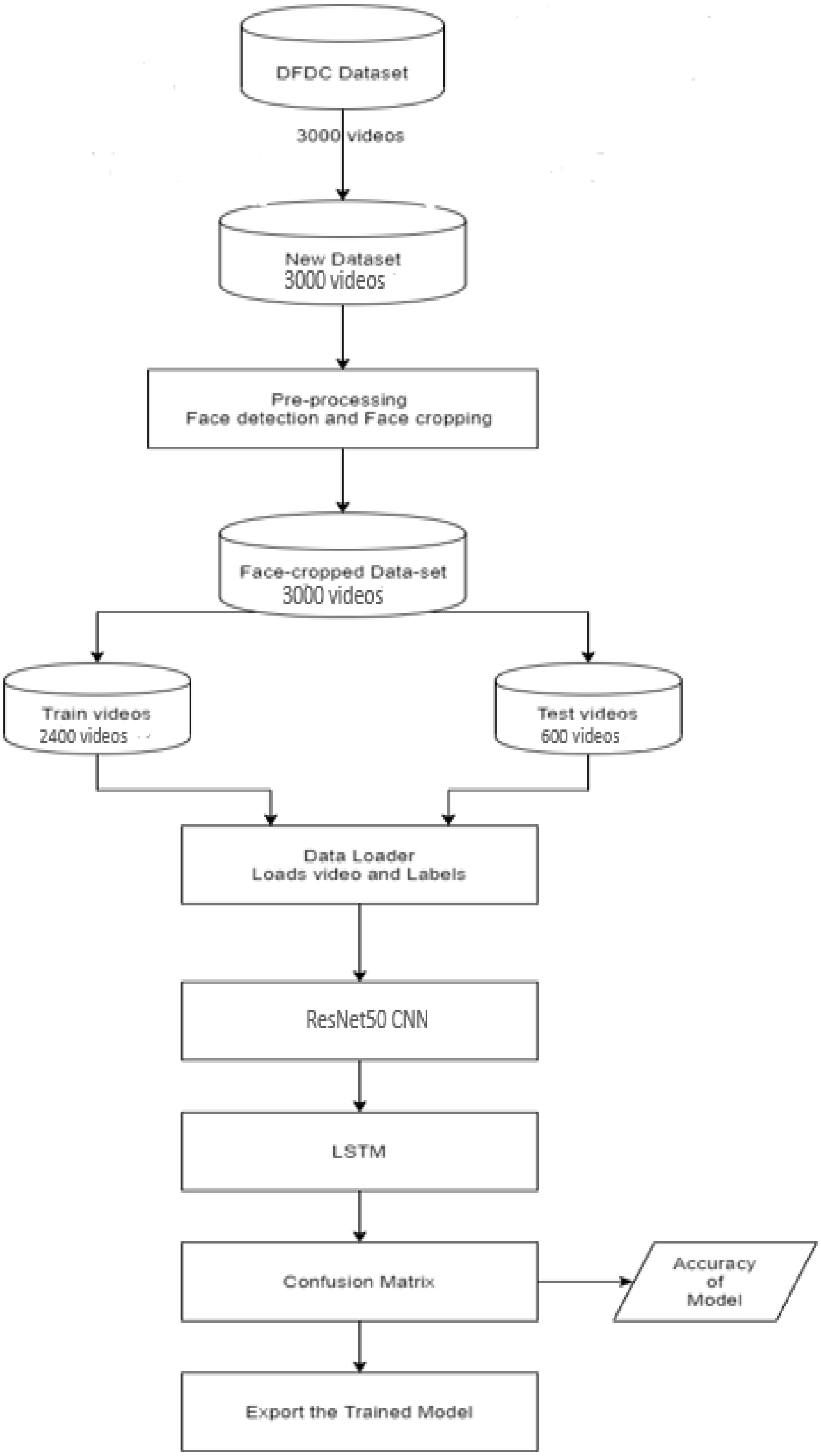


Figure 5. Data Flow Diagram

Evaluation: -

The model is arranged using the twofold cross-entropy loss function. A min-batch of 32 pictures are normalized using mean of [0.485, 0.456, 0.406] and standard deviation of [0.229, 0.224, 0.225]. The standardized face pictures are then increased prior to being taken care of into the model at each preparation cycles. Adam analyzer with a learning pace of 0.1e-3 and weight rot of 0.1e-6 is utilized for enhancement. The model is prepared for an aggregate of 20 epochs. The classification procedure takes in 30 facial pictures and passes it to our prepared model. To decide the characterization exactness of our model, we utilized a log loss function. A log loss depicted in Equation 1 groups the organization into a likelihood circulation from 0 to 1, where $0 > y < 0.5$ addresses the genuine class, and $0.5 \geq y < 1$ addresses the fake class. We picked a log loss classification metric since it profoundly punishes irregular suppositions and confident false forecasts.

4. Chapter-4: PERFORMANCE ANALYSIS

4.1 Analysis

As a result of deepfakes, you can open up additional opportunities in computerized media, virtual reality, mechanics, education, and numerous other fields. In another context, they represent innovations that can ruin and undermine the whole society. With this in mind, we developed a model that combines CNNs and LSTMs for the DF video identification task.

LSTMs can deal with sequences of consecutive frames, whereas CNNs are good at learning local highlights. Our model leverages this combined limit to associate each pixel in a picture and comprehend nonlocal highlights. When preparing and grouping, we gave equal emphasis to the preprocessing of the information.

The networks have been analyzed to find out how they approach classification. We can do this by using the weights of the diverse convolutional kernels and neurons as descriptors of pictures. Inferences can be deciphered as discrete second requests, for example, by using a positive weight, a negative one, and a positive weight once again. Although this is just an indication of the main layer, it doesn't convey much during appearances. To find out what sort of sign is being received by a particular channel [6] another way is to generate an information picture amplifying the initiation of that channel. As shown in the below Figure, the last secret layer of ResNet50 has been actuated to such a great extent for a very long time. Based on the weight assigned to their result for the last arrangement choice, depending on whether their actuation pushes toward a negative or positive score, we can isolate those neurons that influence either the genuine or produced class. In significant contrast, positive-weighted neurons induction exhibited photos with exceptionally clear eye, nose, and mouth areas as opposed to negative-weighted neurons, which contained “differences” on the establishment portion, leaving the face area “smooth.” As Deepfake-made faces are typically blurry, or else somewhat opaque, if not for the emphasis on nuances, are displayed differently to the remainder of the photograph that is left unchanged. The underlying result of a layer can at the same time be viewed as a mean aftereffect for gatherings of certified and fabricated images, and the distinctions among the incitements can be seen as well and perceptibly interpreted as the information photographs that are significant to the course of action. Since the establishment displays the most important apexes on authentic pictures anyway, the eyes are clearly started on authentic pictures. It is apparent to us that it is again a question of fogginess: In genuine pictures, however, the eye is the most definite element while in synthetic pictures, the eye is the primary piece due to the aspect reduction experienced by the face.

4.2 Results

Using the combination of Conv+LSTMs and test-time augmentation, we apply Transfer Learning to pre-trained models such as ResNet50, MesoNet and DenseNet121. The DFDC dataset helps us prepare and evaluate our strategy. Our comparison shows that our approach is relatively superior to the other three approaches. When these networks are considered, we can simply normalize the expectations for each frame to calculate a forecast on video level. With the guidance of the validation set, the arrangement that yields the best adjusted precision is determined. Based on balanced accuracy, Table 1 shows the results. If we process only the face regions, the accuracy increases dramatically. Almost all pre-trained models were overfit after six to eight epochs, as illustrated in Figure 3, the loss of the train sets and validation sets for the ResNet+LSTM model is shown. The validation loss begins to increase after the 5th epoch, which indicates overfitting. Testing was enhanced using test-time augmentation (TTA). With TTA, you can perform data augmentation on a test image to get several versions of it and average predictions. Our evaluation of TTA used a different set of transformations than that of the ResNet model. ResNet50 + LSTM is the model that achieves the highest accuracy, 94.63%.

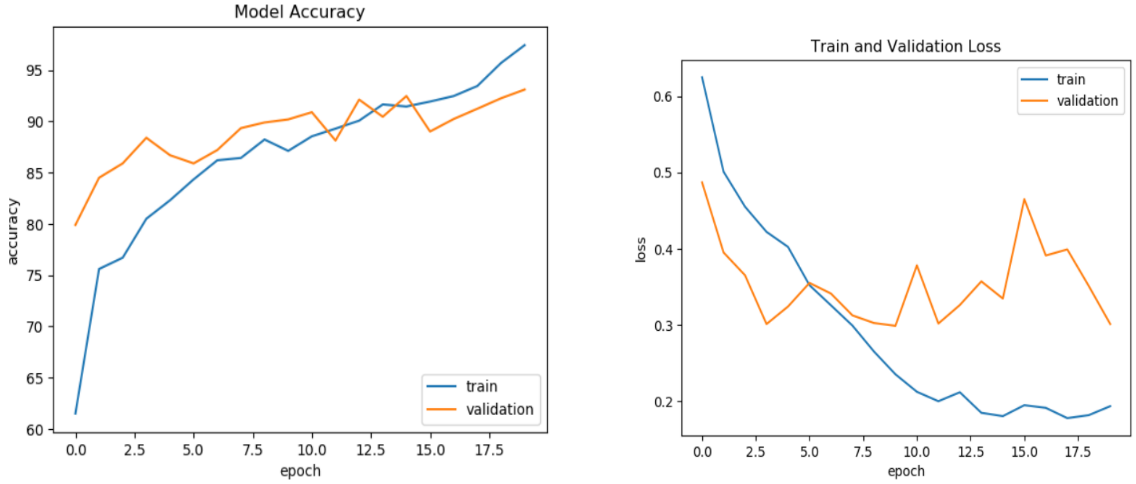
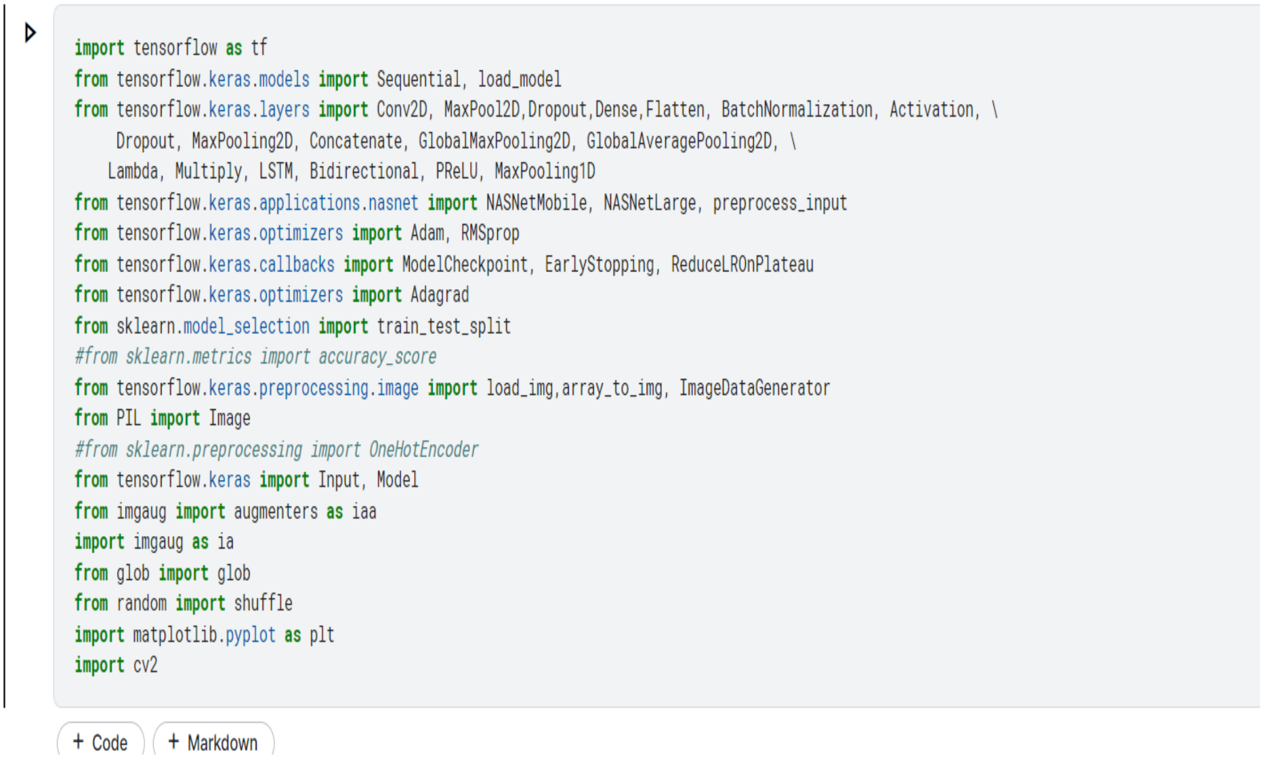


Figure 6. Train and Validation Accuracy **Figure 7.** Train and Validation Loss

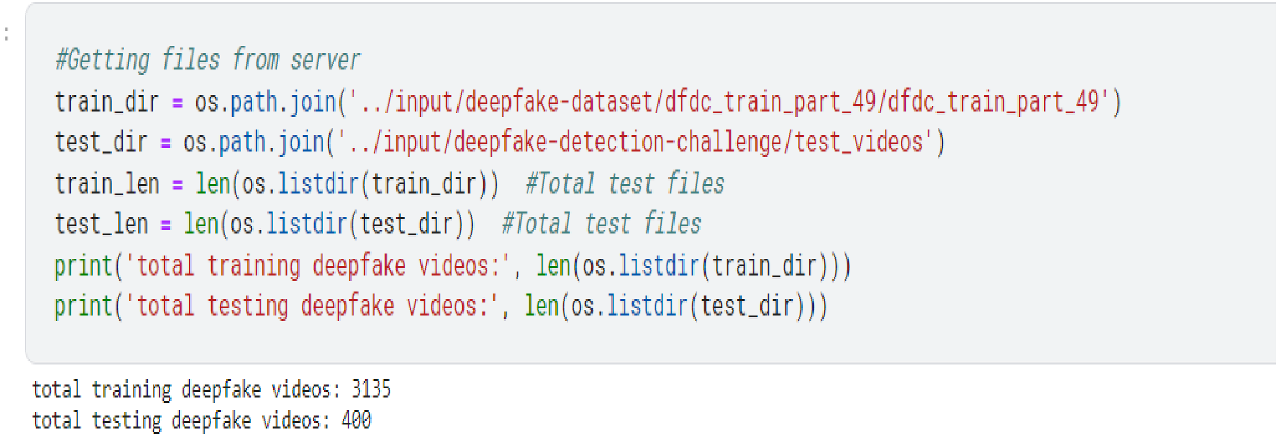
4.3 Screen shots of the various stages of the Project

Importing Packages –



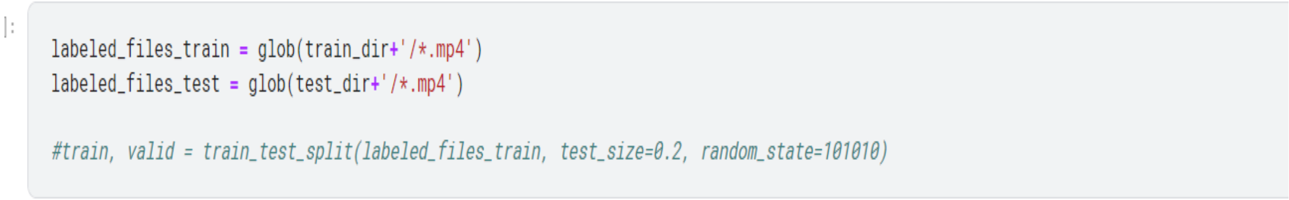
```
import tensorflow as tf
from tensorflow.keras.models import Sequential, load_model
from tensorflow.keras.layers import Conv2D, MaxPool2D,Dropout,Dense,Flatten, BatchNormalization, Activation, \
     Dropout, MaxPooling2D, Concatenate, GlobalMaxPooling2D, GlobalAveragePooling2D, \
    Lambda, Multiply, LSTM, Bidirectional, PReLU, MaxPooling1D
from tensorflow.keras.applications.nasnet import NASNetMobile, NASNetLarge, preprocess_input
from tensorflow.keras.optimizers import Adam, RMSprop
from tensorflow.keras.callbacks import ModelCheckpoint, EarlyStopping, ReduceLROnPlateau
from tensorflow.keras.optimizers import Adagrad
from sklearn.model_selection import train_test_split
#from sklearn.metrics import accuracy_score
from tensorflow.keras.preprocessing.image import load_img,array_to_img, ImageDataGenerator
from PIL import Image
#from sklearn.preprocessing import OneHotEncoder
from tensorflow.keras import Input, Model
from imgaug import augmenters as iaa
import imgaug as ia
from glob import glob
from random import shuffle
import matplotlib.pyplot as plt
import cv2
```

Pre-Processing –

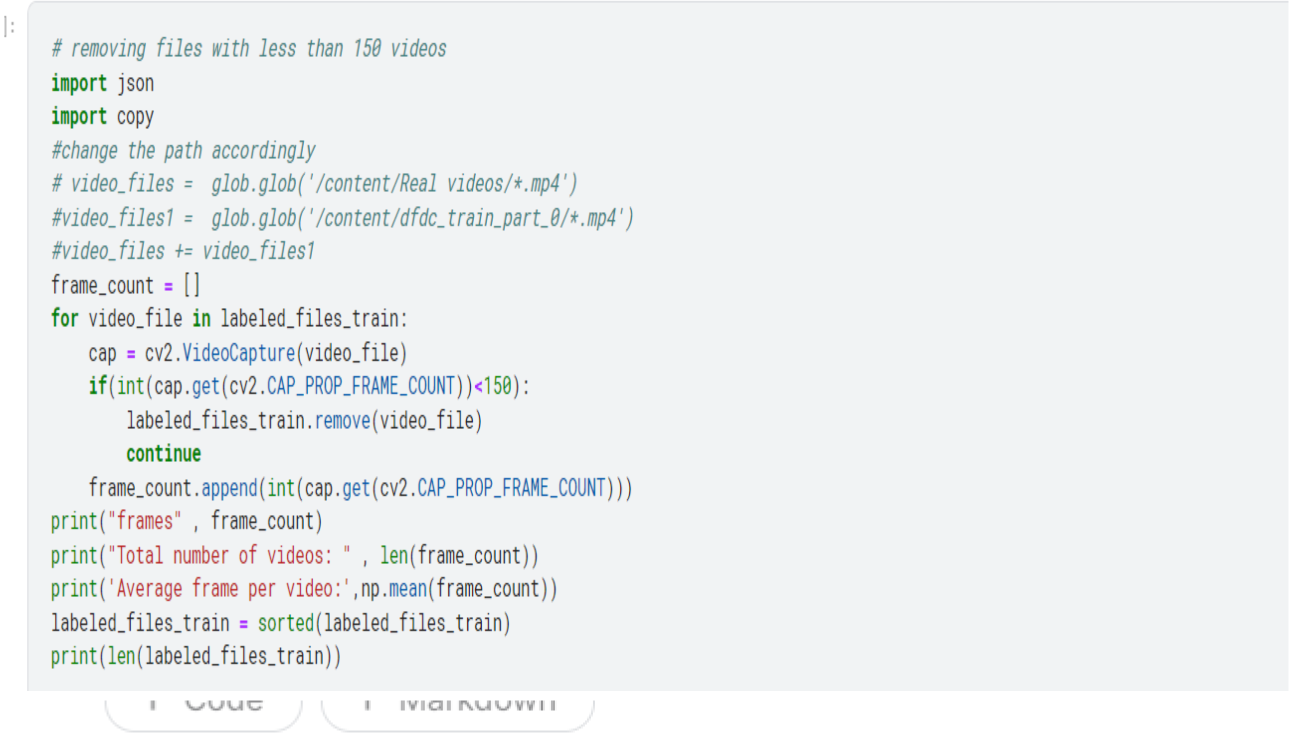


```
#Getting files from server
train_dir = os.path.join('../input/deepfake-dataset/dfdc_train_part_49/dfdc_train_part_49')
test_dir = os.path.join('../input/deepfake-detection-challenge/test_videos')
train_len = len(os.listdir(train_dir))  #Total test files
test_len = len(os.listdir(test_dir))  #Total test files
print('total training deepfake videos:', len(os.listdir(train_dir)))
print('total testing deepfake videos:', len(os.listdir(test_dir)))
```

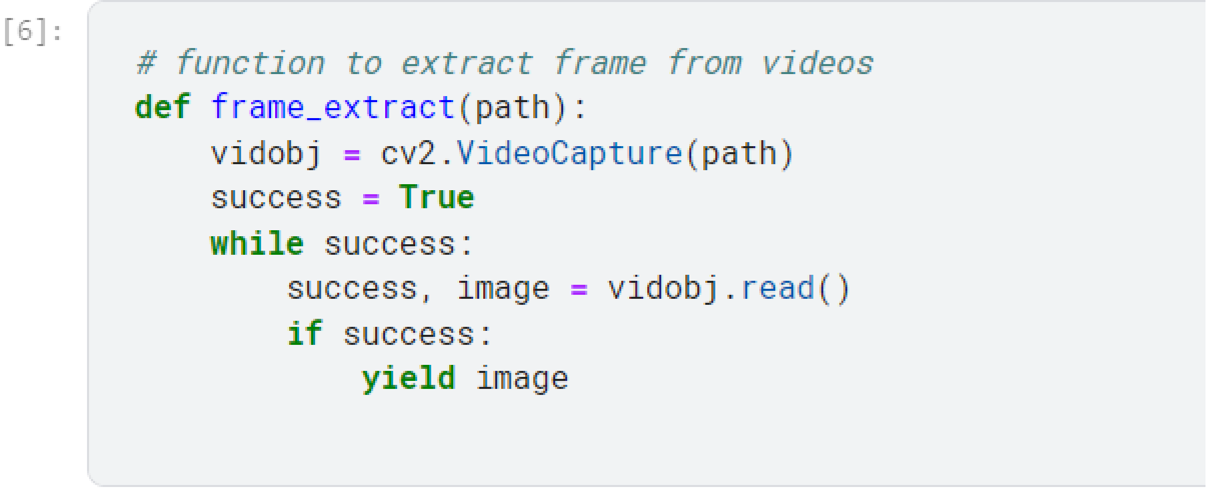
```
total training deepfake videos: 3135
total testing deepfake videos: 400
```



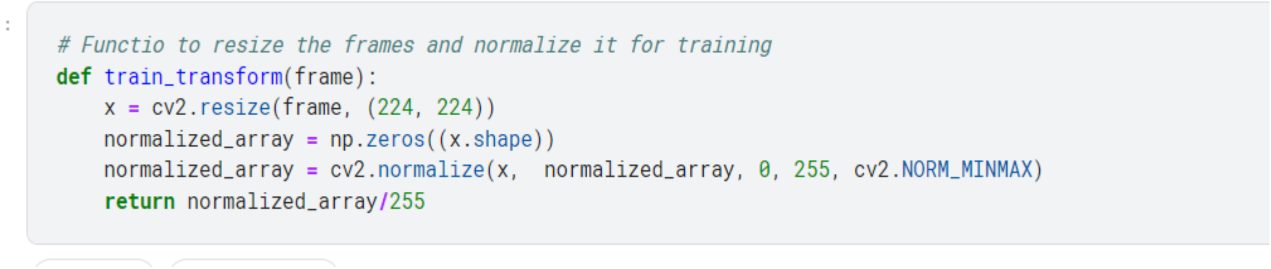
```
labeled_files_train = glob(train_dir+'/*.mp4')
labeled_files_test = glob(test_dir+'/*.mp4')

#train, valid = train_test_split(labeled_files_train, test_size=0.2, random_state=101010)
```



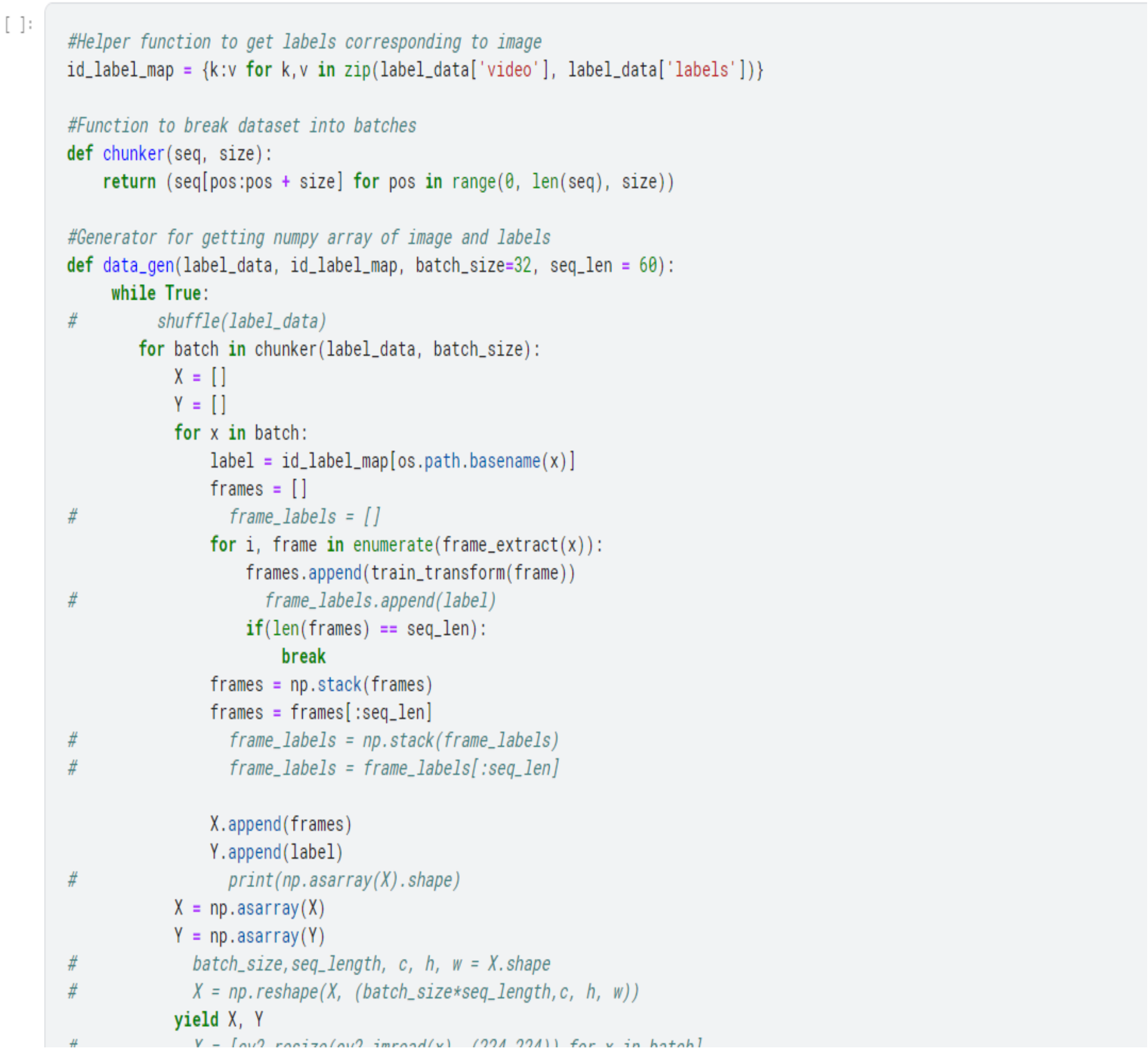
```
# removing files with less than 150 videos
import json
import copy
#change the path accordingly
# video_files =  glob.glob('/content/Real videos/*.mp4')
#video_files1 =  glob.glob('/content/dfdc_train_part_0/*.mp4')
#video_files += video_files1
frame_count = []
for video_file in labeled_files_train:
    cap = cv2.VideoCapture(video_file)
    if(int(cap.get(cv2.CAP_PROP_FRAME_COUNT))<150):
        labeled_files_train.remove(video_file)
        continue
    frame_count.append(int(cap.get(cv2.CAP_PROP_FRAME_COUNT)))
print("frames" , frame_count)
print("Total number of videos: " , len(frame_count))
print('Average frame per video:',np.mean(frame_count))
labeled_files_train = sorted(labeled_files_train)
print(len(labeled_files_train))
```



```
# function to extract frame from videos
def frame_extract(path):
    vidobj = cv2.VideoCapture(path)
    success = True
    while success:
        success, image = vidobj.read()
        if success:
            yield image
```

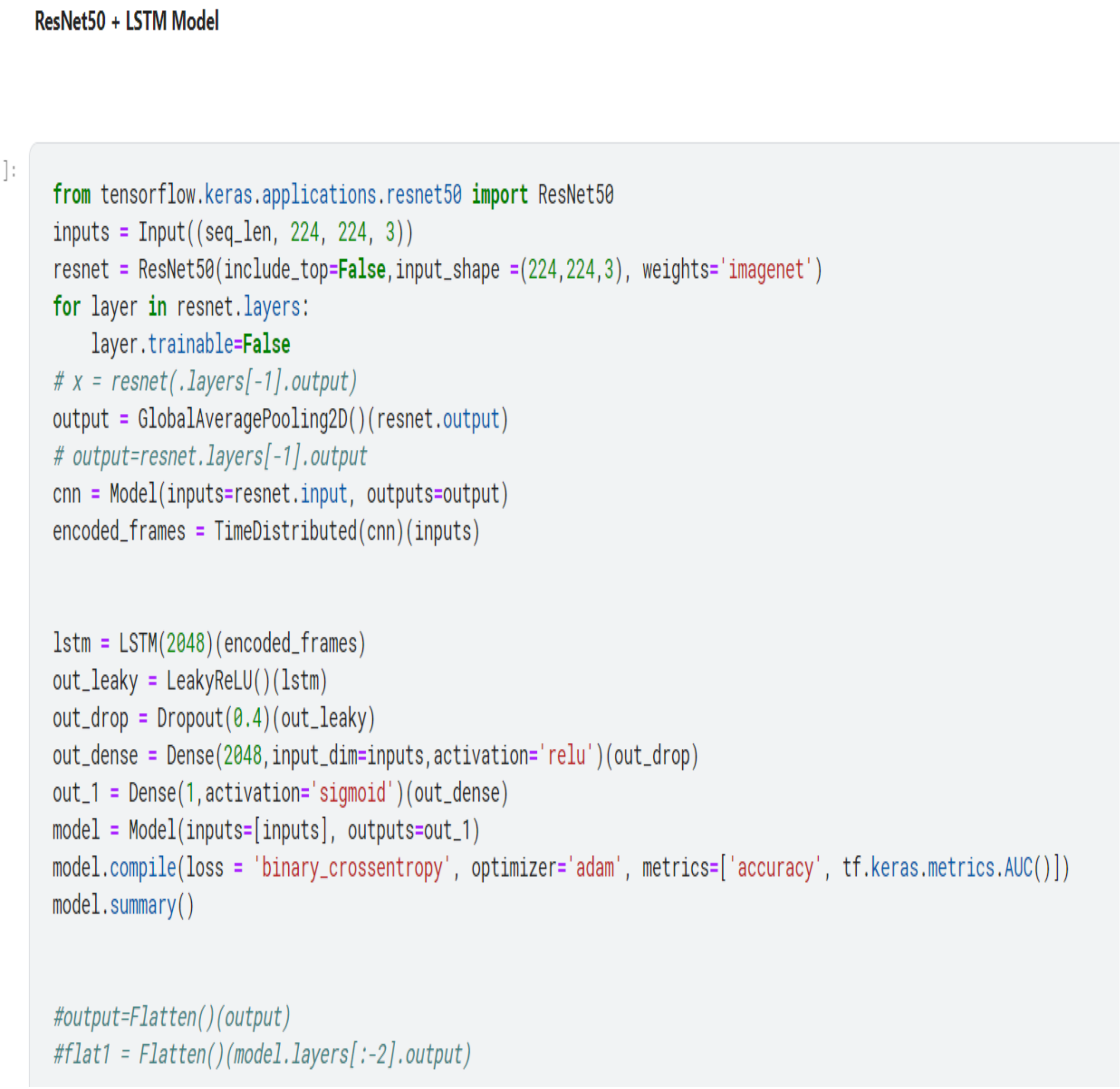


```python
# Functio to resize the frames and normalize it for training
def train_transform(frame):
    x = cv2.resize(frame, (224, 224))
    normalized_array = np.zeros((x.shape))
    normalized_array = cv2.normalize(x,  normalized_array, 0, 255, cv2.NORM_MINMAX)
    return normalized_array/255
```



```
[ ]:
#Helper function to get labels corresponding to image
id_label_map = {k:v for k,v in zip(label_data['video'], label_data['labels'])}

#Function to break dataset into batches
def chunker(seq, size):
    return (seq[pos:pos + size] for pos in range(0, len(seq), size))

#Generator for getting numpy array of image and labels
def data_gen(label_data, id_label_map, batch_size=32, seq_len = 60):
     while True:
#         shuffle(label_data)
        for batch in chunker(label_data, batch_size):
            X = []
            Y = []
            for x in batch:
                label = id_label_map[os.path.basename(x)]
                frames = []
#                 frame_labels = []
                for i, frame in enumerate(frame_extract(x)):
                    frames.append(train_transform(frame))
#                     frame_labels.append(label)
                    if(len(frames) == seq_len):
                        break
                frames = np.stack(frames)
                frames = frames[:seq_len]
#                 frame_labels = np.stack(frame_labels)
#                 frame_labels = frame_labels[:seq_len]

                X.append(frames)
                Y.append(label)
#                 print(np.asarray(X).shape)
            X = np.asarray(X)
            Y = np.asarray(Y)
#             batch_size,seq_length, c, h, w = X.shape
#             X = np.reshape(X, (batch_size*seq_length,c, h, w))
            yield X, Y
#             X = [cv2.resize(cv2.imread(x), (224,224)) for x in batch]
```

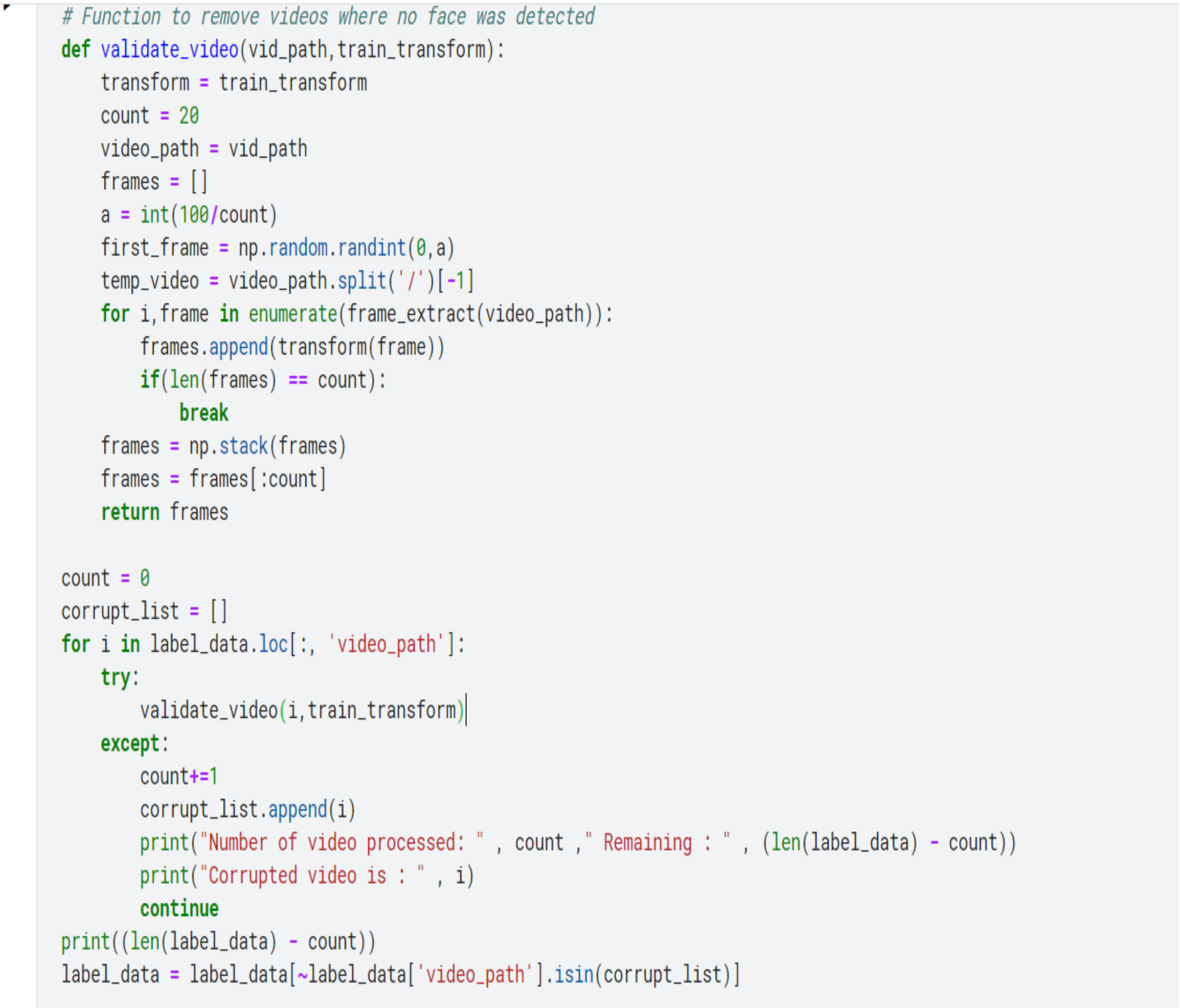
Model Architecture –



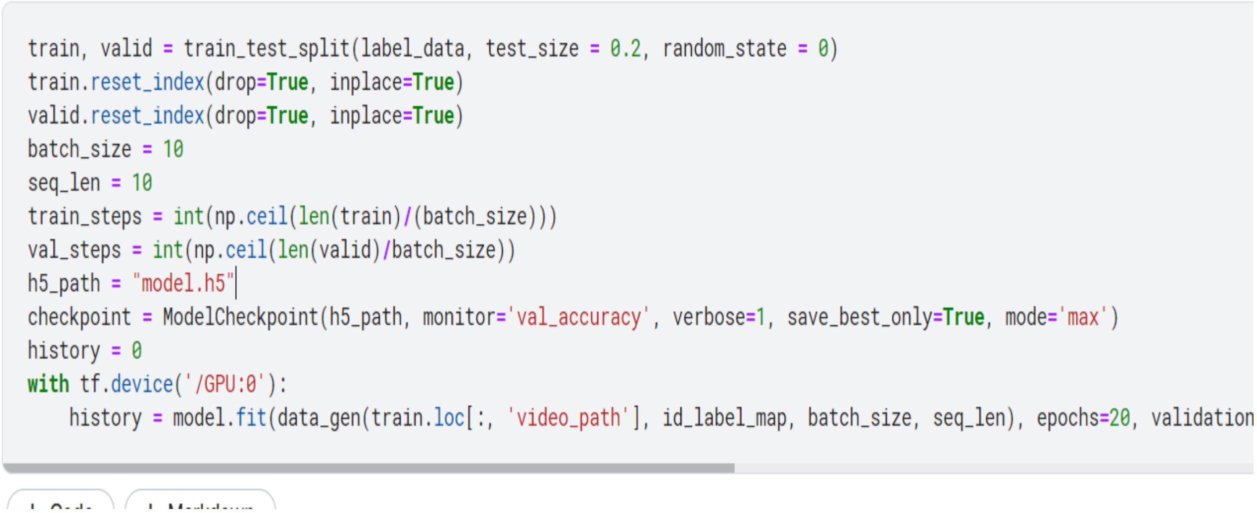
ResNet50 + LSTM Model

```
]:
from tensorflow.keras.applications.resnet50 import ResNet50
inputs = Input((seq_len, 224, 224, 3))
resnet = ResNet50(include_top=False,input_shape =(224,224,3), weights='imagenet')
for layer in resnet.layers:
    layer.trainable=False
# x = resnet(.layers[-1].output)
output = GlobalAveragePooling2D()(resnet.output)
# output=resnet.layers[-1].output
cnn = Model(inputs=resnet.input, outputs=output)
encoded_frames = TimeDistributed(cnn)(inputs)


lstm = LSTM(2048)(encoded_frames)
out_leaky = LeakyReLU()(lstm)
out_drop = Dropout(0.4)(out_leaky)
out_dense = Dense(2048,input_dim=inputs,activation='relu')(out_drop)
out_1 = Dense(1,activation='sigmoid')(out_dense)
model = Model(inputs=[inputs], outputs=out_1)
model.compile(loss = 'binary_crossentropy', optimizer='adam', metrics=['accuracy', tf.keras.metrics.AUC()])
model.summary()


#output=Flatten()(output)
#flat1 = Flatten()(model.layers[:-2].output)
```

Training Model -

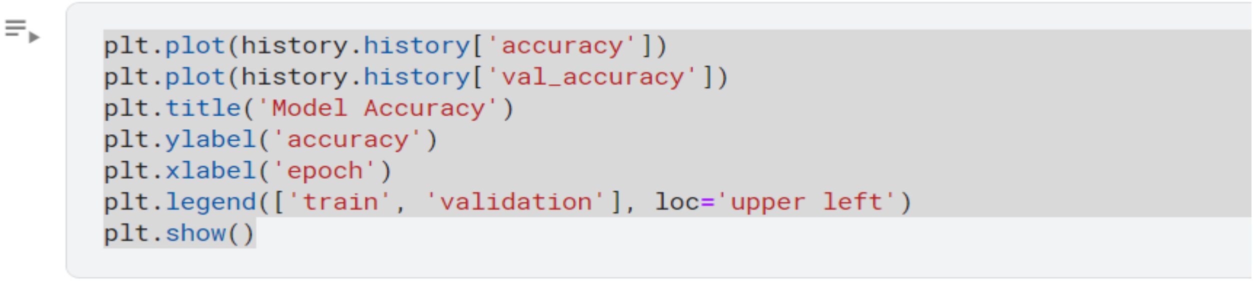


```
# Function to remove videos where no face was detected
def validate_video(vid_path,train_transform):
    transform = train_transform
    count = 20
    video_path = vid_path
    frames = []
    a = int(100/count)
    first_frame = np.random.randint(0,a)
    temp_video = video_path.split('/')[-1]
    for i,frame in enumerate(frame_extract(video_path)):
        frames.append(transform(frame))
        if(len(frames) == count):
            break
    frames = np.stack(frames)
    frames = frames[:count]
    return frames

count = 0
corrupt_list = []
for i in label_data.loc[:, 'video_path']:
    try:
        validate_video(i,train_transform)
    except:
        count+=1
        corrupt_list.append(i)
        print("Number of video processed: " , count ," Remaining : " , (len(label_data) - count))
        print("Corrupted video is : " , i)
        continue
print((len(label_data) - count))
label_data = label_data[~label_data['video_path'].isin(corrupt_list)]
```

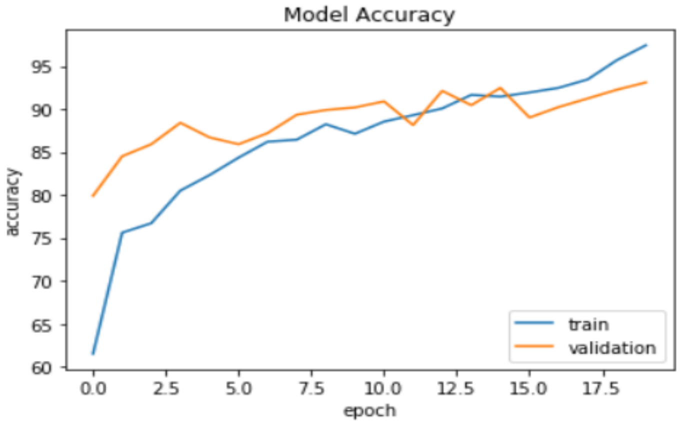


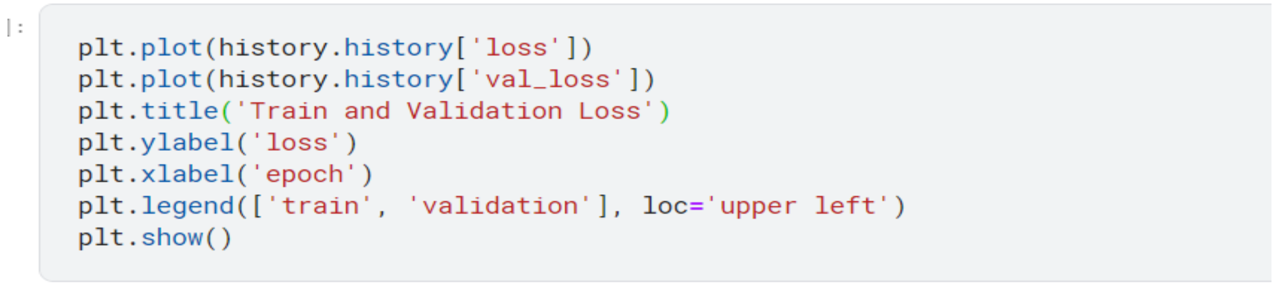
```
train, valid = train_test_split(label_data, test_size = 0.2, random_state = 0)
train.reset_index(drop=True, inplace=True)
valid.reset_index(drop=True, inplace=True)
batch_size = 10
seq_len = 10
train_steps = int(np.ceil(len(train)/(batch_size)))
val_steps = int(np.ceil(len(valid)/batch_size))
h5_path = "model.h5"
checkpoint = ModelCheckpoint(h5_path, monitor='val_accuracy', verbose=1, save_best_only=True, mode='max')
history = 0
with tf.device('/GPU:0'):
    history = model.fit(data_gen(train.loc[:, 'video_path'], id_label_map, batch_size, seq_len), epochs=20, validation
```

Model Accuracy & Loss –

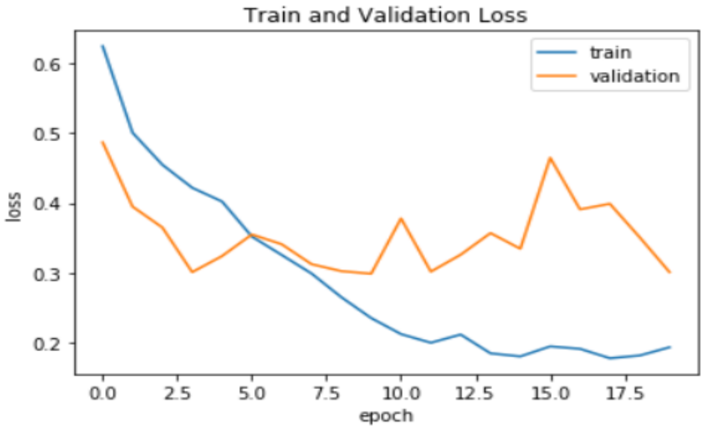


```
plt.plot(history.history['accuracy'])
plt.plot(history.history['val_accuracy'])
plt.title('Model Accuracy')
plt.ylabel('accuracy')
plt.xlabel('epoch')
plt.legend(['train', 'validation'], loc='upper left')
plt.show()
```





```
plt.plot(history.history['loss'])
plt.plot(history.history['val_loss'])
plt.title('Train and Validation Loss')
plt.ylabel('loss')
plt.xlabel('epoch')
plt.legend(['train', 'validation'], loc='upper left')
plt.show()
```



4.4 Comparisons

Our dataset was used to train multiple models. From those, ResNet50 + LSTM offers the most accurate testing and training results. Our observations have shown that the network regularly fails to engage in exceptionally practical and effective manipulation based on poorly-lit or foggy images that are inferior in quality. Despite their difficulty, manipulations made in excellent recordings appear to be accurately identified.

Table 2. Evaluation Results

Model	***Train Data***	***Test Data***
Custom Model	0.8523	0.8057
ResNet50+LSTM	**0.9795**	**0.9463**
MesoNet	0.9568	0.8997
DenseNet121	0.9699	0.9181

5. Chapter-5: CONCLUSIONS

5.1 Conclusions

The risks of face tampering in video are well known today among most people. A number of different fields are open to us as a result of DF: advanced media, VR, mechanical technology, schooling, and many more. In another context, they represent innovations that can ruin and undermine the whole society.

As part of the design, we incorporated a neural network based method to classify the video as either deep fake or genuine, as well as the certainty of the proposed model. The proposed method is inspired by the way the deep fakes are created by the GANs with the help of Autoencoders. Using the ResNet50 CNN, frame level detection is done, followed by video classification using the RNN and LSTM. As a result of the listed parameters in the paper, the proposed method can identify a fake video or a real video. Analysis of our technique shows that it can reliably identify DF on the web under genuine states of dispersion, with an average of **94.63%**. There is a high expectation that real-time information will be as accurate as possible. Having the opportunity to create a solution for a given issue without the need for an earlier hypothetical review is a central part of profound learning. However, we also have the option to understand this arrangement's beginning to evaluate its characteristics and constraints, so we spent considerable time imagining the channels within our network. Our prominent empirical findings have shown that the eyes and mouth are integral to recognizing appearances made with DF. It is anticipated that future devices will make our organizations more powerful, efficient, and to make them better able to understand profound businesses.

5.2 Future work

Model Ensembling

By utilizing a variety of modeling algorithms or by using a variety of training data sets, ensemble modeling enables various models to predict a result. Once all of the base models are combined, an ensemble model calculates a single final prediction for the inconspicuous data. Our prediction process can be enhanced by ensembleing multiple models (ResNet50, MesoNet, DenseNet, and Custom Model).

Data Augmentation

Adding augmented data to data analysis is a process that uses newly manufactured information added to previously existing data over time, or just slightly altered duplicates of data already available. It acts as a regularizer for an AI model and reduces overfitting.

Early Stopping

During training, early stopping can lead to better results. It enables you to specify an individual degree of training epochs and stop the preparation of a model after a certain amount of time has elapsed, enhancing a hold out validation dataset.

BlockChain

Researchers have found that blockchain technology can help detect deepfakes more effectively. There are not many examinations of blockchain's use in deepfake detection based on the fact that it has been used successfully in numerous fields. The ultimate tool for computerized provenance arrangement, it allows you to make an unchangeable chain of unique blocks of metadata.

Image Aggregation

Videos are often compressed when they are assessed, especially online videos, which can result in huge amounts of lost information. Then again, showing the same face in a succession of frames could help make the video more accurate in terms of its overall score if it allows for multiple experiences. The average of the network prediction over the video can be used to accomplish this. Theoretically, the association between frames of a single video is so strong that a gain in scores or a confidence interval indicator can't be justified. In practice, A majority of filmed faces contain stable clear frames to give viewers comfort. A majority of good predictions on a sample of frames taken from the video suggests that the effects of movement blur, face occlusion, and random misprediction on a sample of frames are outweighed.

Audio Manipulation

Until now, our model only worked on deepfakes that had been manipulated on the video side. However, thanks to the rapid advancement of the CNNs [4, 20], the Generative Adversarial Networks (GANs), and their variants, it is now possible to create deepfakes that have been manipulated on the audio side as well. Video with manipulated audio will be declared fake, but the faces will remain genuine, This makes it much more difficult to distinguish true, untampered audiovisuals from those that have been altered. Due to the ability to create a sound, video, and images that appear real, various stakeholders have taken action to deter such developments from being used maliciously. Consequently, researchers are attempting to devise deepfake detection mechanisms that can also detect synthesized audio.

5.3 Applications

- Fake News Detection
- Prevent damage to reputation of individuals
- Malicious hoaxes detection
- Prevent distortion of democratic discourse
- Reduces exacerbation of social divisions
- Prevents Financial Fraud

REFERENCES

[1] Joshua Brockschmidt, Jiacheng Shang, and Jie Wu. On the Generality of Facial Forgery Detection. In *2019 IEEE 16th International Conference on Mobile Ad Hoc and Sensor Systems Workshops (MASSW)*, pages 43–47. IEEE, 2019.

[2] Yuezun Li, Ming-Ching Chang, and Siwei Lyu. In Ictu Oculi: Exposing AI Generated Fake Face Videos by Detecting Eye Blinking. *arXiv preprint arXiv:1806.02877v2*, 2018.

[3] TackHyun Jung, SangWon Kim, and KeeCheon Kim. Deep-Vision: Deepfakes Detection Using Human Eye Blinking Pattern. *IEEE Access*, 8:83144–83154, 2020.

[4] Konstantinos Vougioukas, Stavros Petridis, and Maja Pantic. Realistic Speech-Driven Facial Animation with GANs. *International Journal of Computer Vision*, 128:1398–1413, 2020.

[5] Hai X. Pham, Yuting Wang, and Vladimir Pavlovic. Generative Adversarial Talking Head: Bringing Portraits to Life with a Weakly Supervised Neural Network. *arXiv preprint arXiv:1803.07716*, 2018.

[6] Yuezun Li, Siwei Lyu, "ExposingDF Videos By Detecting Face Warping Artifacts," in arXiv:1811.00656v3.

[7] Yuezun Li, Ming-Ching Chang and Siwei Lyu "Exposing AI Created Fake Videos by Detecting Eye Blinking" in arxiv.

[8] Huy H. Nguyen , Junichi Yamagishi, and Isao Echizen " Using capsule networks to detect forged images and videos ".

[9] Umur Aybars Ciftci, ˙Ilke Demir, Lijun Yin "Detection of Synthetic Portrait Videos using Biological Signals" in arXiv:1901.02212v2.

[10] https://www.kaggle.com/c/deepfake-detection-challenge/data

[11] Liu, M. Y., Huang, X., Mallya, A., Karras, T., Aila, T., Lehtinen, J., and Kautz, J. (2019). Few-shot unsupervised image-to-image translation. In Proceedings of the IEEE International Conference on Computer Vision (pp. 10551-10560).

[12] Park, T., Liu, M. Y., Wang, T. C., and Zhu, J. Y. (2019). Semantic image synthesis with spatially-adaptive normalization. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (pp. 2337-2346).

[13] DeepFaceLab: Explained and usage tutorial. Available at https://mrdeepfakes.com/forums/thread-deepfacelab-explained-and-usage-tutorial.

[14] DSSIM. Available at https://github.com/keras-team/keras-contrib/blob/master/keras_contrib/losses/dssim.py.

[15] Lattas, A., Moschoglou, S., Gecer, B., Ploumpis, S., Triantafyllou, V., Ghosh, A., & Zafeiriou, S. (2020). AvatarMe: realistically renderable 3D facial reconstruction "in-the-wild". In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (pp. 760-769).

[16] Ha, S., Kersner, M., Kim, B., Seo, S., & Kim, D. (2020, April). MarioNETte: few-shot face reenactment preserving identity of unseen targets. In Proceedings of the AAAI Conference on Artificial Intelligence (vol. 34, no. 07, pp. 10893-10900).

[17] Deng, Y., Yang, J., Chen, D., Wen, F., & Tong, X.(2020). Disentangled and controllable face image generationvia 3D imitative-contrastive learning. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (pp. 5154-5163).

[18] Tewari, A., Elgharib, M., Bharaj, G., Bernard, F., Seidel, H. P., P´erez, P., ... & Theobalt, C. (2020). StyleRig: Rigging StyleGAN for 3D control over portrait images. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (pp. 6142-6151).

[19] Li, L., Bao, J., Yang, H., Chen, D., & Wen, F. (2019). FaceShifter: Towards high fidelity and occlusion aware face swapping. arXiv preprint arXiv:1912.13457.

[20] Nirkin, Y., Keller, Y., & Hassner, T. (2019). FSGAN: subject agnostic face swapping and reenactment. In Proceedings of the IEEE/CVF International Conference on Computer Vision (pp. 7184-7193).

[21] Olszewski, K., Tulyakov, S., Woodford, O., Li, H., & Luo, L. (2019). Transformable bottleneck networks. In Proceedings of the IEEE/CVF International Conference on Computer Vision (pp. 7648-7657).

[22] Chan, C., Ginosar, S., Zhou, T., & Efros, A. A. (2019). Everybody dance now. In Proceedings of the IEEE/CVF International Conference on Computer Vision (pp. 5933-5942).

[23] Thies, J., Elgharib, M., Tewari, A., Theobalt, C., & Nießner, M. (2020, August). Neural voice puppetry: Audio-driven facial reenactment. In European Conference on Computer Vision (pp. 716-731). Springer, Cham.

APPENDIX

ResNet50 Structure: -

The advantage of using residual networks is that they can skip blocks of convolutional layers by using shortcut connections. In this architecture, the down-sampling process occurs at convolutional layers with stride of two, after which batch normalization is performed. In the final stage of the process, a ReLU activation is applied. The architecture consists of 101 layers in total, ending with a softmax activation at the top. Five stages comprise ResNet-50, each including convolution and identity blocks. Within each convolution block and within each identity block, there are 3 convolution layers. Over 23 million parameters can be trained with the ResNet-50.

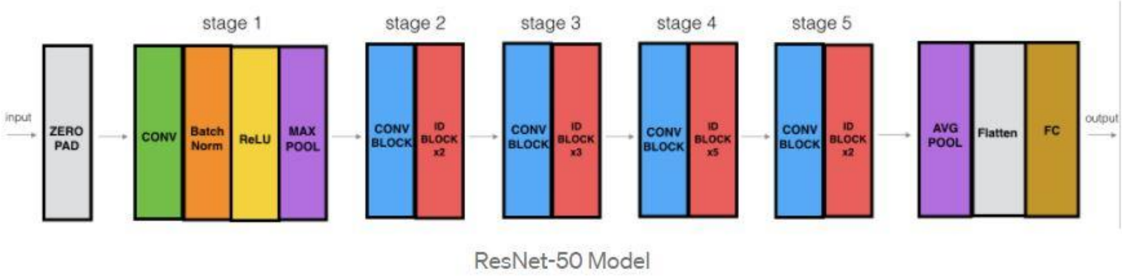


Figure 8. ResNet50 Model Structure

LSTM Layer: -

A LSTM network is a type of recurrent neural network that is capable of learning order dependence in sequence prediction problems. The purpose of such behaviour is to simplify complex problem spaces such as machine interpretation and speech processing. LSTMS are a complex aspect of deep learning. In order to gain a clear understanding of LSTMs and how terms like bidirectional and sequential go together, it is sometimes difficult to grasp their significance.

Significantly, LSTM is similar to a RNN cell. LSTM's inner workings can be viewed here. Here is the inside working of the LSTM organization. In the picture below, you can see that the LSTM consists of three sections, each of which fulfills a specific function.

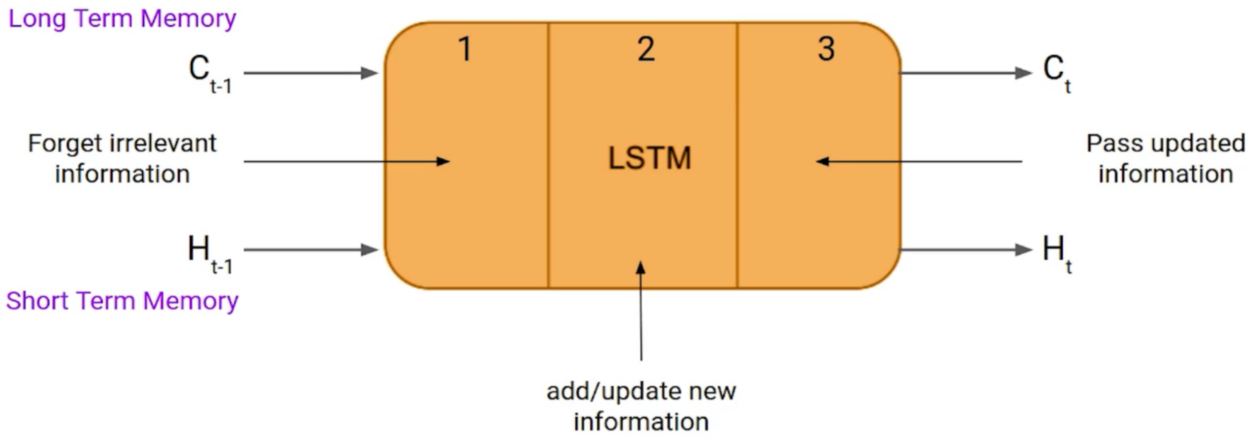


Figure 9. LSTM Architecture

Based on the timestamp of the past, the initial segment decides if the data from that time should be recalled or not. Cellular integration takes place in the following part when the new data is assimilated into this cell. Finally, in the third part, the cell passes the refreshed data from the current timestamp to the following timestamp.

LSTM cells consist of three pieces known as entryways. First there is the Forget door, then the Input entryway, and last but not least, the Output entryway. Similar to a RNN, an LSTM also has a secret state, where H(t-1) addresses the current timestamp's secret state and Ht addresses the past timestamp's secret state. LSTMs have an additional cell state addressed by C(t-1) and C(t) on their own, depending on whether they are addressing past or present timestamps.

Cell's state is called Short term memory, and the secret's state is called Long term memory. In light of this, it's interesting to observe that the cell state is what contains all the timestamps.