INTRODUCTION

1.1 Deepfake :

The term “Deepfake” is a combination of two words “Deep Learning” and “Fake”. Deepfake are a type of algorithm based on deep learning and Artificial Intelligence which can harness fake videos and images. Deepfakes can be created by traditional visual effects or computer-graphics approaches. Creation of deep fake videos are a simple task, but when it comes to detection, it’s a major challenge. [1]

The increasing sophistication of smartphone cameras and the availability of good internet connection all over the world has increased the ever-growing reach of social media and media sharing portals have made the creation and transmission of digital videos is easier than ever before. The growing computational power has made deep learning so powerful that would have been thought impossible only a handful of years ago. Like any transformative technology, this has created new challenges. So-called "Deepfake" produced by deep generative adversarial models that can manipulate video and audio clips. Spreading of the deepfake over the social media platforms have become very common leading to spamming and peculating wrong information over the platform. These types of the deepfake will be terrible and lead to threating, misleading of common people. [2]

The first deepfake video emerged in 2017 which was created by Chris Ume using visual and Artificial Intelligence effects. The idea behind deepfake videos is to make fake videos look realistic with the help of AI but this technology has mostly received negative criticism due to the dangerous possibilities it offers. The back bone of deepfake are deep neural networks trained on facial images to automatically map the expression from a source to a target. In this work, we will effectively distinguish fake videos from real videos. We will focus only on facial manipulations in the video. We won’t be discussing the audio manipulations in the deep fake video. We emphasize that deepfake should not be confused with adversarial machine learning which is to fool the machine learning algorithms. [1]

In a narrow definition, deepfakes (stemming from “deep learning” and “fake”) are created by techniques that can superimpose face images of a target person onto a video of a source person to make a video of the target person doing or saying things the source person does. This constitutes a category of deepfakes, namely face-swap. In a broader definition, deepfakes are artificial intelligence-synthesized content that can also fall into two other categories, i.e., lip-sync and puppet-master. Lip-sync deepfakes refer to videos that are modified to make the mouth movements consistent with an audio recording. Puppet-master deepfakes include videos of a target person (puppet) who is animated following the facial expressions, eye and head movements of another person (master) sitting in front of a camera. [3]

Deepfakes use specialized technique which generally modifies fixed areas on face which has to be used as a base for superimposition. The algorithm works in similar way for generating different deepfakes thus leaving some discrepancies during the editing process. Factors like compression changes, lighting differences along with temporal discrepancies like lip and eye movements can be specifically targeted to train models to detect Deepfake videos. Among the methods that have been suggested for Deepfake detection, Convolution Neural Networks (CNN) has been a popular choice. CNNs have shown great ability and scalability for applications regarding image and video processes when compared with other methods for supervised learning in Artificial Intelligence. CNN has the special ability to extract features from an image which can then be used for several applications. Along with feature extraction by a convolutional neural network other supervised learning tools can then be used for final classification for Deepfake to generate better and more precise models for Deepfake Detection. [4]

1.1 Deepfake Creation :

These deepfakes are created using artificial intelligence and deep learning methods. They rely on a type of convolutional neural network called auto-encoder which is used for encoding the input image by applying dimension reduction and image compression and a decoder which reconstructs the image from the constructed representation by the encoder. The auto encoder is a self-supervising algorithm as it uses targets provided by itself to train on. An upgrade to this method is GAN, i.e. Generative Adversarial Network, an unsupervised deep learning algorithm, which further improves the quality of deepfake created. [4]

1) **Autoencoder-decoder :**

Encoder-decoder model is a convolutional neural network which makes use of a single encoder with two distinct decoders. In this architecture, the feature vector of the face is extracted by the encoder and the decoder decodes the encoded image. The two networks share the encoding parameters as they use common encoder during training. This helps the network to identify the similarities between the two faces. Once the feature vector for each face is generated by the encoder, the next task is done by the decoder. In this process, decoder B decodes the feature vector produced by encoder A to rejuvenate face B from primary face A. [5]

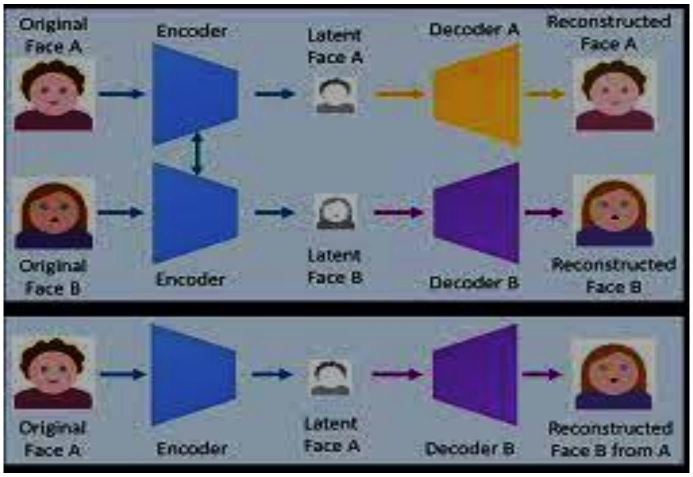


Figure 1: Deepfake Generation using Auto encoder-decoder

2) **Generative Adversarial Network:**

A GAN is made up of two competing neural networks generator and discriminator. The generator creates images as close to the real images as possible. The discriminator is then fed a training set containing both real and generated images and it tries to distinguish between them. As it continues to train, the generator makes images which are wrongly classified by the discriminator and the latter gets better at discriminating these images. Hence they form a pair that learn from each other and improve over time. In this way better quality deepfakes can be created. Deep Convolutional GANs (DCGANs) are even more effective as it uses convolutional layers to increase its efficiency. [4]

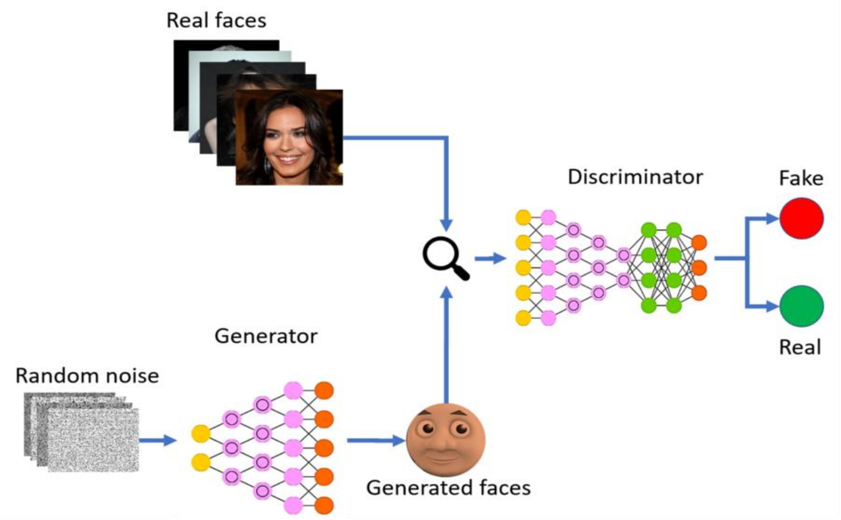


Figure 2: Deepfake Generation using GAN

An existing image of a person can be replaced with someone else’s face by superimposing the latter onto the given image using artificial neural networks. There is also a method called FaceSwap which can be used to swap faces in manipulated images and videos. It uses image compression to adjust the superimposed image onto the given image. The color of the two faces is also matched. The emerging artificial intelligence technologies can even replace expressions of one person to another in real time. The first application in the direction to create deepfake was FakeApp which allowed users to swap faces with other persons. More such applications have been built over time such as FaceSwap, DeepFaceLab, DFaker, FaceSwap-GAN, DeepFake-tf, and many more. As deep fakes are spreading faster than expected and creating very serious issues; it’s necessary to have automatic tools and technologies to detect fake content. [4]

1.1 Deepfake Detection:

A new deep learning-based method that can effectively distinguish Deepfake videos from the real ones. This method is based same process that is used to create the Deepfake by GAN. The method is based on the properties of the deepfake videos, due to limitation of computation resources and production time, the DeepFake algorithm can only synthesize face images of a fixed size, and they must undergo an affinal warping to match the configuration of the source's face. This warping leaves some distinguishable artifacts in the output deepfake video due to the resolution inconsistency between warped face area and surrounding context. This method detects such artifacts by comparing the generated face areas and their surrounding regions by splitting the video into frames and extracting the features with a ResNext Convolutional Neural Network (CNN) and using the Recurrent Neural Network (RNN) with Long Short Term Memory (LSTM) capture the temporal inconsistencies between frames introduced by GAN during the reconstruction of the deepfake. To train the ResNext CNN model, we simplify the process by simulating the resolution inconsistency in affine face wrappings directly. [5]

1) **Neural Network:**

Neural Networks consist of network of neurons which are the computational units. Neurons consist of a number (initial input or the output of the previous layer) and an activation function. Activation functions are the non-linear functions used which determine the output of the neuron. The commonly used activation functions are ReLU (Rectified Linear Unit), tanh, sigmoid, etc. The connection between layer has weights present and every layer has a bias. Backpropagation in neural networks adjusts these weights and biases according to the label of the training data. Thus, the values of weights and bias of the real and deepfake manipulated frames are different. Similar properties in images cause similar neurons to fire and thus they have similar values of weights and biases. [1]

The convolutional neural network is composed of neurons that are not fully connected to the next layer. The neurons are connected on the basis of filters used i.e. if the filter used is a 3x3 filter nine neurons in the nth layer determine the output of one neuron in the $(n+1)^{th}$ layer. The convolution operation is done on normalized pixels of the image. The convolution operation multiplies filter values and pixels and then adds the values and thus features are extracted. If fxf filter is applied on nxn image, the resulting output is of dimension,

$$\boldsymbol{n_{out} = [n_{in} + 2p - k\mathrm{s}] + 1}$$

n_{in}: number of input features

n_{out} : number of output features

k : convolution kernel size

p : convolution padding size

s : convolution stride size

The convolution operation thus reduces the dimension of image extracting features. Further Max pooling operation is done on pixels where the maximum value in the surrounding pixel is selected to reduce the spatial representation of the image and thus decreases the number of parameters in the network. [1]

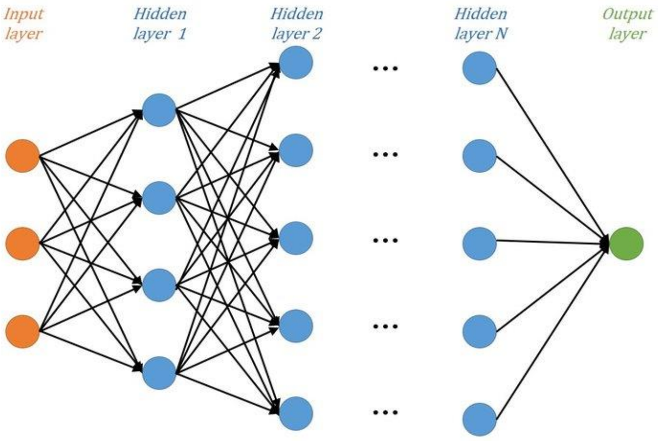


Figure 3: Neural Network Architecture

1) **ResNext Convolutional Neural Network:**

Instead of rewriting the classifier, we are proposing to use the ResNext CNN classifier for extracting the features and accurately detecting the frame level features. Following, we will be tuning the network by adding required layers and selecting a proper learning rate to properly converge the gradient descent of the model. The 2048-dimensional feature vectors after the last pooling layers are then used as the LSTM input in sequence. [6]

Convolutional neural networks are widely used for image recognition and classification problems. Pre-processing required for CNNs are very less when compared to other deep learning models and it can extract more features from a video or image. Hence the deep learning models also require a technology which can extract all the useful features from the videos before feeding them into a classification algorithm. Convolution operation preserves spatial relationship between the pixels while extracting the useful features from the video. Pre-trained ResNeXt-50 CNN serves this purpose. We will fine tune the model by using hyper parameters and adding extra layers the process is repeated until the model converges to global optimum. [7]

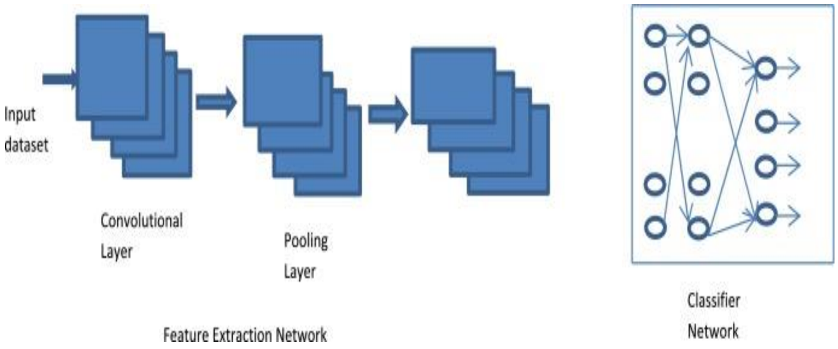


Figure 4: Convolutional Neural Network

1) **LSTM for Sequence Processing:**

Let us assume a sequence of ResNext CNN feature vectors of input frames as input and a 2-node neural network with the probabilities of the sequence being part of a deep fake video or an untampered video. The key challenge that we need to address is the de- sign of a model to recursively process a sequence in a meaningful manner. For this problem, we are proposing to the use of a 2048 LSTM unit with 0.4 chance of dropout, which is capable to do achieve our objective. LSTM is used to process the frames in a sequential manner so that the temporal analysis of the video can be made, by comparing the frame at 't' second with the frame of 't-n' seconds. Where n can be any number of frames before t. [7]

LITERATURE SURVEY

The explosive growth in deepfake video and its illegal use is a major threat to democracy, justice, and public trust. Due to this there is increased the demand for fake video analysis, detection and intervention. Some of the related word in deep fake detection are listed below:

Exposing Deep Fake Videos by Detecting Face Warping Artifacts used an approach to detects 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 Deep Fake 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. Exposing AI Created Fake Videos by Detecting Eye Blinking [2] describes a new method to expose fake face videos generated with deep neural network models. The method is based on detection of eye blinking in the videos, which is a physiological signal that is not well presented in the synthesized fake videos. The method is evaluated over benchmarks of eye-blinking detection datasets and shows promising performance on detecting videos generated with Deep Neural Network based software Deep Fake. Their method only uses the lack of blinking as a clue for detection. However certain other parameters must be considered for detection of the deep fake like teeth enchantment, wrinkles on faces etc. This method is proposed to consider all these parameters. Using capsule networks to detect forged images and videos 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. This method is proposed to be trained on noiseless and real time datasets. Detection of Synthetic Portrait Videos using Biological Signals approach extract biological signals from facial regions on authentic and fake portrait video pairs. Apply transformations to compute the spatial coherence and temporal consistency, capture the signal characteristics in feature sets and PPG maps, and train a probabilistic SVM and a CNN. Then, the aggregate authenticity probabilities to decide whether the video is fake or authentic. 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. [8]

David Guera Edward J. Delp put forth a paper based on Artificial Intelligence. The topic discussed in the paper is Convolution Neural Network (CNN), Recurrent Neural Network (RNN). The author tried to evaluate method against a large set of Deepfake videos collected from multiple video websites. Scenarios where these realistic fake videos are used to create political distress, black-mail someone or fake terrorism events are easily envisioned. This paper proposes a temporal recognition pipeline to automatically detect deep videos. It presents end-to-end trainable recurrent Deepfake video detection system. The author claimed that it is not unusual to find Deepfake videos where the manipulation is only present in a small portion of the video (i.e., the target face only appears briefly on the video, hence the Deepfake manipulation is short in time). To account for this, for every video in the training, validation and test splits, the system extracts continuous subsequences of fixed frame length that serve as the input of the system. This system works only with large dataset. The authors proposed an analysis composed of CNN to extract features followed by RNN network to capture erratic frames in the face swapping process. For the proposed system, a set of 600 videos were analyzed that were collected from multiple hosting websites. [9]

Yuezun Li et al. in put forth a paper based on Artificial Intelligence. The topic discussed in the paper was Convolution Neural Networks (CNN) and Recursive Neural Network (RNN). The author tried to create a new system that exposes fake faces based on eye blinking, that have been generated using Neural Networks. New developments in deep reproduction networks have greatly improved the quality and efficiency of producing authentic face videos. Therefore, in his paper, the author aims at analyzing the eye blin king in the videos, which is a psychological signal that is not well presented in the synthesized fake videos. The authors have performed Preprocessing in the first step in order to locate the face areas in each frame of the video. Then they have used Long Term Recurrent Convolution Network (LRCN) to capture temporal dependencies, as human eye blinking shows strong temporal dependencies. The Model training is then done using 2 steps: In the first step, they trained the VGG based CNN model based on a set of labeled training data consisting of eye regions corresponding to open and closed eyes. The model is trained using back-propagation implemented with stochastic gradient descent and dropout by probability 0.5 in fully connected layers. In step 2, the LSTM-RNN and the fully connected part of the network are co-trained using a back-propagation-by-time (BPTT) algorithm. The authors claimed that they have evaluated the LRCN methods with comparison to Eye Aspect Ratio (EAR) and CNN. They also used VGG16 as their CNN model to distinguish eye state. The author was able to make such claims because EAR method replies on eye landmarks to analyze eye state, in terms of the ratio between the distance of upper and lower lid, and the distance between left and right corner point. This method runs fast as merely cost in ratio computation. Also, CNN shows an exceptional well performance to distinguish the eye state on image domain. The author has displayed the authenticity of his claim via eye blinking detection on an original video and Deepfake generated fake video. Some gaps were presented by the author as the author implemented a full system with the intention of inventing a new method to detect fake videos such as; only eye blinking detection was done which is a relatively easy cue in detecting fake face videos. The author can try to find a more efficient system by adding hardware to the proposed system. [10]

Gustavo B. Souza et al. in put forth a paper based on Artificial Intelligence. The authors have discussed about the use of Width Extended Convolution Neural Networks. The authors tried to solve the problem of inefficiency of CNNs by implementing Width Extended Convolution Neural Networks (wCNN). The face is considered to be one of the most promising biometric features of human identification, including mobile devices. However, facial recognition systems can be easily fooled, for example, by providing a printed image, a 3D mask, or video recorded on the face of an official user. Recently, although some of the CNNs used (Convolutional Neural Networks) have obtained technical results in the detection of face loops, in many cases the proposed structures are very deep, because they are not suitable for limited hardware devices. In this work, we propose a functional architecture for face recognition based on the expanded CNN, which we call wCNN. Each part of the wCNN is trained, separately, in a given face area, where their output is computed to decide whether the faces presented on the sensor are real or fake. In order to evaluate the efficiency of the proposed wCNN in terms of processing required for face spoofing detection, they compared its performance with two state-of-the-art CNNs: Fine-Tuned VGG-Face a newly updated CNN based on random cassettes Instead of presenting the processing times, they present the amount of multiplication operations required by the adopted CNNs in the forward pass of each face image (or patches) for classification, since this measure is independent of the hardware used. Since the pass of the images through the neural networks is the core of the back propagation algorithm, the training of the CNNs is also usually much more complex for the architectures with more expensive forward passes. Its complexity tends to increase substantially since the backpropagation algorithm calculates partial derivatives for all the weights of the network. The author claimed that, besides presenting results compatible with state-of-the-art very deep CNNs, which they could not even train with their limited GPU, it saves lots of processing and time in training and test, being very suitable for environments with significant hardware restrictions, including mobile ones. The author has made such claims because of efficiency provided by the wCNN technology. The author has performed Face Spoofing Detection and Patch Net analysis as evidence for the results. As future work, they plan to evaluate the wCNN in other image domains, such as texture-based representations of the faces, and investigate the learning of local features for face spoofing detection in other color spaces. No new research problems can be thought of, based on the work done by the author. [11]

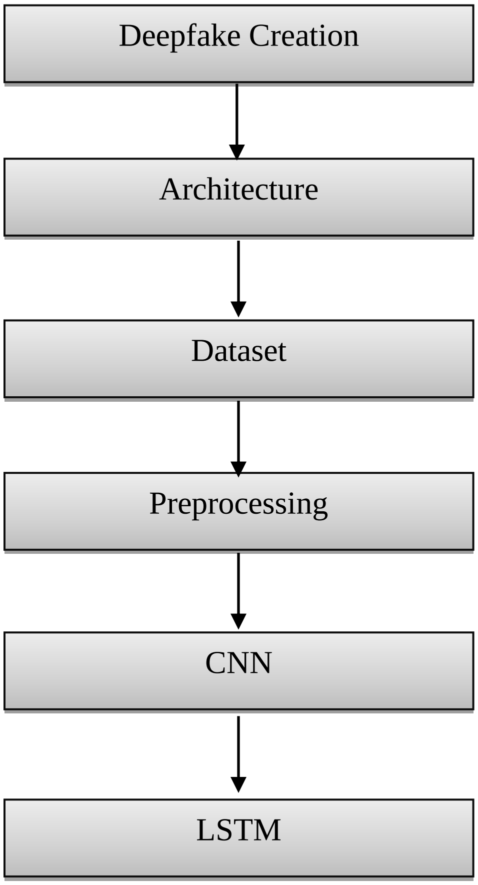
Haya R. Hasan and **Khaled Salah** in "**Combating deepfake Videos Using Blockchain and Smart Contracts**" put forth a paper on Blockchain Technology and Artificial Intelligence. The author proposes a blockchain based system for Deepfake videos. The author tried to solve the scenarios where Fake footage, images, audios, and videos (known as deepfakes) can be a scary and dangerous phenomenon, and can have the potential of altering the truth and eroding trust by giving false reality. The recent rise of AI, deep learning, and image processing have led the way to the production of deepfake videos. Deep videos are dangerous, and can have the power to distort the truth, confuse viewers and misleading facts. With the onset of social media networks, the proliferation of such content may remain unchanged and may add to the problems associated with the fabrication and ideas of corporate strategies. The owner (original artist) of a video first creates a smart contract where other artists can request a permission to edit, alter or distribute according to the terms and conditions of an agreement form. The agreement form is saved on the IPFS server and its hash is available as an attribute in the smart contract. The secondary artist requests first permission to edit, alter or share. A request sent by the secondary artist is also a confirmation to the terms and conditions of the agreement form. This request is assessed by the original artist and the result is then announced. The contract can handle multiple requests at the same time and can handle multiple different requests by the same artist. Once an artist gets an approval to their request, they create a child contract which is similar to the original contract and they update the parent's information. The second artist then asks for proof of his new contract from the first artist for the first video contract. The original artist then approves and grants the attestation after checking the newly created smart contract. A successfully attested smart contract would then be added as a child in the original smart contract. Hence, both the contracts point to each other as each one has the Ethereum address of the other as part of their attributes. The author claims that he made use of a decentralized storage system IPFS, Ethereum name service, and decentralized reputation system. The proposed solution framework, system design, algorithms, sequence diagrams, and implementation and testing details are generic enough and can be applied to other types of digital content such as audios, photos, images, and manuscripts. The author claims it as they can help combat deepfake videos and audios by helping users to determine if a video or digital content is raceable to a trusted and reputable source. The system also provides a trusted way for secondary artists to request permission from the original artist to copy and edit videos. The author has cited an example to assist a user in tracing back a video with multiple versions to its origin. If a video cannot be traced to its original publisher, then it cannot be trusted. The authors are in the process of developing front- end DApps for users to automate the establishment of proof of authenticity of published videos. Also, they plan to develop a pluggable DApp component to provide traceability and establish authenticity when playing or displaying videos within a web browser. Also, work is underway for designing and implementing a fully functional and operational decentralized reputation system. No new research problems can be thought of based on the work done by the author. [12]

Shuvendu Rana, Sibaji Gaj, Arijit Sur and **Prabin Kumar Bora** in put forth a paper based on Neural Network. The topic discussed in the paper are Convolution Neural Network (CNN), Dual tree complex wavelet transform (DT DCT), Depth image-based rendering (DIBR), Multiview video plus depth (MVD),3D highly efficient-video-coding(3D-HEVC). In this, the author tried to detect method to differentiate fake 3D video and real 3D video using CNN. The author tries to identify the real and fake 3D and pre- filtration is done using the dual tree complex wavelet transform to emerge the edge and vertical and horizontal parallax characteristics of real and fake 3Dvideos. The efficiency of the fake 3D video is examined over the training and testing dataset. Using the CNN, each video sequences in the training dataset used to train the CNN. The author claimed that due to this the time complexity and huge computing resources is required to achieve desired accuracy. Highresolution video sequences are used for training. The author implemented CNN architecture for proposed scheme. The author can try to find a more efficient a powerful mechanism for detecting real and fake videos. [13]

METHODOLOGY

As deep fakes are spreading faster than expected and creating very serious issues; it's necessary to have automatic tools and technologies to detect fake content. Social media channels are one of the widely used platforms where such contents are dumped without any verification. We are focusing mainly on the detection of deepfake videos from real ones. The model focuses on detecting all types of deepfakes such as Replacement Deepfake, Interpersonal Deepfakes and Retrenchment Deepfakes.

Flow of Methodology



Deepfake Creation:

It is well known that deep learning techniques have been successfully used to enhance the performance of image compression. Especially, the autoencoder has been applied for dimensionality reduction, compact representations of images, and generative models learning. Thus, autoencoders are able to extract more compressed representations of images with a minimized loss function and are expected to achieve better compression performance than existing image compression standards. The compressed representations or latent vectors that current convolutional autoencoders learn are the first cornerstone behind the face swapping capabilities of. The second insight is the use of two sets of encoder-decoders with shared weights for the encoder networks. Following figure shows how these ideas are used in the training and generation phases that happen during the creation of a deepfake video.

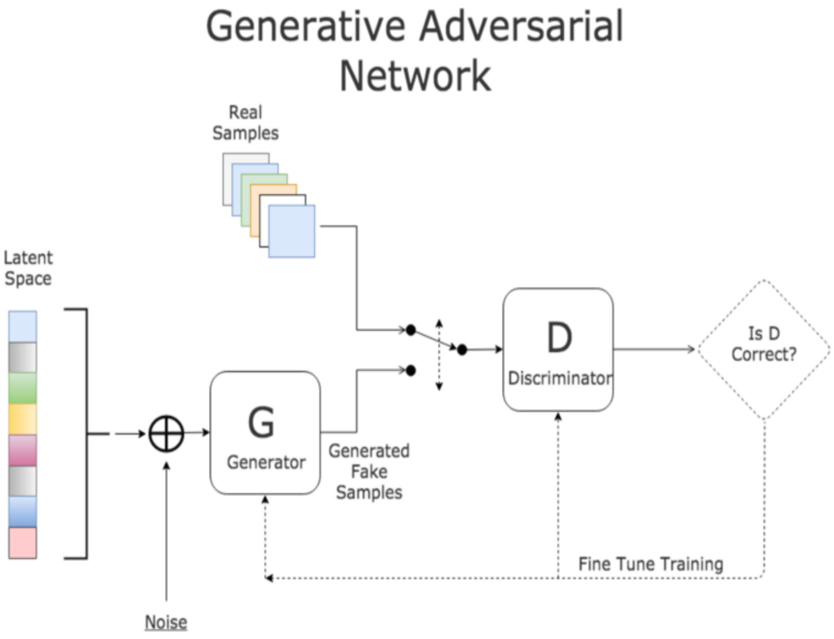


Figure 5: Generative Adversarial Network for Creation of Deepfake

Architecture:

Proposed model will effectively detect manipulated videos from real videos. In our system uses convolutional neural networks (CNN) to extract frame level features. These Features are then trained using a Long Short Term Memory (LSTM) network that classifies real and fake videos separately. Architecture of the system is given in figure 3. This novel deep fake detection model has 3 sections, first the data pre-processing stage, then feature extraction using CNN and finally LSTM network is used to classify if the video is fake or not.

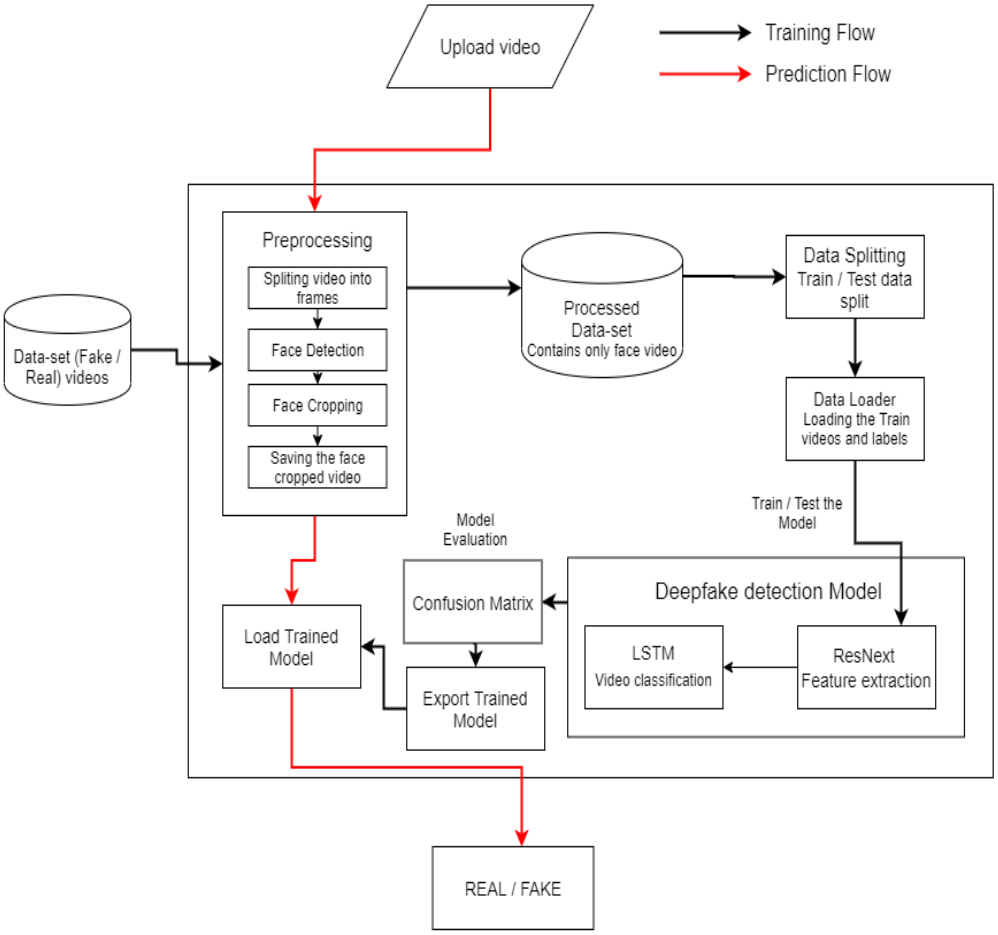


Figure 6: Architecture of Deepfake Detection Model

Dataset:

For making the model efficient for real-time prediction. We have gathered the data from different available datasets like FaceForensic ++, Deepfake detection challenge (DFDC) and Celeb-DF. Further, we have mixed the dataset and created our own new dataset, for accurate and real-time detection of different kinds of videos. To avoid the training bias of the model we have considered 50% Real and 50% fake videos.

The deep fake detection challenge (DFDC) dataset consists of certain audio alerted video, such as audio deepfake is out of scope for this post. We reprocessed the DFDC dataset and removed the audio altered videos from the dataset by running a python script.

After reprocessing the DFDC dataset, we have taken 1500 Real and 1500 Fake videos from the DFDC dataset. 1000 Real and 1000 Fake videos from the Face Forensic ++(FF) dataset and 500 Real and 500 Fake videos from the Celeb-DF dataset. Which makes our total dataset consists of 3000 Real, 3000 fake videos, and 6000 videos in total.

We built a custom dataset by combining different data from multiple sources. Majority of the videos are taken from Deep Fake Detection challenge (DFDC) repository which is open-source data contributed by tech giants like amazon, Facebook, Microsoft etc. 3000 plus videos are collected from DFDC and around 2000 plus videos from Face Forensics++ where manipulations of the video are done by latest video manipulation software like deepfake, Face2Face, Face Swap and Neural Textures. A good amount of videos are contributed by Caleb-DF repository which contains manipulated videos of celebrities with different age, ethnic group and gender. These videos are later split into train and test data with ratio of 70:30.

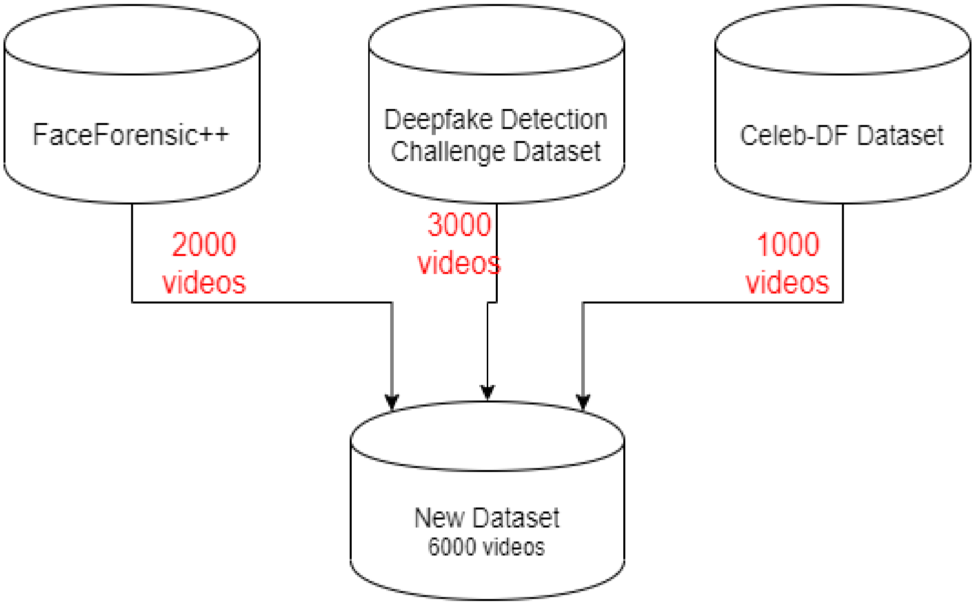


Figure 7: Dataset

Preprocessing:

Since our work only concentrates on the face manipulations on the video, we need proper pre-processing of data to avoid unnecessary computation. Video is split into frames and then faces in the videos are recognized using facial recognition model. Frames which do not have faces are removed in the pre-processing stage. To maintain uniformity, the first 300 frames are cropped from every video. Detecting faces from video is a difficult task. The system demands every feature of the face to properly diagnose if any manipulations have been done to the video. We use special haarcascade facial features to capture the faces from the video. Combining the haarcascade feature vector and openCV facial recognition module we can detect faces. This slightly modified function can extract face, profile face, eyes and smiles. For each object we are using different shapes and colors to distinguish between the features as shown in Figure 4. Green rectangle for frontal face, red circle for eye, red rectangle for smile and blue rectangle for profile face are used for this purpose.

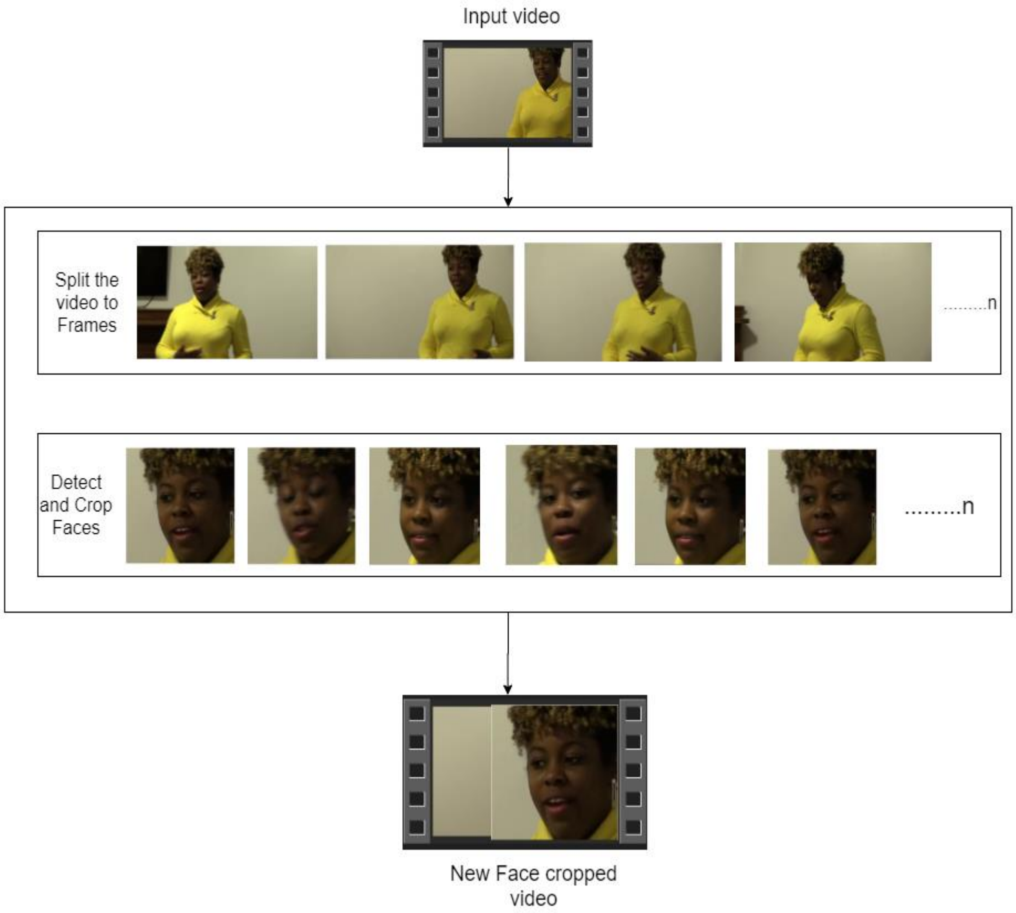


Figure 8: Preprocessing

Dataset preprocessing includes the splitting the video into frames. Followed by the face detection and cropping the frame with detected face. To maintain the uniformity in the number of frames the mean of the dataset video is calculated and the new processed face cropped dataset is created containing the frames equal to the mean. The frames that do not have faces in it are ignored during preprocessing. As processing the 10 second video at 30 frames per second i.e. total 300 frames will require a lot of computational power. Hence for experimental purpose we are proposing to used only first 100 frames for training the model.

ResNext Convolutional Neural Network:

Convolutional neural networks are widely used for image recognition and classification problems. Pre-processing required for CNNs are very less when compared to other deep learning models and it can extract more features from a video or image. So the deep learning models also require a technology which can extract all the useful features from the videos before feeding them into a classification algorithm. Convolution operation preserves spatial relationship between the pixels while extracting the useful features from the video. Pre-trained ResNeXt-50 CNN serves this purpose. We will fine tune the model by using hyper parameters and adding extra layers the process is repeated until the model converges to global optimum.

Instead of writing the rewriting the classifier, we are proposing to use the ResNext CNN classifier for extracting the features and accurately detecting the frame level features. Following, we will be fine-tuning the network by adding extra required layers and selecting a proper learning rate to properly converge the gradient descent of the model. The 2048-dimensional feature vectors after the last pooling layers are then used as the sequential LSTM input.

The model consists of resnext50_32x4d followed by one LSTM layer. The Data Loader loads the preprocessed face cropped videos and split the videos into train and test set. Further the frames from the processed videos are passed to the model for training and testing in mini batches.

LSTM for Sequence Processing:

Unlike recent works, we use the LSTM version of recurrent neural networks for classification. Features which are extracted using convolutional neural networks are fed into LSTM network for classifying a video as fake or real. LSTM’s are very much capable of learning long term dependencies. Chain like structure of LSTM decides what information we are going to pass in each stage and helps to keep the dependencies for long term. This way the model learns the manipulations made in the videos throughout the duration and identifies if the video has been subjected to any type of manipulation. Key challenge in LSTM that we need to address is the design flow to process the sequence of frames in a meaningful manner. LSTMs are very helpful in locating the spatial and temporal difference in the videos.

Let us assume a sequence of ResNext CNN feature vectors of input frames as input and a 2-node neural network with the probabilities of the sequence being part of a deep fake video or an untampered video. The key challenge that we need to address is the de- sign of a model to recursively process a sequence in a meaningful manner. For this problem, we are proposing to the use of a 2048 LSTM unit with 0.4 chance of dropout, which is capable to do achieve our objective. LSTM is used to process the frames in a sequential manner so that the temporal analysis of the video can be made, by comparing the frame at ‘t’ second with the frame of ‘t-n’ seconds. Where n can be any number of frames before t.

PERFORMANCE

The pipeline gives all the detailed information about the implementation of our system. A dataset of real and manipulated images based on the videos was created in our system. After creating dataset, the images in our dataset were preprocessed. The CNN model was created after the preprocessing and then training was done on the training set. After training, the model was run on the testing and validation sets. Further, tuning the parameters and hyperparameters was done to increase accuracy and decrease loss of the system. Lastly, running the model on sample video and detecting classes for each frame of the video was done.

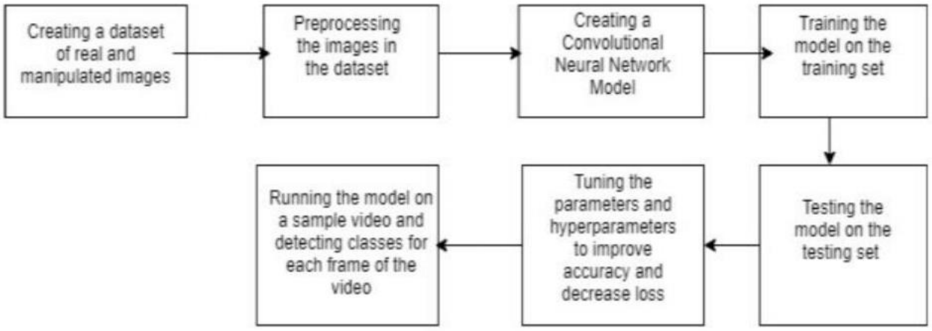
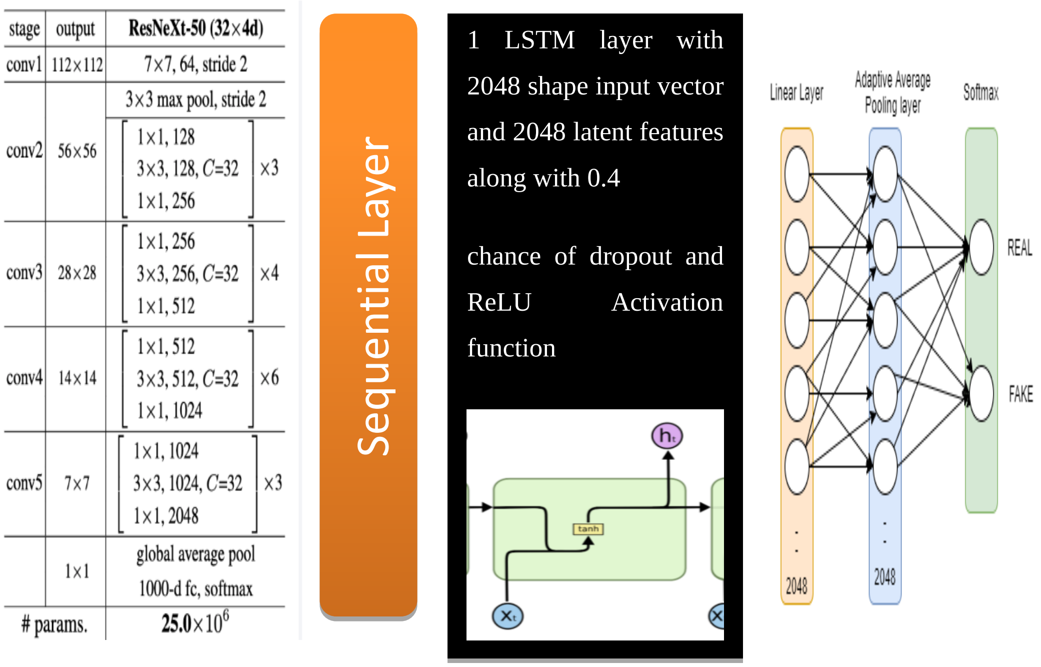

Creating a dataset of real and manipulated images
Preprocessing the images in the dataset
Creating a Convolutional Neural Network Model
Training the model on the training set
Testing the model on the testing set
Tuning the parameters and hyperparameters to improve accuracy and decrease loss
Running the model on a sample video and detecting classes for each frame of the video

Figure 9: Pipeline of Implementation for Deepfake Detection

The dataset is pre-processed before training the model. This involves face alignment and extraction. The proposed model targets faults induced during deepfake creation around the face outline. Thus, face extraction will extract the area that needs to be processed. Face alignment is used to account for different head positions that the target person may have in the deepfake video.

Further this dataset is passed to the preprocessing phase. In this phase, the videos are preprocessed and all the unrequired noise is removed from the videos. Only the required portion of the video that is face is detected and cropped. The first step 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 from the face. Later the crop frame is again converted to a new video by combining each frame of the video. The process is followed for each video which leads to creation of processed dataset containing face only videos. The frame that does not contain the face is ignored while preprocessing. As a video of 10 seconds at 30 frame per second will have total 300 frames and it is computationally very difficult to process 300 frames at a single time in the experimental environment. So based on our graphic processing unit, the computational power in the experimental environment, we have selected 150 frames as a threshold value. While saving the frames to the new dataset, we have only saved the first 150 frames of the video to the new video. To demonstrate the proper use of LSTM, we have considered the frames in sequential manner. That is first 150 frames and not randomly. The newly created video is saved at frame rate of 30 frames per second and resolution of 112*112. The processed video is passed to the model for training.



stage	output	ResNeXt-50 (32×4d)
conv1	112×112	7×7, 64, stride 2
conv2	56×56	3×3 max pool, stride 2
		[1×1, 128 3×3, 128, C=32 1×1, 256] ×3
conv3	28×28	[1×1, 256 3×3, 256, C=32 1×1, 512] ×4
conv4	14×14	[1×1, 512 3×3, 512, C=32 1×1, 1024] ×6
conv5	7×7	[1×1, 1024 3×3, 1024, C=32 1×1, 2048] ×3
	1×1	global average pool 1000-d fc, softmax
# params.		**25.0**×10^6

Sequential Layer

1 LSTM layer with 2048 shape input vector and 2048 latent features along with 0.4

chance of dropout and ReLU Activation function

Figure 10: Model Architecture

These are the different layers available in our model for training a deep neural network in an optimized way. We have use a ResNext 50, (32*4) dimensional model for feature extraction. ResNext 50 consists of 50 layers each layer having 32 nodes and is a 4-d model which is capable of learning $25.0*10^6$ parameters. The output of the ResNext model after the pooling layer is a feature vector which is then fed into the sequential layer. The sequential layer fed the input to the LSTM. We have used one LSTM layer with 2048 latent dimensions and 2048 hidden layers along with 0.4 chance of dropout. The output of the LSTM is further processed by deep layer and after average pooling layer and a softmax layer throws the output as real or fake.

If we look at the workflow of the model a new face cropped video is first split and passed to the ResNext model. ResNext model does the feature extraction as output and the output is fed into the LSTM. LSTM does the sequence processing of the video and classifies the video as real or fake.

Further the preprocessed video loaded into trained and test data with the ratio of 70% train videos and 30% test videos. The train and test split are balanced split. That is both the split contain 50% real videos and the 50 % fake videos. Data loader is used to load the videos and labels. After loading the videos are passed to the model. Training rule started for 20 epochs with batch size 4 on the train data with a learning rate of 0.0001 and wait decay of 0.001. After one epoch of training the model is tasted on the test videos. After the model has completed training then confusion matrix generation started which test the accuracy of the model on test data. The loss of validation and loss of accuracy are created which are displayed further. After training the model is exported into the pytorch so that it can be loaded into frontend and real time prediction.

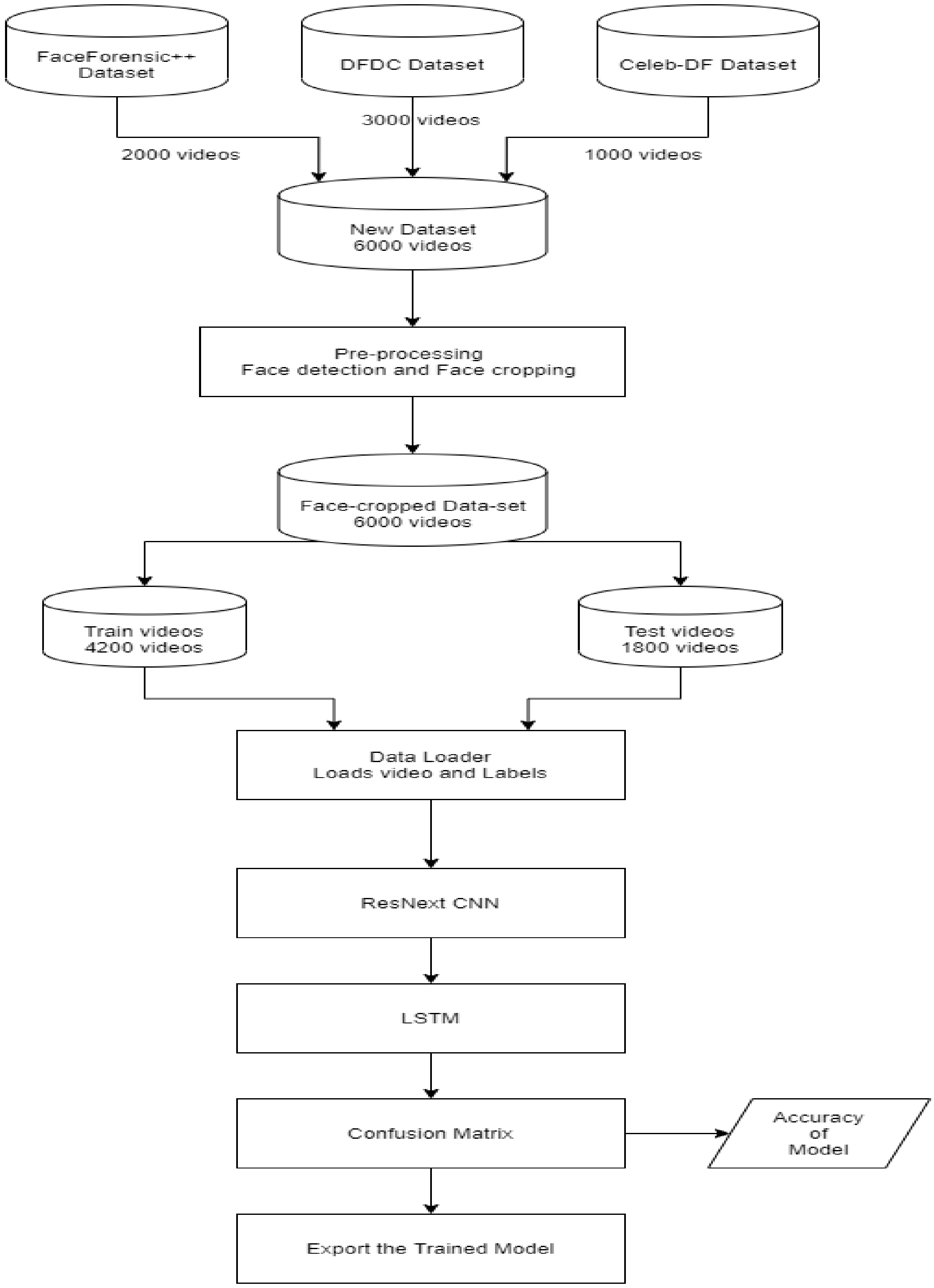


Figure 11: Training Workflow

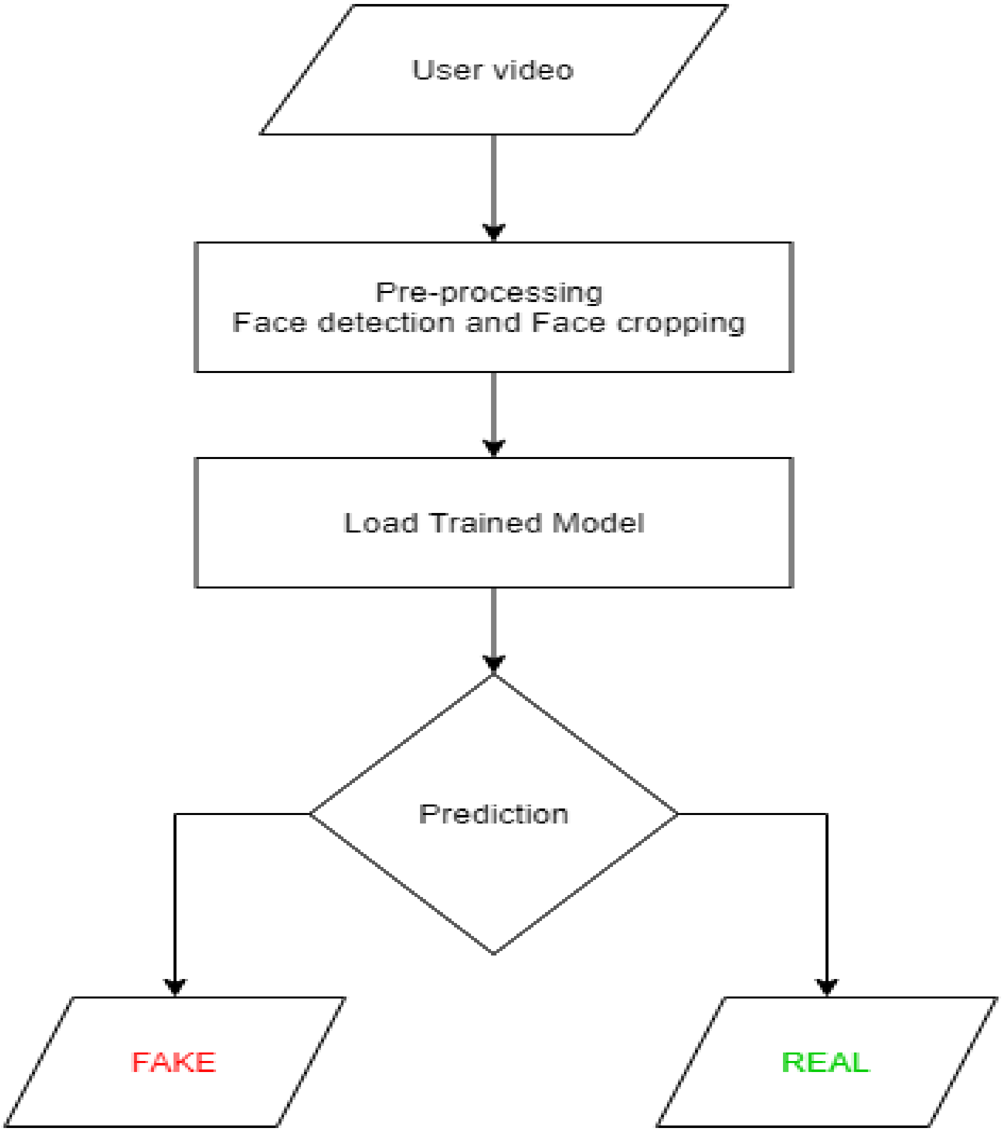
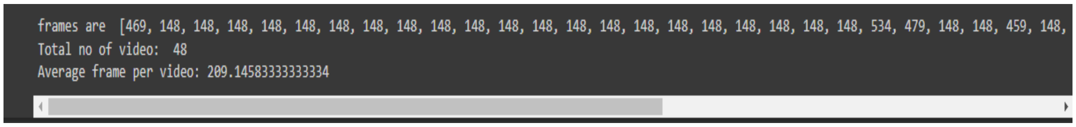


Figure 12: Prediction Workflow

In the flow diagram of prediction, the first phase is user video where we have to upload a video which we have to check whether it is a real video or the fake video. Once we upload a video, it is transferred to the preprocessing unit. The first step of preprocessing is to split the video into the frames. After splitting the video into the frames, the face is detected in the each of the frame and the frame is cropped along with the face. And the next phase is load trained model. The preprocessed video is passed to the trained model. Trained model processed the video on the learned parameter during the training phase. Based on the learn parameter, the model gives the output as fake or real.

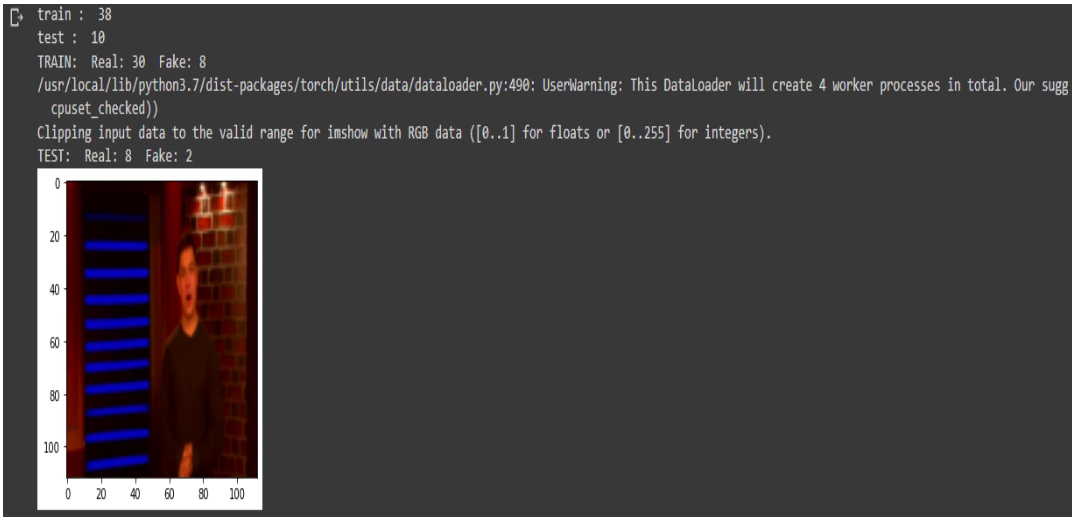
The model was trained on videos, labelled as ‘Fake’(Deepfakes) and ‘Real’ from the dataset and accuracy and loss calculations were obtained.

Following is the classification obtained for the dataset:



```
frames are  [469, 148, 148, 148, 148, 148, 148, 148, 148, 148, 148, 148, 148, 148, 148, 148, 148, 148, 148, 148, 148, 148, 534, 479, 148, 148, 459, 148,
Total no of video:  48
Average frame per video: 209.14583333333334
```

Load the label and video in the data loader:



```
train :  38
test :  10
TRAIN:  Real: 30  Fake: 8
/usr/local/lib/python3.7/dist-packages/torch/utils/data/dataloader.py:490: UserWarning: This DataLoader will create 4 worker processes in total. Our sugg
  cpuset_checked))
Clipping input data to the valid range for imshow with RGB data ([0..1] for floats or [0..255] for integers).
TEST:  Real: 8  Fake: 2
```

Accuracies for various optimizers and loss functions:

[Epoch 1/20] [Batch 9 / 10] [Loss: 0.126837, Acc: 100.00%]Testing

[Batch 2 / 3] [Loss: 0.033142, Acc: 100.00%]

Accuracy 100.0

[Epoch 2/20] [Batch 9 / 10] [Loss: 0.168780, Acc: 94.74%]Testing

[Batch 2 / 3] [Loss: 0.025223, Acc: 100.00%]

```
Accuracy 100.0
[Epoch 3/20] [Batch 9 / 10] [Loss: 0.176603, Acc: 100.00%]Testing
[Batch 2 / 3]  [Loss: 0.026255, Acc: 100.00%]
Accuracy 100.0
[Epoch 4/20] [Batch 9 / 10] [Loss: 0.194723, Acc: 100.00%]Testing
[Batch 2 / 3]  [Loss: 0.022563, Acc: 100.00%]
Accuracy 100.0
[Epoch 5/20] [Batch 9 / 10] [Loss: 0.138393, Acc: 100.00%]Testing
[Batch 2 / 3]  [Loss: 0.020490, Acc: 100.00%]
Accuracy 100.0
[Epoch 6/20] [Batch 9 / 10] [Loss: 0.146989, Acc: 100.00%]Testing
[Batch 2 / 3]  [Loss: 0.016126, Acc: 100.00%]
Accuracy 100.0
[Epoch 7/20] [Batch 9 / 10] [Loss: 0.080267, Acc: 100.00%]Testing
[Batch 2 / 3]  [Loss: 0.012407, Acc: 100.00%]
Accuracy 100.0
[Epoch 8/20] [Batch 9 / 10] [Loss: 0.063990, Acc: 100.00%]Testing
[Batch 2 / 3]  [Loss: 0.011633, Acc: 100.00%]
Accuracy 100.0
[Epoch 9/20] [Batch 9 / 10] [Loss: 0.103676, Acc: 97.37%]Testing
[Batch 2 / 3]  [Loss: 0.006332, Acc: 100.00%]
Accuracy 100.0
[Epoch 10/20] [Batch 9 / 10] [Loss: 0.065032, Acc: 100.00%]Testing
[Batch 2 / 3]  [Loss: 0.006009, Acc: 100.00%]
Accuracy 100.0
[Epoch 11/20] [Batch 9 / 10] [Loss: 0.084515, Acc: 100.00%]Testing
[Batch 2 / 3]  [Loss: 0.007480, Acc: 100.00%]
Accuracy 100.0
```

```
[Epoch 12/20] [Batch 9 / 10] [Loss: 0.099175, Acc: 100.00%]Testing
[Batch 2 / 3]  [Loss: 0.005243, Acc: 100.00%]
Accuracy 100.0
[Epoch 13/20] [Batch 9 / 10] [Loss: 0.119507, Acc: 100.00%]Testing
[Batch 2 / 3]  [Loss: 0.002412, Acc: 100.00%]
Accuracy 100.0
[Epoch 14/20] [Batch 9 / 10] [Loss: 0.081769, Acc: 100.00%]Testing
[Batch 2 / 3]  [Loss: 0.003401, Acc: 100.00%]
Accuracy 100.0
[Epoch 15/20] [Batch 9 / 10] [Loss: 0.117789, Acc: 100.00%]Testing
[Batch 2 / 3]  [Loss: 0.003647, Acc: 100.00%]
Accuracy 100.0
[Epoch 16/20] [Batch 9 / 10] [Loss: 0.120270, Acc: 100.00%]Testing
[Batch 2 / 3]  [Loss: 0.001993, Acc: 100.00%]
Accuracy 100.0
[Epoch 17/20] [Batch 9 / 10] [Loss: 0.474503, Acc: 97.37%]Testing
[Batch 2 / 3]  [Loss: 0.001207, Acc: 100.00%]
Accuracy 100.0
[Epoch 18/20] [Batch 9 / 10] [Loss: 0.093885, Acc: 100.00%]Testing
[Batch 2 / 3]  [Loss: 0.003030, Acc: 100.00%]
Accuracy 100.0
[Epoch 19/20] [Batch 9 / 10] [Loss: 0.133497, Acc: 100.00%]Testing
[Batch 2 / 3]  [Loss: 0.003882, Acc: 100.00%]
Accuracy 100.0
[Epoch 20/20] [Batch 9 / 10] [Loss: 0.159210, Acc: 97.37%]Testing
[Batch 2 / 3]  [Loss: 0.002073, Acc: 100.00%]
Accuracy 100.0
```

The predictions were done for the images in the dataset of the model. The function predict_classes() predicts the class that the images belong to. The accuracies were observed and seen to be the most for Categorial Cross Entropy. The figure 13 shows the increasing accuracy of validation sets as number of epochs increases. The below figure 14 shows the accuracy for training and validation accuracy.

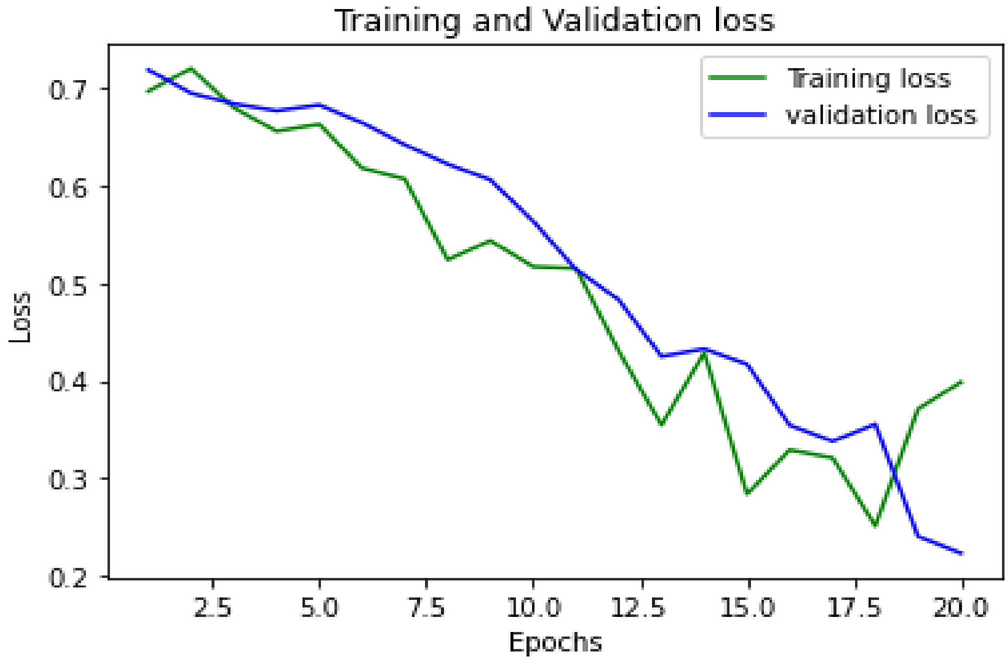


Figure 13: Training and Validation Loss

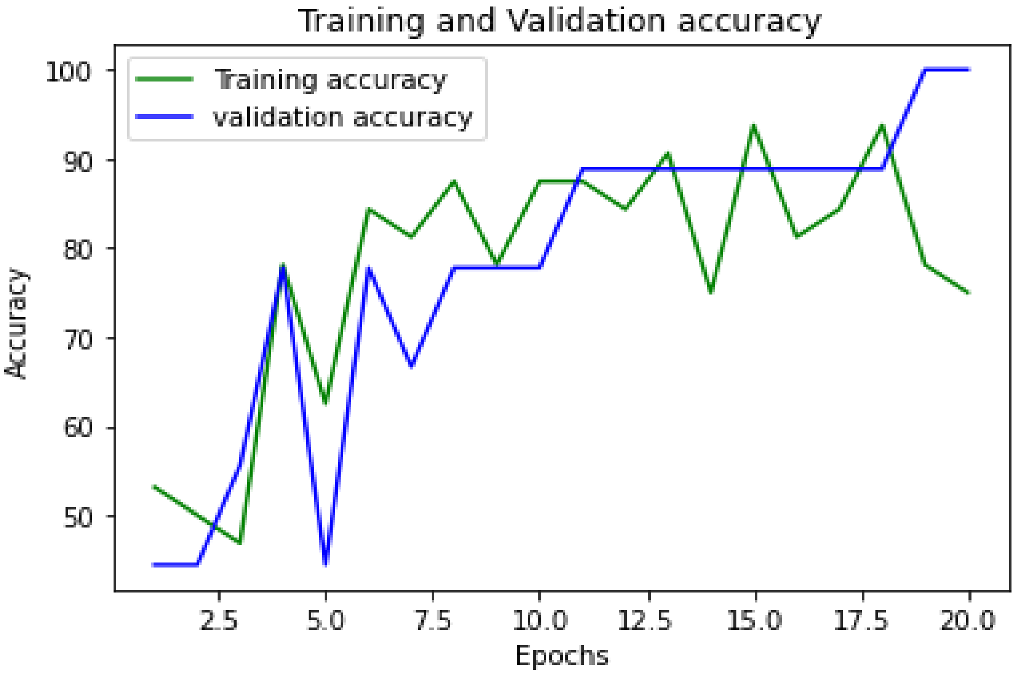


Figure 14: Training and Validation Accuracy

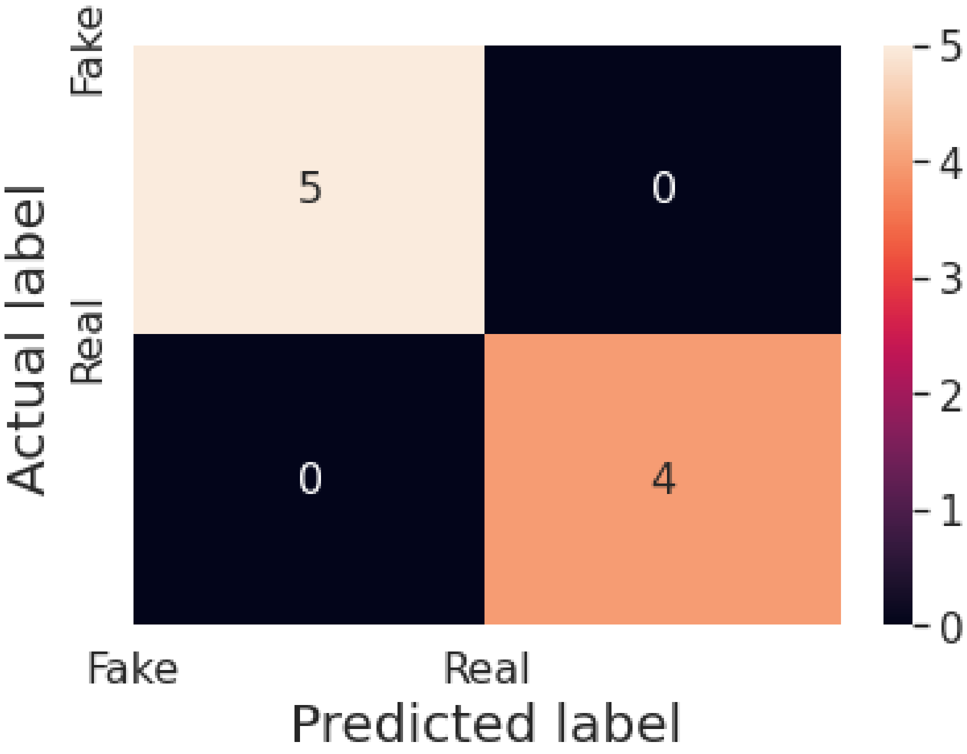


Figure 15: Confusion Matrix

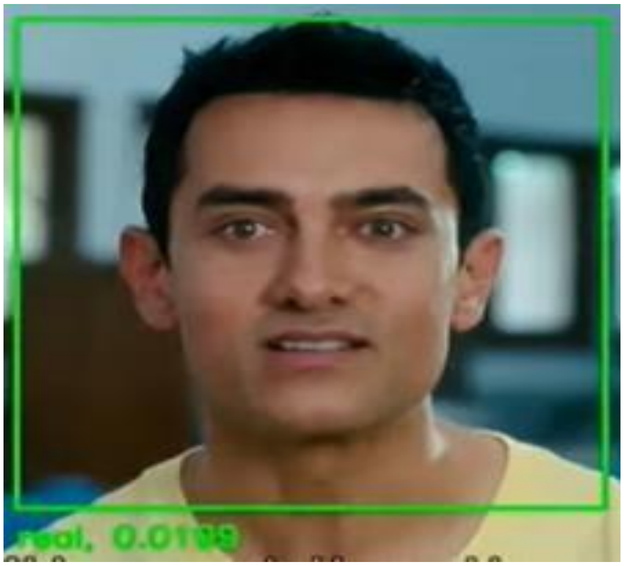


Figure 16: Frame classified as REAL

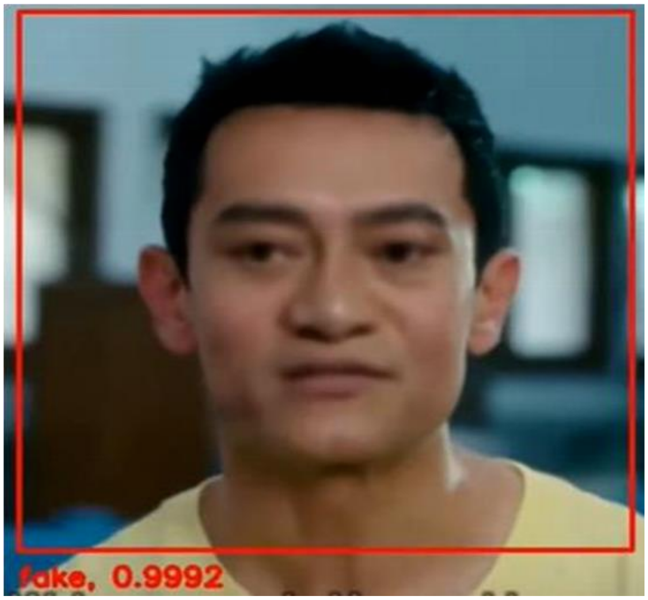


Figure 17: Frame classified as FAKE

Required Tools and Technology Used:

Note: Nvidia GPU is mandatory to run the application.

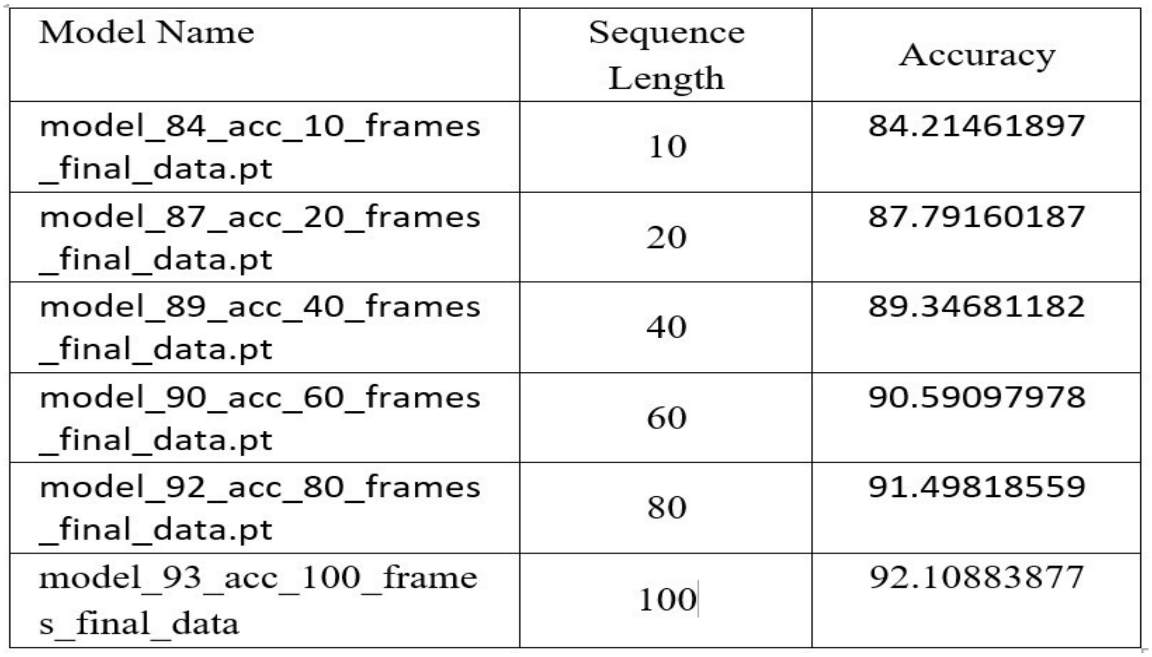
- CUDA version >= 10.0 for GPU
- GPU Compute Capability > 3.0

1) Programming Languages:
   - Python 3.6
   - JavaScript
2) Programming Framework:
   - PyTorch
   - Django
3) IDE:
   - Google Colab
   - Jupyter Notebook
   - Visual Studio Code
4) Cloud Service:

   Google Cloud Platform
5) Version Control:
   - Git

RESULTS

We evaluated our algorithm on sequence length of 10, 20,40,60,80,100.



Model Name	Sequence Length	Accuracy
model_84_acc_10_frames_final_data.pt	10	84.21461897
model_87_acc_20_frames_final_data.pt	20	87.79160187
model_89_acc_40_frames_final_data.pt	40	89.34681182
model_90_acc_60_frames_final_data.pt	60	90.59097978
model_92_acc_80_frames_final_data.pt	80	91.49818559
model_93_acc_100_frames_final_data	100	92.10883877

The above table represents the results achieved on our dataset by the model. The accuracy in the image depicts the test accuracy.

As we can observe in our results that the accuracy of the model is increasing with the increasing number of sequence lengths.

Based on our results we can say that, our model is able to predict whether the video is a deepfake or real by seeing just 10 frames i.e., less than 1 second (considering 30 frames per second video) with a decent accuracy of 84%.

CONCLUSION

Deepfake detection is a major need in today’s world and needs considerable detection techniques as detecting deepfakes will become more challenging in the future. As deepfakes can have major social and political impact improvements should be made continuously in its detection techniques.

Thus, Deepfake videos were studied and analyzed using this methodology and it also produces a good level of accuracy. The frames of the video were extracted and preprocessing was done. Subsequently, Image Classification was done and the images were labelled. With the help of Machine Learning algorithms, predictions were made on the dataset. Thus, any video can be analyzed using this methodology. It helps in detecting fake faces in a video which may have been manipulated, hence can prevent individuals from being defamed unknowingly.

We presented a neural network-based approach to classify the video as deep fake or real, along with the confidence of proposed model. The proposed method is inspired by the way the deep fakes are created by the GANs with the help of Autoencoders. Our method does the frame level detection using ResNext CNN and video classification using RNN along with LSTM. The proposed method is capable of detecting the video as a deep fake or real based on the listed parameters in paper. We believe that, it will provide a very high accuracy on real time data.

FUTURE SCOPE

Our method has not considered the audio. That’s why our method will not be able to detect the audio deep fake. But we are proposing to achieve the detection of the audio deep fakes in the future.

Further, different combinations of hyperparameters with respect to Neural Networks can be used and hyperparameter tuning can be done for the purpose of studying Deepfakes and the outputs of those algorithm models can be analyzed and compared, so that Deepfakes can be combated in the most efficient way, as it is one of the major threats looming large over the authenticity of videos. Modern technologies like Blockchain can be used for immutable storage in order to preserve the originality of videos.

Various ensemble learning techniques can also be implemented to further increase the accuracy of the model and account for variance in the dataset. Aggregation of results over each frame and over different learning models will thus give best results.

REFERENCES

1. Anuj Badale, Chaitanya Darekar, Lionel Castelino, Joanne Gomes, "Deepfake Video Detection Using Neural Netwok", International Journal of Engineering Research and Technology, ISSN: 2278-0181, NTASU 2020, (April 2021)

2. Abhijit Jadhav, Abhishek Patange, Jay Patel, Hitendra Patil, Manjushri Mahajan "Deepfake Video Detection using Neural Networks" International Journal for Scientific Research & Development, Vol. 8, Issue 1, (2021)

3. T.T. Nguyen, Q. V. H. Nguyen, C. M. Nguyen, D. T. Nguyen, Saied Nahavandi, "Deep Learning for Deepfakes creation and Detection", Institute of Electrical and Electronics Engineers, arXiv: 1909.11573v3 [cs.CV] (April 2021)

4. Aarti Karandikar, Vedita Deshpande, Sanjana Singh, Sayali Nagbhidkar, Saurabh Agrawal, "Deepfake Video Detection Using Convolutional Neural Network", International Journal of Advanced trends in computer Science and Engineering, ISSN 2278-3091, Volume 9 No. 2 (April 2020)

5. Abdul Jamshhed V. Janet B., "Deep Fake Video Detection Using Recurrent Neural Network", International Journal of Scientific Research in Computer Science and Engineering, E-ISSN: 2320-7639, Volume 9, Issue:2 (April 2021)

6. Akhil Sunil Kumar, Amruta Khavase, Himesh Rajendran, "Deepfake Video Detection Using Neural Nnetwork", International Journal of Innovative research in technology, ISSN; 2349-6002, Volume 7, Issue 12, (May 2021)

7. Pavel Korshunov, Sebastien Marcel, "Deepfakes A New Threat to face recognition? Assessment And Detection", arXiv:1812.08685v1 [cs.CV] (December 2020)

8. Yuezun Li, Ming-Ching Chang, Siwei Lyu, "Exposing AI Created Fake Videos by Detecting Eye Blinking" arXiv

1. D. Guera, E. J. Delp, "Deepfake Video Detection Using Recurrent Neural Networks", Institute of Electrical and Electronics Engineers, pp.1-6, (2018)

2. Y. Li, M. Chang, S. Lyu, "Exposing AI Created Fake Videos by Detecting Eye Blinking", Institute of Electrical and Electronics Engineers, Hong Kong, pp. 1-7, (2018)

3. G. Botelho de Souza, D. F. da Silva Santos, R. Gonsalves Pires, J. P. Papa and A. N. Marana, "Efficient Width-Extended Convolutional Neural Network for Robust Face Spoofing Detection", 7th Brazilian Conference on Intelligent Systems (BRACIS), Sao Paulo, pp. 230-235, (2018)

4. H. R. Hasan, K. Salah, "Combating Deepfake Videos Using Blockchain and Smart Contracts", Institute of Electrical and Electronics Engineers, vol. 7, pp. 41596-41606, (2019)

5. S. Rana, S. Gaj, A. Sur, P. K. Bora, "Detection of fake 3D video using CNN", Institute of Electrical and Electronics Engineers, Montreal, QC, pp. 1-5, (2016)

6. Ruben t. Ruben Vera-Rodriguez, Julian Fierrez, Aythami Morales, Javier Ortega-Garcia, " DeepFake and Beyond: A Survey of Face Manipulation and Fake Detection" , arXiv: 2001.00179v3 [cs.CV] (June 2020)

7. Deressa Wodajo, Solomon Atnafu, "Deepfake Video detection Using Convolutional Vision Transformer", Institute of Electrical and Electronics Engineering, arXiv: 2102.11126v3 [cs.CV] (March 2021)

8. Brian dalhansky, russ Howes, Ben Pflaum, Nicole Baram, Cristian Canton Ferrar, "The Deepfake zdetection Challenge Preview Dataset", Institute of Electrical and Electronics Engineering, arXiv: 1910.08854v2 [cs.CV] (23 October 2019)

1. D. E. King, “Dlib-ml: A machine learning toolkit,” JMLR, vol. 10, pp. 1755–1758, 2009

2. R. Raghavendra, Kiran B. Raja, Sushma Venkatesh, and Christoph Busch, “Transferable deep-CNN features for detecting digital and print-scanned morphed face images,” in CVPRW. IEEE, 2017

3. Umur Aybars Ciftci, Ilke Demir, Lijun Yin “Detection of Synthetic Portrait Videos using Biological Signals” in arXiv:1901.02212v2.

4. Pavel Korshunov and Sebastien Marcel, “Vulnerability assessment and detection of Deepfake videos”, IAPR International Conference 2019.