DEEPFAKE DETECTION IN VIDEO USING INTEGRITY VERIFICATION METHOD

A Project Report
Submitted by

Sandhya S	2019506080
Divya G	2019506028
Sivani S	2019506090

Under the supervision of

MR. V. Premanand

In partial fulfillment for the award of the

degree of

BACHELOR OF TECHNOLOGY

in

INFORMATION TECHNOLOGY

DEPARTMENT OF INFORMATION

TECHNOLOGY MADRAS INSTITUTE OF

TECHNOLOGY CAMPUS ANNA UNIVERSITY,

CHENNAI – 600044

DECEMBER 2022

ANNA UNIVERSITY : CHENNAI 600 044

BONAFIDE CERTIFICATE

Certified that this project report titled " **MULTIPLE OBJECT DETECTION AND TRACKING IN ADVERSE WEATHER CONDITION** " is the bonafide work of Sandhya S (2019506080), Sivani S (2019506090) and Divya G (2019506028) who carried out the project work under my supervision.

SIGNATURE	**SIGNATURE**
Dr. M. R. SUMALATHA **HEAD OF THE DEPARTMENT**	**MR. V. PREMANAND** **SUPERVISOR**
Professor	Teaching Fellow
Department of Information Technology	Department of Information Technology
MIT Campus, Anna University,	MIT Campus, Anna University,
Chennai - 600044.	Chennai - 600044.

ACKNOWLEDGEMENT

It is essential to mention the names of the people, whose guidance and encouragement made us accomplish this project.

We express our thankfulness to our project supervisor **MR. V. Premanand**, Department of Information Technology, MIT Campus, for providing invaluable support and assistance with encouragement which aided to complete this project.

We are thankful to the panel members **Dr. Dhananjay Kumar,** and **Dr. G. Rajesh,** Department of Information Technology, MIT Campus for their invaluable feedback in reviews.

Our sincere thanks to **Dr. M. R. Sumalatha**, Head of the Department of Information Technology, MIT Campus for catering all our needs giving out limitless support throughout the project phase.

We express our gratitude and sincere thanks to our respected Dean of MIT Campus, **Dr. J. Prakash**, for providing excellent computing facilities throughout the project.

SANDHYA S	**2019506080**
SIVANI S	**2019506090**
DIVYA G	**2019506028**

ABSTRACT

The new advancement in deep generative networks have remarkably improved the quality and efficiency in generating realistically-looking fake face videos. Deepfakes, or artificially generated visual renderings, can be intented to accuse a public figure or influence the person's opinion.With the recent discovery of Generative Adversial Network(GAN),an aggressor employing a traditional PC can create renditions that are realistic enough to fool the observer. Disclosing the deepfakes is thus turning into necessary precaution for reporters, social media platforms, and the general public. In this method, the eye blinking which is a physiological signal that is not well presented in the synthesized fake videos and head pose estimation in the videos which reveals facial landmarks specific to each person is analyzed. The eye blinking is detected using improved aspect ratio algorithm and General CNN-based classifier has been extended by incorporating the temporal relationship between consecutive frames which discloses the number of eye blinks, as eye blinking is a temporal process. The deep fakes are created by splicing central part of the synthesized face and merging synthesized face region into the original image.Head pose estimation is based on these observations and this introduces errors that can be revealed when head poses are estimated from the face images. Using features based on this condition and using a set of real face images and Deep Fakes, an SVM classifier is evaluated. This method is tested over benchmarks of video detection dataset namely UADFV, Celeb-df(v2) and shows the accuracy rate of 82% on detecting videos that are deep fakes.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF ABBREVATIONS

GAN	Generative Adversial Network
CNN	Convolutional Neural Network
LSTM	Long Short Term Memory
NAS	Neural Architecture Search
FC	Fully Connected Layer
IDE	Integrated Development Environment
RNN	Recurrent Neural Network
ReLU	Rectified Linear Unit
AI	Artificial Intelligence
ML	Machine Learning
SVC	Support Vector Classification
VLAD	Vector of Locally Aggregated Descriptors
AR	Aspect Ratio
AUC	Area Under Curve
LRN	Local Response Normalization
TPU	Tensor Processing Units
JVM	Java Virtual Machine
GPU	Graphics Processing Units
HEVC	High Efficiency Video Coding

VAE	Variational auto-encoder
SVM	Support Vector Machine
DCGAN	Deep Convolutional GANs
DTN	Domain Transfer Network
IOU	Intersection over Union
STM	Short-Term Memory
LRCN	Long Term Recurrent Convolutional Networks
AUC	Area under the ROC Curve
BPTT	Back-Propagation-Through-Time
VGG16	Visual Geometry Group
API	Application Programming Interface
CPU	Central processing unit
PM	Performance Measurement
R-CNN	Region Based Convolutional Neural Networks

CHAPTER 1
INTRODUCTION

1.1 OVERVIEW

The advancement of AI and technologies have lead to the growth of synthesized videos.The front line of this pattern is the alleged Deep Fakes. The highly regarded “DeepFake” is referred to a deep learning based technique that is able to produce fake videos by swapping the face of an individual by the face of another person. The term "Deepfake" is combination of "Deep Learning " and "Fake", which means the fake videos are created with the assistance of Deep Learning methods to create more realistic videos. This term was originated once a Reddit user named “deepfakes” claimed in late 2017 to possess developed a machine learning algorithmic rule that helped him to transpose celebrity faces into smut videos. These are generated by swapping synthesized faces into original images and videos using the deep neural networks.Fake images and videos as well as facial data generated by digital manipulation, especially with DeepFake ways, became a good public concern recently. Due to this increased technology there is very high chance of spreading misinformation and also it becomes very difficult to differentiate false information from true ones. These Deepfakes matters more in recent times as they can be used to manipulate and threaten the individuals and organisations. But by understanding this technology better, the corporate organisations can protect themselves from the manipulators. Historically, journalists have played an important role in critical examination of information before publication and dissemination. However, with misleading information, it becomes harder to verify for journalists thus creating fears among public.

Fake information may exploit feelings, assessments and even it could prompt disrupted and destabilizing public activities.Back then the video editing was a cumbersome and exhausting task due as there is no sophisticated software for editing like Photoshop. Editing a video of the length of 15-second video with 25 frames per second requires a tremendous number of editing activities involved as such editing 400 images. The majority of the videos can be recognized fairly based on some obvious visual artifacts and such a high realistic fake videos were uncommon. Conversely, the circumstance has been changed drastically with the new age of generative deep neural networks. which have the ability to of manipulating videos from the huge size of training data with the least manual editing.

Lately, utilizing a normal PC with an inbuilt GPU an attacker can make realistic videos that are unable to differentiate from true information which can deceive the common people. To create a deepfake video, the person's face is swapped and replaced with the another person , utilizing the facial recognition algorithm and deep learning network called Variational auto-encoder [VAE]. Autoencoders are one of the type of neural networks which consists of an encoder and decoder.Encoder decreases the image to a lower-dimensional latent space, whereas the image from the latent representation is regenerated by the decoder.This architecture is utilized by Deepfakes which uses a universal encoder to code a person into the latent space. The key features about the facial features and body posture of the person is contained in the latent representation. This can be decoded with a model trained explicitly for the target.

This implies the detailed information of the target's original video, will be laid over on the basal facial and body features represented in the latent space. A well-known enhancement to this structure is joining a generative adversarial network to the decoder. Generative adversarial network-GANs are an approach to generative architecture using deep learning methods[3]. In GANs DeepFakes are created by iterating an actual data-based generation and verification task through two opposite deep learning models Generator(G) and Discriminator. A

GAN trains a generator, the generator generates new images from the latent representation that is available from the source material, meanwhile the discriminator tries to determine whether image is generated or not. This makes the generator generate images that emulate reality incredibly well as any deformities would be found by the discriminator. The two algorithms advance constantly in a zero-sum game. This makes deepfakes hard to battle as they are constantly developing; any time a defect is determined; it can be resolved. At its starting stages, the Deepfakes videos could be distinguished through the unaided eyes as a result of the pixel's collapse phenomena that create unnatural visual curios in the skin tone visual artifacts. But over the period of time the GAN models began generating realistic faces because they were trained by using tens of thousands of images, utilizing that harmoniously spliced into the original video. The created video can prompt the manufacture of the subject’s identity in the video.This idea signifies that faces or specific body portions in videos or photographs can be synthesized to artificially obtain the information of other people. Thus Deepfakes have evolved to be highly indistinguishable from natural images. Thus, considerable research attention has been paid for the development of a different techniques or algorithm, that will verify the integrity of deepfakes.

1.2 RESEARCH CHALLENGES

In response to those more and more subtle and realistic manipulated content, massive efforts are being meted out by the analysis community to style improved strategies for face manipulation detection. Ancient fake detection strategies in media forensics are usually based mostly on: i) in-camera fingerprints, the analysis of the intrinsic fingerprints introduced by the camera device, both hardware and software package, like the camera lens, color filter array, and interpolation, and compression, among others, and ii) out camera fingerprints, the analysis of the external fingerprints introduced by a written material software package, like copy-paste or copy-move completely different components of the image, reduce the frame rate during a video, etc. However, most of the features considered in ancient fake detection strategies are extremely keen about the particular training situation, being so not sturdy against unseen conditions. This is of special importance within the era we tend to board as most media fake content is typically shared on social networks, whose platforms mechanically modify the first image/video, as an example, through compression and size operations. Most common DeepFake detection methods are based on binary categorization at the frame level, i.e., determining the chance of an individual frame as real or of DeepFake. Although straightforward and simple, there are 2 problems with this technique. First, the temporal consistency among frames isn't expressively considered, as (i) several DeepFake videos exhibit temporal artifacts and (ii) real or DeepFake frames tend to look in continuous intervals. Second, it necessitates an additional step once a video-level integrity score is needed: we've to combine the scores over individual frames to cipher such a score. The availability of large-scale datasets of DeepFake videos is an associate factor in the development of the DeepFake detection technique. Anyhow, a closer check-up on the Pristine videos in existing datasets shows some stark difference in visual quality to the particular Pristine videos circulated on the web. Several common visual artifacts may be found in these datasets together with low-quality synthesized faces, visible junction boundaries, color mismatch, visible components of the original face, and inconsistent synthesized face orientations. These artifacts are probably the results of imperfect steps of the synthesis technique and also the lack of curating of the synthesized videos before enclosed within the datasets. Correspondingly, high detection performance on these datasets might not bear sturdy relevancy once the detection strategies are deployed within the wild. A related issue is that DeepFake detection strategies trained using completely different DF datasets have hassle extending the performance to different datasets. A major challenge is that generator and discriminator in the GANs model have advanced to bypass verification in many methods. The proposed algorithm provides an alternative solution to find the integrity verification indices that are difficult to verify using the discriminator in the GANs model.

1.3 OBJECTIVES

- To detect deepfake videos by discriminating the real video distribution from that of forged videos.
- To study Integrity verification methods to detect deepfakes efficiently.
- To find the integrity verification indices that are difficult to verify using the discriminator in the GANs model.
- Comparative analysis of sequence learned eye blinker method with CNN-based classifier.
- To detect whether the video is real or pristine by analyzing eye blinks through temporal dependency phenomenon and alignment errors in facial landmarks estimated through head pose estimation by feeding it to SVM Classifier.

1.4 SCOPE OF THE PROJECT

We propose a model in which extracts the frames from the given video. Then we extract facial features and localize face from the images using OpenCV. Alignment error is calculated by finding the difference of cosine distance estimated from the whole face and head pose estimated from the central face region. Finally, we pass the number of eye blinks and alignment errors into the SVM classifier which classifies whether a video is a deepfake or not.

1.5 CONTRIBUTION

We have altered our model to do sequence learning that incorporates the temporal relationship between consecutive frames which memorizes the previous state to produce output when binary classifier is in a confused state to come up with a outcome. This improvisation in algorithm makes our model to perform efficiently even when eye states are ambiguous in the video dataset. Also we had trained the SVM classifier with combined data from eye blinks and head pose estimation to produce the desired output of real or pristine.

1.6 APPLICATIONS OF THE PROJECT

Determining the truth in the digital domain has become increasingly difficult. It is even difficult when dealing with deepfakes as they are mostly used to serve malicious purposes and almost anybody can generate deepfakes these days by using available and existing deepfake tools. Deepfakes can cause short- and long-term social harm. From the chance of making fake videos implicating proof to "counterfeit news" and publicity, this innovation can without much of a stretch be utilized for accursed purposes and blackmail. The advancement in digital technology, the wide accessibility of cell phones, and the increasingly widespread of social networks such as Facebook, Twitter, WhatsApp, Instagram, and Snapchat and video sharing portals namely YouTube and Vemeo have created the creation, redaction, and propagation of digital videos a lot of convenient than ever. This has conjointly achieved the digital meddling of videos as a good way to propagate falsified data. In mass media where advances in digital communication and AI threaten to magnify the scale, persistence, and consequence of misinformation. Media owners such as broadcasters and journalists have a significant role in vetting information before publication and dissemination. AI-based simulated synthetic media may step up the already declining trust in media. Such disintegration can add up to a culture of factual relativism, fraying undeniably stressed textures of common society. The traditional news gatekeepers were replaced by digital platforms. The distrust in social institutions is sustained by the democratizing nature of information dissemination and the monetary motivating forces of the social media channels. Falsity is profitable in the event that it is popular and further shared among various platforms. Combined with distrust, the existing inclination, and political conflict can help make echo chambers and filter bubbles, making a disagreement in the public eye. Deepfake detection is required in such cases where trueness of information is required. Detective work of such pretended videos becomes a pressing requirement for the analysis community of digital media forensics.

1.7 GENERATIVE ADVERSIAL NETWORK

The model that is emerging in recent times is the generative adversarial network (GAN). It is one of the subsets of machine learning frameworks. GAN is depicted by Ian Goodfellow and his colleagues in 2014. It consists of two neural networks competing with each other.

Given a training set, with the same statistics as the training set, this method learns to create new data. For instance, a GAN model which is trained on images can create new images that look at least obviously authentic to human onlookers, having numerous realistic characteristics [2,4]. Despite the fact that initially proposed as a type of a generative model for unsupervised learning, GANs have likewise shown value for semi-supervised learning, fully supervised learning, and reinforcement learning.

The main idea of a GAN is the root in the "indirect" training through the discriminator, [clarification needed] which itself is also being upgraded dynamically. This brings the context that the generator is not trained to minimize the distance to a specific image, but rather to bypass the discriminator. Thus the model enabled learning in an unsupervised manner.

The main purpose of the generator and discriminator is the generative network that is used to produce candidates while the discriminative network is used for evaluating them. This cycle of repetitions operates in terms of data distributions. Typically, the generative architecture learns to plot from a latent space to a data distribution of interest, while the discriminative network differentiates pristine videos created by the generator from the real data distribution. The training motive of the generative architecture is to raise the error rate of the discriminative network.

For training data for the discriminator initially, well-known datasets are used. Training involves presenting samples from the training dataset until it achieves admirable accuracy. The generator is trained until it can bypass the discriminator. In a typical generator G, the generator is seeded with the randomized input and from a predefined latent space those randomized inputs are sampled (e.g. a multivariate normal distribution). Then, candidates synthesized by the generator are tested by the discriminator. Free backpropagation techniques are applied to both networks so that the generator produces better samples, while the discriminator becomes more skilled at indicating synthetic samples. For image creation, the generator is usually a deconvolutional neural network, whereas for discriminating the created image convolutional neural network is used.

The initially projected GANs model has great significance in this, it fabricated a brand new manner of learning by producing information with the Generator and confirming it with the differentiator. However, its faults like the minimax drawback or the saddle drawback, leading to unnatural spectra within the definition and shade of generated footage. In 2016, DCGAN(Deep Convolutional Generative Adversarial Networks) proposed by Alec Radford et al. created potential arithmetic operations with filters between pictures victimization using latent vector by applying CNN (Convolutional Neural Network) models to GANs, rising a lot of clever forgeries. This development was more engineered upon in 2017 by a probe team from the University of Washington that made refined pretend videos that matched a speaker's voice and mouth in a very video and made the form of his mouth for each moment.

Through this, the previous limits of element crush, jaw form, wrinkles, etc. were greatly improved upon by applying strategies like jaw correction.

GANs often experience the ill effects of a "mode breakdown" where they can't generalize properly, eventually missing entire modes from the input data. For instance, a GAN which is trained on the MNIST dataset containing numerous samples of each digit, might yet not considering a subset of the digits from its output. The two most significant problems with the GANs are that they are difficult to train and difficult to evaluate. It is essential for the discriminator and generator to achieve Nash equilibrium during the training but, with the reference that GANs are difficult to train it is common that the generator fails to learn well about the full distribution of the datasets. This model is well-known as a collapse issue. While in the evaluation part the basic issue is how effectively to measure the dissimilarity between the real distribution of the target pr and the generated distribution pg. But accurate estimation of pr and pg is not possible. Thus, it is difficult to produce good estimations of the correspondence between pr and pg. Some researchers see the main problem to be a weak discriminative network that fails to notice the pattern of exclusion, while others point to a bad choice of the objective function. Numerous solutions have been proposed.

1.8 CNN FOR FEATURE EXTRACTION

Deep Learning (DL) is a subclass of ML techniques. The main advantage of deep learning techniques is that there is almost no need to perform feature engineering, which is to create features by hand, usually by a domain expert .This advantage emerges from the capability of performing representation learning, which means extracting knowledge from raw data. The most commonly used DL technique for image classification is the Convolutional Neural Network (CNN). Convolutional neural networks is currently one of the most prominent algorithms for deep learning with image data. Whereas for traditional machine learning relevant features have to be extracted manually, deep learning uses raw images as input to learn certain features. CNNs consist of an input and output layer, and several hidden layers between the input and output. Examples of the in between layers are convolutional layers, max-pooling layers and fully connected layers. CNN architectures vary in the number and type of layers implemented for its specific application. For continuous responses, the network should include a regression layer at the end of a network, whereas for categorical responses the system must include a classification function and layer. CNNs usually present a structured pattern of layers, combining successive convolutional (conv) layers with ReLU (rectified linear unity as activation function, sometimes seen as a layer by itself) and in the hidden layers stage the pooling layers, represented as a box . The convolution operation performed by the convolutional layer is the feature extractor, which is a filter (also known as a kernel) that slides over data, combining information. Each subsequent layer increases the abstraction level ofthe features captured by the filters. The filters in the first layer, for instance, would be responsible for detecting the presence of edges, while the second layer would be in charge of detecting the combination of edges. The third convolutional layer should be able to detect parts of familiar objects. Note that the abstraction level increases from layer to layer. The stride is nothing but sliding step distance of the filter . Usually, the convolution operation uses stride equals to one, which means that the kernel was shifted by one position. The output of this filter is the dot product from the filter elements and the data within the kernel boundaries. This operation can be interpreted as a forced merging of similar features into one single feature. To avoid overfitting, effective technique used is dropout. A dropout layer is responsible for randomly changing the output of a fraction of neurons to zero, a fraction is known as dropout rate. An input array [0.1, 0.5, 0.7, 0.3, 0.4] submitted to a dropout rate of 0.2 (20%), for example, would present one of its value set to zero. Dropout is performed only during the training stage. During testing, the output values are scaled down by a factor equal to the dropout rate in order to balance the fact that more units are being used than during the training period.

1.8.1 PADDING

The filter does not completely fit some of the input pictures. We have two options, padding the chosen picture with zeros (zero-padding) so that it fits. The part of the image can be dropped where the filter did not fit to keep the valid part only. Padding is done by enlarging the area of which a convolutional neural network processes an image. The neural networks filter which moves across the image is kernel, scanning each pixel and converting the data into a smaller, or sometimes larger, format. In order to aid the kernel with processing the image, padding is added to the frame of the image to allow for more space for the kernel to cover the image. Adding padding to a picture refined by a CNN allows for more precise examination of images

1.8.2 NON-LINEARITY

ReLU is short for Rectified Linear Unit for a non-linear operation. The output is $f(x) = \max(0,x)$. The importance of ReLU : ReLU is meant to introduce non-linearity in our ConvNet. We need non-negative linear values for our ConvNet to learn. Pooling layers section can lower the number of parameters needed when the images are too large .Spatial pooling so called subsampling or downsampling which reduces the dimensionality of each map but retains crucial information. The different types of spatial pooling:

- Max Pooling
- Average Pooling
- Sum Pooling

The largest element from the rectified feature map is taken by Max pooling. We could also take the average pooling. Sum pooling is the sum of all elements in the feature map. Recently, the activations from the convolutional layers are interpreted as local features, consequently achieving better performance than the existing hand-crafted local features in some visual tasks such as image retrieval. Typically, local features play an irreplaceable role in spatial verification because global features often will disregard spatial information. Integration of local and global features is acknowledged to deliver improvised performance for broad range image/video retrieval task, where global features are primarily utilized to rank the reference images/videos with resemblance decreasing for a given query while local features are then employed to re-rank a small set of the top matches. Local features are often aggregated into a powerful global feature with aggregation approaches like Vector of Locally Aggregated Descriptors (VLAD) and Fisher Vectors (FV). To bridge the gap between these aggregated representations and the matching techniques, AMSK and its variants, propose to combine them with a matching kernel. More recently, CNN-based or pooling approaches have also been developed. Nested invariance pooling method is introduced to derive compact deep global features. Inspired by VLAD, develop CNN-based VLAD layers which are trainable in an end-to-end manner. Zhang extracts feature for each frame of videos and constructs a visual dictionary with clustering method to represent videos with frame clusters. Moreover, deep learning-based hashing methods show remarkable power in large-scale visual search and presents a comprehensive survey of the existed deep learning-based hashing methods for mobile visual search. For video hashing, proposes a deep network to extract binary codes in a set of consecutive frames and results in multiple binary codes to represent each video. In addition, action recognition methods, often extract deep features for segments at first and then aggregate them to encode the entire video into a compact feature representation. Lately, proposes to directly extract spatial-temporal features using 3-dimensional (3D)

CNNs. It should be noted that the problem of feature coding has attracted the interest of research in the community for several years. Visual feature coding for images was proposed for different visual analysis tasks, where employed high efficiency video coding (HEVC) standard to compress features. Visual Search (CDVS) from video sequences. In our previous work, the encoding of global deep features extracted from videos was addressed for the first time by utilizing the inter-frame correlation between features. In addition, some works came up to compress the key points or the key point trajectories. In any case, how to develop a universal and extensible solution to compress both local and global deep features still remains open. Meantime, approaches are suggested to compress local and global features separately, but no work looks into their interactions to further improve the overall coding performance. In this paper, we propose an effective deep feature joint coding framework for real-valued local and global deep features (DFJC) extracted from videos, which exploits the fracture correlations to achieve better coding performance.

In CNN the image-based data is a structured topology in the regular lattice of pixels. We can directly put these weights in data by using a pre-trained model. To extract features from the CNN model first we need to train the CNN network with the last sigmoid/logistic dense layer with respect to the target variable. When the network is feed-forwarded, it directly maps the final weights for calculating the features at the intervening layer without constructing the network again. This is a type of transfer learning. Based on the need we can extract features at any intervening dense layer with the desired dimension. Generally, in machine learning, for a data, feature map is generated and a classifier is applied to this to solve the problem. For a different set of problems different and unique sets of strategies are to be applied. To overcome this disadvantage CNNs are used. In CNN features are generated automatically and classifiers are combined with them. CNN mainly advantages that CNN it has the layers that transform input volume to output volume which are simplest among all classifiers. A certain number of neurons are dropped during the forward pass and it only remembers the left out neurons in the forward pass and only updates the neurons that are not dropped during the backward pass. To avoid overfitting it makes the model learn the robust features that are independent of neurons during the training stage. This robust feature learning involves the repetition of the three layers Conv2D layer, ReLu layer and Maxpooling layer six times with varying the filter size, and reducing the number of neurons step by step. The number of neurons in the output layer is reduced and this helps in reducing the number of weights at the fully connected layer at the end. The Target of the training network is to spot the proper weights for the network by multiple forward and backward iterations, which will eventually try to minimize binary cross-entropy(misclassification cost). AUC is the evaluation metric and optimization is carried out by maximizing AUC value in each epoch.

1.9 LONG SHORT TERM MEMORY

Recurrent units from RNNs suffer from perpetuating STM. If a group of features are passed into recurrent units in multiple timesteps, RNNs fail in utilizing features from earlier time steps to later timesteps when making an informed decision about the task. LSTMs were created as an answer to the STM drawback. Long short-term memory (LSTM) is an artificial RNN architecture [37] used in the field of deep learning. LSTM has feedback connections which is not available in standard feedforward neural networks. It has the ability to not only process single data points(images) but also whole sequences of data (such as speech or video). LSTM networks are best for classifying, processing, and making predictions based on time series data since there can be delays of unknown spans between crucial events in a time series.When training traditional RNNs vanishing gradient problem occurs and to resolve these problems LSTMs emerged. The main advantage of LSTM over RNNs is the relative insensitive gap length between them. LSTMs are designed with internal gates that regulate the flow of information between timesteps[40].

The following equations describe the computations involved in an LSTM cell

$it = \sigma(xtU\ i + ht-1Wi\ )$

$ft = \sigma(xtU\ f + ht-1Wf\ )$

$ot = \sigma(xtU\ o + ht-1Wo\ )$

$c'\!t = \tanh(xtU\ g + ht-1Wg\ )$

$ct = \sigma(ft * ct-1 + it * c'\!t)$

$ht = \tanh(ct) * ot$

where i, f, and o indicate input, forget and output gates respectively. Here, c stands for the cell state and h stands for the hidden state of an LSTM unit. The core idea of a working LSTM is its cell state and therefore the operations administered by the gates. In theory, the cell state is predicted to hold relevant information down the temporal sequence of the recurrent layer. During training, the gates act as tiny neural units that decide which information is taken into account relevant. Information is taken from the previous hidden state and the current input is skilled with the forget gate ft where the sigmoid activation decides if the knowledge must be memorized or ignored. The state of the cell is modified by adjusting the input modulation gate and placing the forget gate below the cell state. From the equation, the previous cell state is forgotten by multiplying with the forget gate. The new information are added through the output of the input gates. The remember vector is also called the forget gate. The output from the forget gate tells the cell state which information to be forgotten by multiplying the position in the matrix with 0. In case output from the forget gate is 1, the information is kept in the cell state else the information is forgotten from the cell state. The input vector is also known as the save vector which determines which information should enter the cell state and which information should enter the long-term memory. At the input gate, tanh activation constrains the information between -1 and 1, thereby regulating the network. This knowledge is multiplied with the resultant forget state to ensure that only relevant information is propagated. Because the equation of the cell state is an addition between the previous cell states, the sigmoid function cannot be able to forget memory but only add memory. The reason why the input gate has an tanh activation function is used as the float number between [0,1] is never zero and thus cannot be forgotten Eventually, the output gate ot decides what the next hidden state ht should be. The hidden state is responsible for carrying the information from the previous inputs of the LSTM layer.

1.10 LONG TERM RECURRENT CNN

As human eye blinking shows sturdy temporal dependencies, we tend to use the long Recurrent Convolutional Networks (LRCN) model[41] to capture the temporal dependencies that exist. This end-to-end trainable class of architectures for visual recognition and description was created in 2016. The main plot is to combine the learning visual features of CNNs and LSTMs to transform a series of image embedding into a class label, sentence, probabilities, or whatever is needed. Thus, raw video/image is processed with a CNN and those outputs are given into a stack of recurrent sequence models. The LRCN model consists of three components, namely, feature extraction, sequence learning, and state prediction[41]. Comparing to CNNs, LSTMs are appealing because they allow end-to-end fine-tuning. The advantages of LSTMs for designing sequential data in vision issues are twofold. In the first place, when integrated with current vision systems, LSTM models are direct to fine-tune end-to-end[40]. Secondly, LSTMs are not limited to fixed-length inputs or outputs permitting simple modeling for sequential data of varying lengths, such as text or video. LRCN works by proceeding with each visual input xt to a feature transformation with variables V, generally a CNN, to create a fixed-length vector representation φV (xt)[41]. Then the outputs of φV are passed into a recurrent sequence learning module. The feature extraction module converts the input eye region into discriminative features. it is enforced with a Convolutional Neural Network based on the VGG16 framework however without fc7 and fc8 layers5. VGG16 is composed of 5 blocks of consecutive convolutional layers conv1 ∼ 5, and max-pooling operation follows every block. Finally, three fully connected layers namely fc6 ∼ fc8 are appended on the last block. The output from the feature extraction phase is fed into sequence learning, which is enforced with a Recurrent neural network (RNN) with Long Short Term Memory (LSTM)cells. The utilization of LSTM-RNN is to extend the memory capacity of the RNN and avoid the gradient vanishing in the back-propagation-through-time (BPTT) algorithmic rule within the training section. LSTMs are memory units that manage when, where, and how to forget previous hidden states and when, where, and how to update hidden states. We tend to use LSTM as, wherever σ(x) = one 1+e−x is a sigmoid operation to push input into [0, 1] range[39].

Given input Ct−1, ht−1, xt, the LSMT updates along with time t by

$ft = \sigma(Wfhht-1 + Wfxxt + bf)$

$it = \sigma(Wihht-1 + Wixxt + bi)$

$gt = \tanh(Wchht-1 + Wcxxt + bc)$

$Ct = ft\ ft + it\ \ gt$

$ot = \sigma(Wohht-1 + Woxxt + bo)$

$ht = ot\ \ \tanh(Ct)$

where ft is forgotten gate to manage what previous recollections will be discarded, it's input gate to selectively pass this input, which is manipulated by gt, ot is output gate to regulate how much memory is going to be transferred into hidden state ht. Memory cell Ct is combined with previous memory cell Ct−1 controlled by ft and input gt controlled by it. We thereby use 256 hidden units, which is the dimension of LSTM output zt. For the ultimate state prediction stage, the output of every RNN neuron is then sent to a neural network consists of a completely connected layer, that takes the output of LSTM and generates the likelihood of eye open and close state, denoted by zero and 1 respectively[40].

1.11 VGG16 MODEL

VGG16 is a convolutional neural network(CNN) model proposed by K. Simonyan and A. Zisserman from the University of Oxford. The 16 in VGG16 refers to its 16 layers that have weights. Deepfake videos are mostly produced with slight resolutions, which require an affine face warping approach such as scaling, rotation, and shearing to match the structure of the original ones. Because of the resolution mismatch between the swapped face area and the surrounding context, this process leaves traces that can be detected by CNN models such as VGG16. Here, the input to cov1 layer is of mounted size 224 x 224 RGB image. The image is gone through a group of convolutional (conv.) layers, where the filters were used with a scarce amount of receptive field: 3×3 (which is the smallest size to capture the notion of left/right, up/down, center). In one of the configurations, it conjointly utilizes 1×1 convolution filters, which may be seen as a linear transformation of the input channels (followed by non-linearity). The convolution stride is mounted to one pixel; the abstraction artifact of conv. layer input which is spatial information is preserved after convolution, i.e. the artifact is 1-pixel for 3×3 conv. layers. abstraction pooling is administered by 5 max-pooling layers, that follow a number of the conv.

layers (not all the conv. layers are followed by max-pooling). Max-pooling is performed over a 2×2 picture element window, with stride two.

Three Fully-Connected (FC) layers follow a stack of convolutional layers (which includes a completely different depth in numerous architectures): the primary two have 4096 channels each, the third performs a thousand-way ILSVRC classification and so contains 1000 channels (one for every class). The extreme layer is that the soft-max layer. The architecture of the completely connected layers is the similar in all networks.

All hidden layers are provided with the rectification (ReLU) non-linearity. It is additionally noted that none of the networks contain Local Response Normalization (LRN), and also such normalization will not improve the performance on the ILSVRC dataset, however, it will result in the increased memory consumption and computation time.

1.12 OPTICAL FLOW

Simply, an optical flow is defined as the pattern of motion caused by moving objects in a video. There are two types of extracting flow from a visual motion sequence, namely sparse flow and dense flow which is provided by OpenCV[35]. Sparse flow generates the flow vectors of the salient feature set of pixels in an image i.e., geometric center or corners of objects by utilizing Lucas-Kanade[34] whereas instance-level flow vectors were generated for the entire frame regardless of the object in focus dense flow. OpenCV computes the direction and magnitude of the flow from a 2-D array of flow vectors for the dense flow. Gunnar Farneback's algorithm [36] is used by dense flow in OpenCV.

1.13 LEVENBERG-MARQUARDT OPTIMIZATION

Levenberg-Marquardt Optimization is a degree optimization in nonlinear which remarkably surpasses the local minima and combines gradient methods for standard capacity problems. This algorithm is an iterative technique that finds the lowest of a multivariate function that is showed as the sum of squares of non-linear real-valued functions. This has become a standard procedure for non-linear least-squares problems, generally accepted in a wide spectrum of disciplines. It is a pseudo-second-order method which means that it works with only function evaluations and gradient details but it founds the Hessian matrix using the sum of outer products of the gradients Levenberg-Marquardt is a frequently used algorithm that is iterative in nature, for solve non-linear minimization problems.

The Levenberg-Marquardt curve-fitting procedure is a mixture of two minimization methods: one is the gradient descent method and the other is the Gauss-Newton method. In the former method, by updating the parameters in the steepest-descent direction, the sum of the squared errors is reduced. Whereas in the latter method, by assuming the least-squares function is locally quadratic and discovering the minimum of the quadratic the sum of the squared errors is reduced. The Levenberg-Marquardt method acts accordingly with the distance between optimal value and parameters. When the parameters are distant from their optimal value the method acts as a gradient-descent method, whereas when the parameters are too close to their optimal value the method acts as the Gauss-Newton method. Levenberg-Marquardt optimization discovers local minima beginning at an initial guess of the parameter values. In structures where there is only one minima, Levenberg-Marquard will converge to the global minimum even if the initial guess is arbitrary. In structures with multiple minima, Levenberg-Marquard is more probably to find the global minimum if the initial value is already close to the solution. However, Levenberg-Marquard permits picking the primary values to be far away from the solution than Gauss-Newton.

Advantages:

- Levenberg-Marquard is more robust than the Gauss-Newton as there are two possible choices for the algorithm's way at each iteration.
- It's faster to cluster than either the Gauss-Newton or gradient descent.
- It is capable of handling models with numerous free parameters— which are not known exactly.

If your initial value is distant from the position, the algorithm can still find an optimal solution..

1.14 HUMAN EYE BLINK FOR DEEPFAKE DETECTION

Blinking refers to the fast closing and opening movement of the eyelids. There are 3 major forms of blinking, namely Spontaneous blink, Reflex blink, and Voluntary blink. The Spontaneous blink refers to blinking without external stimuli and internal effort, which is controlled by the pre-motor brain stem and happens without any conscious effort, like respiratory functions and digestion. Spontaneous blinking serves a very important biological function that moisturizes with tears and takes away irritants from the surface of the membrane and mucosa. For a healthy adult human, generally, between every blink, there is an interval of 210 seconds however the particular rates vary by individual [44]. Humans blink iteratively and habitually daily, to keep a specific thickness of the tear film on the cornea [28]. However, eye blinks have more needs than maintaining the cornea [30], as recommended by how there is a distinction in the recurrence of eye blinks between adults and infants [29]. In particular, blinking frequency changes depending on an individual's activity. When performing actions of presented visual information such as reading aloud, the number of eye blinks increases, whereas when an individual focuses on visual information or reading silently it decreases[31]. Eye blinking which is a unique action that is repetitive and happens unconsciously provides another solution to seek out the integrity verification indices that are tough to verify using the discriminator within the GANs model[4]. If the eye blinking pattern that happens on an irregular basis is formulated and analyzed through a variety of algorithms, it may not only solely be tough to verify using a discriminator but also be extremely helpful in terms of integrity verification. The mean resting blinking rate is 17 blinks/min or 0.283 blinks per second2 (during speech activities this rate will increase to 26 blinks/min and reduces to 4.5 blinks/second and while reading this difference could also be fascinating in our analysis since several of the talking-head politicians are reading after they are being filmed)[44]. For now, however, we assume a mean rate of 17 blinks/min. The length of a blink is zero.1-0.4 seconds/blink. Notably, blinking frequency fluctuates based on a person's activity. In addition, a study by pondering in 1928 reported that once individuals conversing with each other, their eye blinks inflated. This means that eye blinks are also stricken by psychological feature activities and behavioral factors. Moreover, the range of eye blinks varies throughout the day reckoning on time: the highest number of eye blinks is typically ascertained at night around eight pm. The actual fact is that blinking frequency is suffering from a spread of things, like an individual's wellbeing, psychological feature activities, physiological factors, and knowledge process level, means that, by aggregation and statistically analyzing this data, the amount and variation of eye blinks will be foreseen to some extent. Based on this finding we can further come up with the assumption of whether the video is real or pristine.

1.15 HEAD POSE ESTIMATION FOR DEEPFAKE DETECTION

Head Pose Estimation is the prediction of the posture of a human head in an image/video. Explicitly it concentrates on the prediction of the Euler angles of a human head. The Euler angles include three values: yaw, pitch, and roll. These three values portray the rotation of an object in 3D space. By precisely anticipating these three qualities, we can sort out which direction a human head is facing. The methodology is predicated on intrinsic limitations within the deep neural network face synthesis models, which is the core part of the Deep pristine production pipeline. Specifically, these algorithms produce the faces of a unique person while keeping the facial features of the original person. However, these two faces have mismatched facial landmarks, that are locations on human faces corresponding to special structures like eye and mouth tips, as the neural network synthesis algorithmic program doesn't guarantee the original face and the synthesized face to possess consistent facial landmarks, The errors in landmark locations might not be visible directly to human eyes, however, are often discovered from head poses(i.e., head orientation and position) calculated from facial landmarks within the real and faked elements of the face. Specifically, we compare head poses estimated from all facial landmarks and those calculated solely from the central region. As a result of swapping faces within the central face region within the Deep fake method, the landmark locations of fake faces typically deviate from those of the initial faces. A landmark within the central face region P0 is first of all affine-transformed into P0 in = MP0. After the generative neural network, its corresponding landmark on the faked face in Q0 out. Because the configuration of the generative neural network in Deep fake doesn't guarantee landmark matching, and other people have totally different facial structures, this landmark Q0 out on generated face may have totally different locations to P0 in. Based on the scrutiny of fifty-one central region landmarks of 795 pairs of pictures in 64 × 64 pixels, the mean shifting of a landmark from the input of the generative neural network is 1.540 pixels, and its variance is 0.921 pixels. Once an inverse transformation Q0 = M−1Q0 out, the landmark locations Q0 within the faked faces can dissent from the corresponding landmarks P0 within the original face. However, because of the actual fact that Deep fake solely swap faces within the central face region, the locations of the landmarks on the outer contour of the face. This mismatch between the landmarks at the center and outer contour of faked faces is unconcealed as inconsistent 3D head poses calculable from central and whole facial landmarks. The rationale is that the two calculated head poses are close for the real face, except for a Deepfake, as the central face region is from the synthesized face, the errors due to the mismatch of landmark locations from original and generated pictures can cause a bigger difference between the two calculated head poses. We experimentally make sure the numerous differences within the estimated head pose in Deep Fakes. For simplicity, we glance at the top orientation vector solely. Then we have a tendency to use the differences in estimated head pose as a feature vector to coach a straightforward SVM-based classifier to differentiate original and Deep Fakes. Experiments on realistic Deep fake videos demonstrate the effectiveness of our algorithmic program.

1.16 ORGANIZATION OF THESIS

The rest of the thesis is organized as follows. Chapter 2 presents the literature survey on video feature extraction by various methods. Chapter 3 describes the modules that are to be implemented, the proposed system architecture, and its corresponding explanation. Chapter 4 discusses the evaluation of the test input and sample input and output. Chapter 5 presents the conclusion of the project and some possible avenues for future research on the topic.

CHAPTER 2

LITERATURE SURVEY

2.1 TYPES OF FACE SYNTHESIS

• **Entire Face Synthesis:** This creates entire non-existent face images using powerful GAN, e.g., through the recent StyleGAN approach proposed in [45]. These techniques achieved amazing results, generating high-quality facial images with high reality. This type of synthesis is much used in sectors such as the video game and 3D-modelling industries, but on the other hand it has much harmful applications such as the generation of fake profiles in social networks with high level of realism in order to generate misinformation.

• **Identity Swap:** This is created by replacing the face of one person in a video with the face of another person. Two different approaches are usually used for creating this manipulation: i) the basic computer graphics related techniques such as FaceSwap6, and ii) novel deep learning techniques known as DeepFakes7 , e.g., the recent ZAO mobile application.This type of manipulation also creates very realistic videos and these type of videos are available in YouTube. This type of synthesis can be used in different sectors, particularly in the film industry. On the other side, it could also be mis-used for wrong motives such as the creation of celebrity pornographic videos, financial fraud and so on.

• **Attribute Manipulation:** This manipulation is also known as face editing or face retouching. This type of manipulation is created by modifying some attributes of the face such as the colour of the hair or the skin, the gender, the age, adding glasses, etc [46]. This synthesis activity is usually carried out by GAN such as the StarGAN approach proposed in [47]. One big example of this type of manipulation is FaceApp mobile application. This can be used for trying broad range of products such as cosmetics and makeup, glasses, or hairstyles in a virtual environment.

• **Expression Swap:** This manipulation is also known as face re-enactment. It creates manipulated video/photo by modifying the facial expression of the person. Although there are different manipulation techniques at image level through popular GAN architectures [48] in this group we focus on the most popular techniques Face2Face and NeuralTextures [49], [50], which replaces the facial expression of one person in a video with the facial expression of another person. This type of manipulation could be used with serious consequences by using them to make popular video of Obama saying things he never said[38].

2.2 DEEPFAKE GENERATION TECHNIQUES

Deepfake creation is a somewhat new field in digital forgery. Realistic images and videos have been created by utilizing detailed 3D computer graphics models. Firstly, we will outline the traditional approaches, and then the adversarial methods. Generally, there are two types of deepfake generation, face replacement and face re-enactment. The face of a target person is overlaid on the face of a source in face replacement. The over-laid faces are post-processed to blend the edges such that they match the source's facial outline. To match the source's facial outline, the edges of the overlaid faces are blended. This post-process can be used, for instance, to make any person (the source actor) appear in a movie in place of the original (target) actor.1 Faceswaps are a graphical commence, where the facial landmarks such as nose, eyes, eyebrows, lips, chin, and cheek areas play a vital role in altering the target's face with the source's, and the output is post-processed with edge blending and color refinement. On the other hand, Facial re-enactment is the method that is used to make a target seem to act and speak like the source. This is used, for instance in the videos where former U.S. President Barack Obama is made to enact whatever the source video says. Facial re-enactment techniques model both the source and target faces to identify facial landmarks and manipulate the target's landmarks to match the source's facial movements. Face2Face is a face reenactment method that converts the facial expressions of a source subject with a target while retaining the facial features of the target. Although The generation of Deepfakes began with the process of traditional vision and voice impersonation, most recent works involve generative adversarial networks (GANs). Goodfellow et al. [2] first proposed generative adversarial networks (GANs), which commonly comprise two networks: generator and discriminator. The generator intends to create a sample that ought not to be differentiated from training data distribution, while the discriminator is to evaluate the sample produced by the generator. These two networks are trained concurrently, such that both the generator and discriminator will improve over the training period. Upon successful unification of the generator and discriminator, the generator can be used to produce realistic-looking examples. The generator G is seeded with a noise vector to advance variation, but this noise vector can be coupled with a latent representation of an object (word, image, sentence), to drive the resulting image. Zhu et al. and Kim et al. built on the concept of GAN and the noise vector was replaced with input images. This alteration enabled their cycle-consistent GANs to modify the domains of the output images based on the domains of the input image. Denton et al [3] proposed a Laplacian pyramid GAN to create images in a coarse-to-fine fashion. Radford et al. [4] proposed Deep Convolutional GANs (DCGAN) and showed the ability for unsupervised learning. Arjovsky et al. [5] utilized Wasserstein distance to make training stable. Isola et al. [6] explored conditional adversarial networks to take in mapping from the input image to output image in addition to the loss function to train the mapping. Taigman et al. [7] proposed the Domain Transfer Network (DTN) to map a sample from one domain to another domain in the analog sample and accomplished ideal execution on less resolution face and digit images. Shrivastava et al. [8] diminished the gap between synthetic and real image distribution utilizing the fusion of adversarial loss and self-regularization loss. Liu et al. [1] proposed an unsupervised image-to-image translation framework based on Coupled GANs, with the intent to learn the joint distribution of images in various areas. In the deepfake generation, it is possible to preserve the facial expressions of a source person while moving identities to a target person. Lu et al. proposed an identity-guided conditional CycleGAN for turning low-resolution face images into high-resolution face images. A similar transfer was achieved by Kim et al., with the contrast of transferring expressions as well as 3D pose into a target image, creating a video in the process. Making use of adversarial and perceptual losses realistic-looking imagery was created by Faceswap-GAN. The inclusion of a perceptual loss was shown to lessen unnatural artifacts such as awkward eyeball movements. The created videos appear more realistic due to the temporal blending of the frame-to-frame face detection box and an attention mask. Siarohin et al. introduced a learnable optical flow network estimating it to a first-order Taylor polynomial. Their method facilitated a few-shot capability to produce a manipulated video of a person using a single image. Li et al. discovered an adaptive-attention-based denormalization generator for high-quality face replacement using a novel learning metric. In an intriguing turn, Suwajanakorn et al. accomplished photorealistic results by just requiring audio as an input to generate fake videos. The weekly addresses of Barack Obama are utilized where the mouth shapes are first mapped to raw audio features and then mouth textures, 15 then 3D pose mapping, and finally compositing of the head and middle from stock video.

2.3 DEEPFAKE DETECTION TECHNIQUES

Since the threats caused by deepfake video manipulations became evident in early 2018, various works have looked into detecting them. A few of the detection approaches targeted handcrafted characteristics like blinking inconsistencies, biological signals, and unrealistic details. The greater part of these hand-crafted detection features exploits known weaknesses in creation methods. Like a cat-and-mouse game, deepfake formation methods are rapidly adapted to bypass detection, and the cycle repeats. Recent detection methods rely on machine learning on deepfake datasets to automatically discover forgeries from real videos. MesoNet uses a shallow convolutional network to distinguish forgery at a mesoscopic level of detail, deliberately avoiding focusing too much on tiny features that could be lost due to video compression. They also presented a variation of their model that replaces standard convolution blocks with MesoInception blocks to get slight enhancements. The Capsule-Forensics method proposed by Nguyen et al. uses capsule networks for the detection of replay attacks as well as computer-generated images and videos. They contend that the odds of distinguishing high-quality forgeries would be increased with the accordance between capsules through dynamic routing. Cozzolino et al. applied an autoencoder-based design to show its effectiveness for transfer learning. NNguyen et al. extended Cozzolino et al.'s network by putting back the decoder that additionally generates a mask of the manipulated region through multi-task learning in the place of the standard decoder. Detecting eye blinking has been studied in computer visions for applications in fatigue detection [9]–[13] and face spoof detection [14]–[18], and different strategies have been proposed to tackle this problem. Pan et al. [19] build undirected conditional irregular field structure to infer eye closeness such that eye blinking is recognized. Sukno et al. [20] utilizes Active Shape Models with the Constant Optimal Characteristics to illustrate the outline of eyes and calculates the eye vertical distance to decide eye state. Torricelli et al. [21] use the variation between consecutive frames to break down the state of eyes. Divjak et al. [22] utilize optical flow to acquire eye movement and extract the prevailing vertical eye movement for blinking analysis. Yang et al. [23] models the shape of eyes based on a set of parameterized parabolic curves, and fit the model in each frame to track eyelid. Drutarovsky et al. [24] break down the difference of the vertical motions of the eye region which is detected by a ViolaJones type algorithm. At that point, a group of KLT trackers is utilized on the eye region. Each eye region is partitioned into 3x3 cells and an average movement in every cell is computed. Soukupova et al. [25] completely relies on facial landmarks and put forward a single scalar value – eye aspect ratio (EAR), to define the eye state in each frame. At that point, an SVM is trained to utilize EAR values within a brief timeframe window to classify the final eye state. Kim et al [26] contemplate CNN-based classifiers to detect the state of the eye whether the eye is in the opened or closed state. They take on ResNet-50 [27] model and contrast the performance with AlexNet and GoogleNet.

2.4 SUMMARY OF LITERATURE SURVEY

The activations from the convolutional layers in convolutional neural networks can be treated as local deep features describing particular details inside an image region, which are then aggregated as a powerful global descriptor. The aggregated global features should be lossy and consequently would degrade the overall performance. The joint coding of local and global features needs further investigation.

CHAPTER 3
SYSTEM ARCHITECTURE AND DESIGN

3.1 SYSTEM ARCHITECTURE

Our architecture is based on performing integrity verification through tracking significant changes in the eye blinking pattern of a subject along with head pose estimated in the video. We are detecting faces in each frame of the video and regions corresponding to each eye are extracted out to form a stable sequence. Features of mouth and nose are detected for performing head pose estimation. After these pre-processing steps, eye blinking is detected by quantifying the degree of openness of an eye in each frame by sequence learning. Head poses are estimated using the difference in the facial landmarks from the whole face and those only from the central face region. The alignment error is revealed as differences within the head poses. This difference of the head poses and eye blinking pattern is then fed to an SVM classifier to differentiate the original one from the Deep Fake. The system architecture is shown in **Figure 3.1**

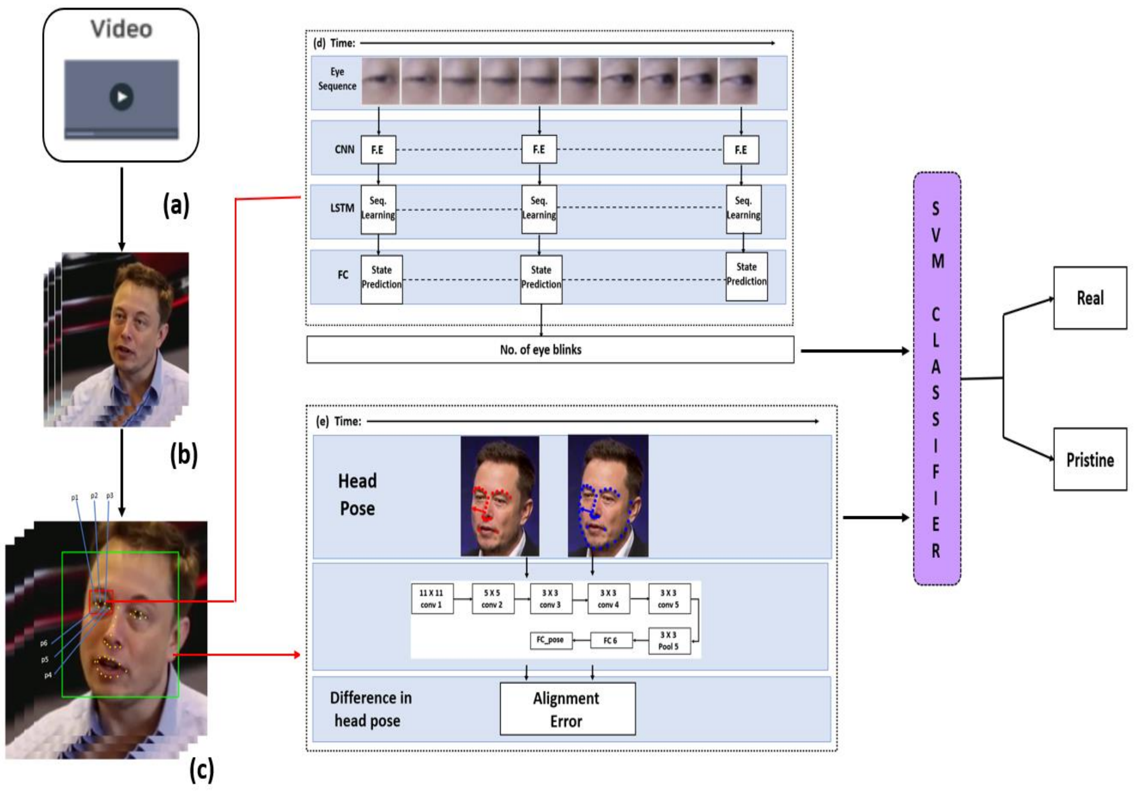


Figure 3.1 – System Architecture

3.2 PREPROCESSING

3.2.1 FACIAL FEATURES DETECTION:

The HyperFace algorithm is used for detecting the face region in the given image. In the Pre-Process, it performs data type conversion, image resizing, and normalization[43]. The first module generates region -proposals that are independent of class from the given image and scales the images to 227 × 227 pixels. This is passed through the second module of HyperFace which is CNN. This module makes use of the candidate regions that are resized in the previous module and classifies them whether they are face or non-face regions. In case if the region is gets categorized as the face, then the bounded box is drawn across the face and then passed through the next module for extracting individual face features. The face features are plotted using 68 feature points which detect include the left and right eyes, left and right eyebrows, the nose, the mouth, and the Jawline of the face. Here for this method we require only face regions such as eyes and nose. The particular features from the face are extracted by splicing the array that contains those features. In R-CNN, the Selective Search algorithm is utilized for creating face region proposals in an image. If the Intersection over Union (IOU) overlap with the candidate region is more than 0.5 then it is a positive sample and it contains a face. If the IOU overlap with the candidate region is less than 0.35 then it is a negative sample and other regions are omitted. The softmax loss function is used for training the face detection task.

3.2.2 EYE BLINK DETECTION:

The Eye Tracker was implemented based on AR (Aspect-Ratio). AR takes six points(pi) round the eyes and calculates absolutely the area of the horizontal axis and vertical axis. In general, eye blinks occur at the same time in both eyes. So equation has been proposed that sums and divides in half the eye's ratio (ARi) through using the value of the left eye (ARl) and right eye (ARr)[44]. When eyes are closed, the ARi is reduced and drops below the threshold in the consecutive video frames. Then, when the eyes are open, the ARi is restored as before. That is when the eye is in the open state the aspect ratio of the eye is found to be a constant value, but the value abruptly falls approximately to 0 when the eye is in the closed state. In this process, the time required to blink, and blinking frequency is obtained. But the main drawback of the aspect ratio method is they are dependent on the eye landmarks found. Also in many cases, it is found that eye landmarks are not reliable as in some frames the eye area is too small to find the correct aspect ratio value which perplexed finding the eye state. By only depending on the image domain the eye state cannot be determined correctly. By using the temporal domain, the states of the eye can be memorized. Current strategies use Convolutional Neural Networks (CNN) as a binary classifier to differentiate the open and close eye state of every frame.

However, CNN generates predictions depending on a single frame, that doesn't leverage the information in the temporal domain. As human eye blinking has a powerful temporal correlation with previous states, we tend to use Long-Recurrent Convolutional Neural Networks (LRCN) that distinguish open and closed eye state with the thought of previous temporal information[41]. The Eye tracker algorithm is improved for detecting the eye blinking pattern by making the algorithm incorporate the temporal relationship between consecutive frames which memorizes the previous state to produce output based on sequence learning. If the eye blink has occurred in the previous frames, then the state of the eye in the next set of frames is likely to be in the open state[42]. If the eye has been in the open state in previous frames, the state of the eye is most likely to be the open state for the next frames. This improvisation in the algorithm would result in less time in detecting the state with better accuracy.

3.2.3 ALIGNMENT ERROR CLASSIFICATION:

It is well-known that Deep Fake takes only the central face region and swaps faces whereas the outer contour of the face will remain the same. This inconsistent match between the landmarks at the center region and the outer face region of faked faces is an alignment error. We consider the head orientation vector and the difference between the headpose of the central face region and the whole face region are small in real images, but large in fake images. Ra is denoted as the rotation matrix calculated using facial landmarks from the entire face, Rc as the one calculated solely using landmarks within the central region. we tend to acquire the 3D unit vectors Ta and Tc equivalent to the orientations of the head calculated. Then cosine distance between the 2 unit vectors Ta and Tc is compared, which takes value in [0, 2] with zero which means the 2 vectors accept as true with one another. The smaller this value is, the nearer the 2 vectors are to every alternative. The cosine distances of the 2 estimated head pose vectors for the real pictures concentrate on a considerably smaller range of values up to 0.02, whereas for Deep Fakes the majority of the values are within the range between 0.02 and 0.08. The variation within the distribution of the cosine distances of the 2 head orientation vectors for real and Deep Fakes counsel that they can be differentiated based on this cue.

3.3 POSE ESTIMATION

Head pose Estimation research in computer vision focuses on the prediction of the pose of an individual's head in a picture. In computer vision, the pose of an object refers to its relative direction and location with respect to a camera. The pose can be altered by either moving the object with respect to the camera or the camera with respect to the object. Specifically, it considers the prediction of the Euler angles of an individual's head. The Euler angles consist of 3 values: yaw, pitch, and roll. These 3 values describe the rotation of an object in a 3D area. By accurately predicting these 3 values, we can understand the direction an individual's head is facing. The 3D head pose relates to the rotation and translation of the world coordinates to the camera coordinates. In particular, [U, V, W] T be the world coordinates of one facial landmark, [X, Y, Z] T be its camera coordinates, and (x, y) T be its image coordinates. The change between the world and the camera coordinate systems can be figured.

In 3D head pose assessment, we need to tackle the reverse problem of estimating s, R, and ~t utilizing the 2D image coordinates and 3D world coordinates of the same set of facial landmarks got from a standard model, e.g, a 3D average face model, expecting we know the camera boundary. In particular, for a group of n facial landmark points, this can be defined as an improvement issue that can be settled efficiently using the Levenberg-Marquardt algorithm. The assessed R is the camera pose which is the rotation of the camera with regards to the world coordinate, and the head pose is acquired by reversing it as RT (as R is an orthonormal matrix). Having a computer helps us to understand the direction an individual's head is facing and this provides several helpful applications. For example, it will be accustomed to plot a 3D object to join the direction of the head almost like those seen in TikTok, Snapchat, and Instagram filters. Additionally, it may be utilized in self-driving cars to trace whether or not a driver is that specialized in driving.

3.4 SUPPORT VECTOR MACHINE CLASSIFIER

The support vector machine algorithm aims to find a plane in an N-dimensional space for classifying the points separately by drawing a plane between the classes of data points. SVM can be classified into two types:

Linear SVM: It is used for linearly separable data. If data in the dataset can be grouped into two classes by using a single straight line, then these data are called linearly separable data and they are classified using a Linear SVM classifier.

Non-linear SVM: It is used for non-linearly separated data. If data in the dataset cannot be grouped by using a straight line, then these data are called non-linear data and they are classified using a Non-linear SVM classifier.

To separate the two classes of data points, numerous possible hyperplanes could be drawn between the two classes of data points to separate them. But the goal of SVM is to find a plane that has the maximum distance between data points of both classes. Maximizing the margin distance provides, gives confidence that that future data points can also be classified correctly. Hyperplanes are decision boundaries that help to characterize the data points. Data points lying on either side of the hyperplane can be assigned to different classes. Also, the dimension of the hyperplane relies on the quantity of features. The hyperplane is a line in the case where the number of input features is 2, hyperplane becomes a two-dimensional plane when the number of input features is 3. If the value exceeds 3 it becomes difficult to envision the hyperplane. Support vectors are data points that are nearer to the hyperplane and they impact the position and direction of the hyperplane. Utilizing these support vectors, we expand the margin of the classifier. Removing the support vectors will affect the position of the hyperplane. These are the points that help us generate our SVM. In logistic regression, we take the output of the linear function and by using the sigmoid function we squeeze the value within the range of [0,1]. If the value is larger than a threshold value (0.5) we set it a label 1, else we set it a label 0. The outcome of the linear function is taken. In the event that the output is larger than 1, we recognize it with one class and if the output is -1, we recognize it with another class. The support values([-1,1]) acts as a margin as the threshold values are changed between -1 and 1. Here the alignment error of head pose and eye blinking data are passed to the SVM classifier as feature Vector which detects whether the given video is deepfake or real. These features are flattened into a vector, which is standardized by subtracting its mean to predict the desired result of deepfake or real.

The features are extracted in the following procedures:

(1) For every image or video frame, we tend to run a face detector and extract sixty-eight facial landmarks using code package DLib.

(2) Then, with the standard 3D facial landmark models of a similar sixty-eight points from OpenFace2, the top poses from the central face region (Rc and tc) and whole face (Ra and ta) area unit are calculated with landmarks 18 − 36, 49, 55 and 1 − 36, 49, 55. Here, we tend to approximate the camera focal distance because of the image breadth, camera center as image center, and ignore the result of lens distortion.

(3) The differences between the obtained rotation matrices (Ra −Rc) and translation vectors (~ta − ~tc), eye blinks are normalized into a vector, which is standardized by subtracting its mean and divided by its variance for classification.

CHAPTER 4

ALGORITHM DEVELOPMENT AND IMPLEMENTATION

4.1 THE TARGET DETECTOR:

```
Input: Fast-HyperFace(model) hf
Loop(frames):
frame ← Pre_Process(frame) # astype, resize,
landmarks, detections ← hf (frame)
Feature_point← Slice_array
Define range of Features
eye_feature,facial_features← Slice_array(range of eye,nose tip,mouth tip)
Forward_to_EyeTracker(landmarks, eye_feature)
Forward_to_HeadPoseTracker(landmarks,facial_features)
```

Explanation:

Step1: Frames are extracted from video dataset and sent in loops to the model.

Step2: Frames are preprocessed and normalized to obtain localized face.

Step3: Feature points of localized face is analysed and feature points of eye,nose tip,mouth tips ares defined.

Step4: Range of features are sliced in array of points.

Step5: sliced eye region and localized face are sent to eye tracker phase.

Step 6: Detected facial features are sent to head pose tracker phase.

4.2 THE EYE TRACKER

```
Input: landmarks,cropped eye area sequences
output: eye blinking rate,state prediction
def Track_Tempdependency(eye):
Feature extraction(CNN)
Discriminative features <--- eye regions
return (eye features)
h_axis ← dist.euclidean(p [1], p [4])
v_axis1 ← dist.euclidean(p [2], p [6])
v_axis2 ← dist.euclidean(p [3], p [5])
return (v_axis1 + v_axis2) / (2 ∗ h_axis)
LeftAR ← Track_AR(l)
RightAR ← Track_AR(r)
AR ← (LeftAR + RightAR) / 2
logs ← logging(frame, AR, time_capture())
Eye_open(0)/eye_close(1)🡐 CNN,AR
Temporal sequence analysis🡐 LSTM(AR)
State prediction
Eyeblink_rate🡐 fc_layer
```

Explanation:

Step1: Features are extracted from eye by utilizing CNN.

Step2: Aspect ratio of eye is found by calculating degree of openness of eye.

Step3: Threshold value is defined by analysing normal person's eye blinking rate.

Step4: Based on Aspect ratio,eye state in each frame is predicted.

Step5: Aspect ratio is passed as x co-ordinate to the sequence learning phase that detects the number of eye blinks in the video input.

4.3 THE HEAD POSE ESTIMATOR

Input : 2D facial landmarks

Output : Alignment error

Rotation_Matrix ($R_a = P_0 + P_1$)$\leftarrow$ 2D facial features of whole face

Rotation_Matrix ($R_c = P_0$)$\leftarrow$ 2D facial features of central region

Def(3D unit vectors v_a and v_c),compare cosine distance

Head_pose of central region $\leftarrow$ R_c,T_c

Head_pose of whole face $\leftarrow R_a,T_a$

Alignment_error $\leftarrow$ diff in cosine distance of (R_a - R_c)and (T_a – T_c).

Explanation:

Step1: Rotation_Matrix (R_a) is obtained from the 2D facial features of whole face.

Step2: Rotation_Matrix (R_c) is obtained from the 2D facial features of central region.

Step3: 3D unit vectors v_a and v_c are obtained to compare cosine distance.

Step4: Head_pose of central region corresponds to R_c,T_c .

Step5: Head_pose of whole face corresponds to R_a,T_a .

Step6: Alignment error is calculated by finding difference of cosine distance estimated from whole face (R_a - R_c) and head pose estimated from central face region(T_a – T_c).

4.4 IMPLEMENTATION ENVIRONMENT

Python programming language is employed for deploying this model. PyCharm, Ubuntu programming setting is employed for implementation purposes. UADFV and Celeb-DF datasets are employed for training and evaluating deepfake detection. The video stream is captured and therefore the image process techniques are executed with the assistance of the OpenCV library in python.

4.4.1 UBUNTU

Ubuntu is a Linux distribution based on Debian and composed largely of free and open-source software. Ubuntu was formally released in three editions namely Desktop edition, Server edition, and Core for IoT devices and robots edition. All three editions will run on the pc alone, or in a virtual environment. Following mentioned are some of the compelling features of Ubuntu −

- The normal software such as Firefox, Chrome, VLC which are available in windows, is supported in the desktop versions of Ubuntu.
- It has a built-in software call Thunderbird to access the emails.
- It has free applications for users to view and edit photos and also has applications to manage and share videos.
- It has a smart searching facility with which can search the contents.
- It is a free operating system backed by a huge open-source community.

4.4.2 PYCHARM

PyCharm is an integrated development environment utilized in programming, specifically for Python programming. It is developed by the Czech company JetBrains. It provides code analysis, a graphical computer program, an associate integrated unit tester, integration with version management systems (VCSes), and supports web development with Django moreover as information Science with anaconda. PyCharm is cross-platform, with Windows, macOS, and UNIX system versions available. The Community Edition is discharged under the Apache License, and there is an additionally skilled Edition with additional options – discharged under a proprietary license. The features of PyCharm includes :

- PyCharm provides coding assistance and analysis along with features such as code completion, highlighting syntax and error, linter integration, and quick fixes
- Project and code navigation is made easy with PyCharm especially the featured project views, file structure views, and quick jumping in between files, classes, methods, and usages.
- It includes Python refactoring such as renaming a file, extracting method, introducing variable, introducing constant, pull up, push down, and others.
- The Professional edition of PyCharm supports web frameworks such as Django, web2py, and Flask.
- It has an Integrated Python debugger within which has integrated unit testing, that covers code line-by-line.

Professional edition of PyCharm supports scientific tools such as matplotlib, NumPy and scipy.

4.4.3KERAS

Keras is an open-source computer code library that facilitates a Python interface for artificial neural networks. Keras acts as a common interface for the TensorFlow library. Keras contains varied implementations of ordinarily used neural-network building blocks like layers, objectives, activation functions, optimizers, and a bunch of tools operating with image and text information easier to alter the coding necessary for writing deep neural network code. Additionally, besides standard neural networks, Keras has support for convolutional and perennial neural networks. It supports different common utility layers like dropout, batch standardization, and pooling. Keras lets users to productize deep models on smartphones, on the web, or the Java Virtual Machine(JVM). It additionally permits the use of distributed training of deep-learning models on clusters of Graphics process units (GPU) and tensor process units (TPU).

It supports the following features −

- Consistent, simple, and extensible API.
- It supports multiple platforms and backends.
- It is a user-friendly framework and it runs on both CPU and GPU.
- Highly scalability of computation
- It has Larger community support.
- Easy to test.
- It supports both convolution and recurrent networks.
- Since Deep learning models are discrete components they can be combined in many ways.

4.4.4TENSORFLOW

TensorFlow is a free and open-source software package library for machine learning. It was initially evolved by Google's Machine Intelligence research organization for deep neural networks research and machine learning. It is used across a variety of tasks however encompasses an explicit focus on training and logical thinking of deep neural networks. TensorFlow is a symbolic mathematical library based on dataflow and differentiable programming. TensorFlow is cross-platform. It runs GPUs and CPUs—even mobile and embedded platforms—including tensor processing units (TPUs).

TensorFlow has lots of advantages:

- It is portable. The graph can be saved to use later or even can be executed immediately. They can run on multiple platforms such as CPUs, GPUs, and TPUs. Likewise, it can be employed in production without having to rely on any of the code that constructed the graph, only the runtime essential to execute it.
- It's transformable and optimizable, as the graph can be reconstructed to produce a more suitable version for a given platform. Also, memory or compute developments can be performed and trade-offs made between them. This is helpful in supporting quicker mobile inference after training on larger machines.
- Support for distributed execution
- TensorFlow's noble APIs, in coexistence with computation graphs, enable a rich and workable development environment and powerful building capabilities in the same framework.

CHAPTER 5
RESULTS AND DISCUSSIONS

5.1. DATASET FOR DEEPFAKE DETECTION METHODS

UADFV[33] dataset contains total of 98 videos. 49 real videos from YouTube and 49 fake ones generated by FakeAPP .

Celeb-DF(v2)[32] dataset contains real and synthesized videos. The Celeb-DF (v2) dataset is greatly extended from Celeb-DF (v1), which contains 795 DeepFake videos.Celeb-DF includes 590 original videos collected from YouTube with subjects of different ages, ethic groups and genders, and 5639 corresponding DeepFake videos.

5.2. BALANCING AND SPLITTING DATASETS

As we noticed that the number of fakes are much greater than the number of real faces (due to the fact that one real video is used for creating multiple deepfake videos), it is essential to perform a down-sampling on the fake dataset in accordance with the number of real crops for managing for possible class misproportion issues during the training phase.

It is also necessary to split the dataset into training, validation and testing set in the ratio of 80:10:10 as the final step in the data preparation phase.

5.3. EYEBLINKING EVALUATION METRICS

The number of eye blinks of the video is found by incorporating the temporal relationship between consecutive frames which memorizes the previous state to produce output in the basis of sequence learning. To evaluate the videos using the number of eye blinks we will be using the following metrics and is shown in **Table 5.1**

If Eye Blinking Rate is 4-6 blinks, then it is a real video

If Eye Blinking Rate is 0-2 blinks, then it is a fake video

File Category	Average video Length	Eye blinking rate
Real videos	10-11 sec	4-6 blinks
Fake videos	10-11 sec	0-2 blinks

Table 5.1 EyeBlinking Evaluation metrics

5.4. HEADPOSE EVALUATION METRICS

The headpose is estimated and cosine difference between the whole face and central region are found and taken as alignment error. To evaluate the videos using the alignment error we will be using the following metrics and is shown in **Table 5.2**

If alignment error value <0.02 then it is a real video

If alignment error value 0.02-0.08 then it is a fake video

File Category	Average video length	Alignment error
Real videos	10-11 sec	<0.02
Fake videos	10-11 sec	0.02-0.08

Table 5.2 Headpose Evaluation Metrics

5.5. OVERALL EVALUATION METRICS

Alignment error of head pose and eye blinking data are passed to SVM classifier as feature Vector which detects whether the given video is deepfake or real. The output metrics shown in **Table 5.3** :

Eye blink rate	Alignment error(AE)	Probability	Classification
4-6	AE < 0.02	0.5<=x<=1	Real
0-2	0.02 <= AE AE <=0.08	0<=x<=0.5	Pristine

Table 5.3 Evaluation Metrics

5.6. PERFORMANCE EVALUATION

Performance evaluation is nothing but performance measurement (PM). It is complex yet an important aspect of the machine learning process. The three main subtasks on which performance evaluation focuses are: to measure performance, resample the data, and assess the statistical significance of the results. Performance evaluation is the task of determining the efficiency and effectiveness of project construction activities. The following performance evaluation metrics are used in our deepfake classifier.

PRECISION

It is the fraction of positive occurrences out of the total predicted positive occurrences. Precession in simple words can be defined as finding out 'how much the model is right when it says it is right'. The precision is expressed as shown in **Figure 5.1.**

$$\frac{TP}{TP + FP}$$

Figure 5.1 Precision

RECALL

Recall is the fraction of positive occurrences out of the total actual positive occurrences. The denominator (TP + FN) here is the exact number of positive occurrences present in the total dataset. In simple terms recall can be described as 'how much extra right ones, the model missed when it showed the right ones'. The recall is expressed as shown in **Figure 5.2.**

$$\frac{TP}{TP + FN}$$

Figure 5.2 Recall

F1-SCORE

F-score, also known as the F1-score, is used for calculating the accuracy of model on a dataset. It is used to estimate binary classification systems, which classify examples into 'positive' or 'negative'. The F1-score is a way of combining the precision and recall of the model using harmonic mean can be technically described as harmonic mean of the precision and recall of the model and shown in **Figure 5.3**.

$$\frac{2}{\frac{1}{precision}+\frac{1}{recall}} = \frac{2 * precision * recall}{precision + recall}$$

Figure 5.3 F1-score

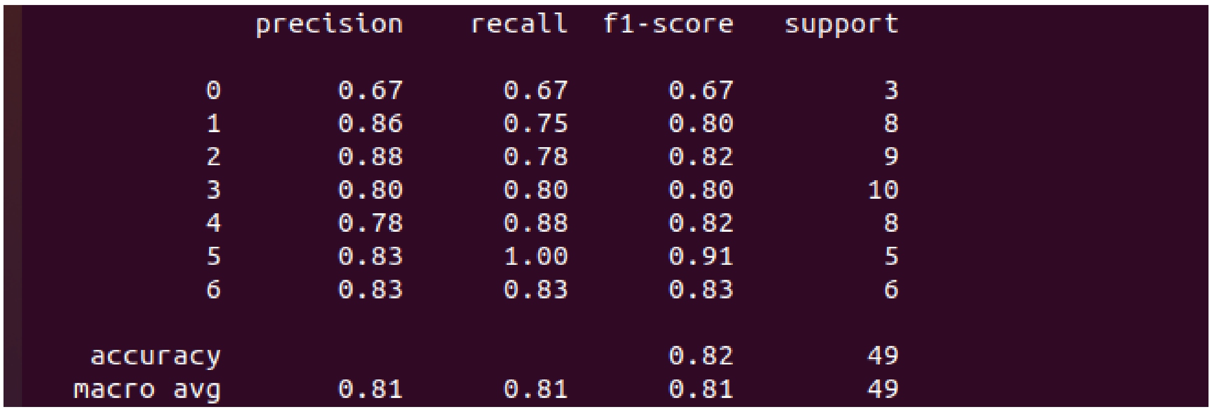
True positives (TP): Actual positive value and is predicted as positive.

False positives (FP): Actual negative value and is predicted as positive.

True negatives (TN): Actual negative value and is predicted as negative.

False negatives (FN): Actual positive valu and is predicted as negative.

Performance of our deepfake classifier is evaluated by testing videos from our dataset and following performance has been observed in **Table 5.4**:



```
              precision    recall  f1-score   support

           0       0.67      0.67      0.67         3
           1       0.86      0.75      0.80         8
           2       0.88      0.78      0.82         9
           3       0.80      0.80      0.80        10
           4       0.78      0.88      0.82         8
           5       0.83      1.00      0.91         5
           6       0.83      0.83      0.83         6

    accuracy                           0.82        49
   macro avg       0.81      0.81      0.81        49
```

Table 5.4 Performance Observation

It is observed that accuracy of our Deepfake classifier comes as 0.82 and the macro average is obtained as 0.81.

CHAPTER 6

CONCLUSION AND FUTURE WORK

6.1 CONCLUSION

The new advancement in deep generative networks has remarkably improved the quality and efficiency in generating realistically-looking deepfake videos. s. In this work, we express a new method to expose fake face videos which are generated with neural networks.This method is based on the detection of eye blinking which is a spontaneous and unconscious human reflex function along with headpose estimation in the videos, which is a physiological signal that is not well presented in the synthesized fake videos that can be used as an approach to detect the Deepfakes. This work can be improved through several measures because cyber-security attacks and defense evolve continuously. Firstly, we will explore other deep neural network architectures for more effective methods to detect closed eyes and roll, yaw, the pitch of head pose instead of pickle file, and also exploring other types of physiological signals to detect fake videos.

6.2 FUTURE WORK

The proposed system was implemented for stored videos. As future work, the proposed system can be scaled up to support live video or streaming video from an IP camera for a live feed. Also, we can explore other deep neural network architectures for more effective methods to detect closed eyes and roll, yaw, the pitch of head pose. Using action units derived from the facial landmark movement between the frames to distinguish natural facial motion with that of computer-generated. Manually extracting multiple degrees of motion and analyzing its potential as a feature set. In the long run, we are concerned with exploring other types of physiological signals to detect fake videos.

REFERENCES

[1] M.-Y. Liu, T. Breuel, and J. Kautz, "Unsupervised image-to-image translation networks," in NIPS, 2017, pp. 700–708.

[2] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial nets," in Advances in neural information processing systems, 2014, pp. 2672– 2680.

[3] E. L. Denton, S. Chintala, R. Fergus et al., "Deep generative image models using a laplacian pyramid of adversarial networks," in Advances in neural information processing systems, 2015, pp. 1486–1494.

[4] A. Radford, L. Metz, and S. Chintala, "Unsupervised representation learning with deep convolutional generative adversarial networks," arXiv preprint arXiv:1511.06434, 2015.

[5] M. Arjovsky, S. Chintala, and L. Bottou, "Wasserstein gan," arXiv preprint arXiv:1701.07875, 2017.

[6] P. Isola, J.-Y. Zhu, T. Zhou, and A. A. Efros, "Image-to-image translation with conditional adversarial networks," arXiv preprint, 2017.

[7] Y. Taigman, A. Polyak, and L. Wolf, "Unsupervised cross-domain image generation," arXiv preprint arXiv:1611.02200, 2016.

[8] A. Shrivastava, T. Pfister, O. Tuzel, J. Susskind, W. Wang, and R. Webb, "Learning from simulated and unsupervised images through adversarial training," in CVPR, vol. 3, no. 4, 2017, p. 6.

[9] W.-B. Horng, C.-Y. Chen, Y. Chang, and C.-H. Fan, "Driver fatigue detection based on eye tracking and dynamk, template matching," in Networking, Sensing and Control, 2004 IEEE International Conference on, vol. 1. IEEE, 2004, pp. 7–12.

[10] Q. Wang, J. Yang, M. Ren, and Y. Zheng, "Driver fatigue detection: a survey," in Intelligent Control and Automation, 2006. WCICA 2006. The Sixth World Congress on, vol. 2. IEEE, 2006, pp. 8587–8591.

[11] W. Dong and X. Wu, “Fatigue detection based on the distance of eyelid,” in VLSI Design and Video Technology, 2005. Proceedings of 2005 IEEE International Workshop on. IEEE, 2005, pp. 365–368.

[12] T. Azim, M. A. Jaffar, and A. M. Mirza, “Fully automated real time fatigue detection of drivers through fuzzy expert systems,” Applied Soft Computing, vol. 18, pp. 25–38, 2014.

[13] B. Mandal, L. Li, G. S. Wang, and J. Lin, “Towards detection of bus driver fatigue based on robust visual analysis of eye state,” IEEE Transactions on Intelligent Transportation Systems, vol. 18, no. 3, pp. 545–557, 2017.

[14] Z. Boulkenafet, J. Komulainen, and A. Hadid, “Face anti-spoofing based on color texture analysis,” in Image Processing (ICIP), 2015 IEEE International Conference on. IEEE, 2015, pp. 2636–2640.

[15] J. Galbally and S. Marcel, “Face anti-spoofing based on general image quality assessment,” in Pattern Recognition (ICPR), 2014 22nd International Conference on. IEEE,2014, pp. 1173–1178.

[16] L. Li, X. Feng, X. Jiang, Z. Xia, and A. Hadid, “Face anti-spoofing via deep local binary patterns,” in Image Processing (ICIP), 2017 IEEE International Conference on. IEEE, 2017, pp. 101–105.

[17] H. Steiner, A. Kolb, and N. Jung, “Reliable face anti-spoofing using multispectral swir imaging,” in Biometrics (ICB), 2016 International Conference on. IEEE, 2016, pp. 1–8.

[18] H. Li, P. He, S. Wang, A. Rocha, X. Jiang, and A. C. Kot, “Learning generalized deep feature representation for face anti-spoofing,” IEEE Transactions on Information Forensics and Security, vol. 13, no. 10, pp.2639–2652, 2018.

[19] G. Pan, L. Sun, Z. Wu, and S. Lao, “Eyeblink-based anti-spoofing in face recognition from a generic webcamera,” in Computer Vision, 2007.ICCV 2007. IEEE 11th International Conference on. IEEE, 2007, pp.1–8.

[20] F. M. Sukno, S.-K. Pavani, C. Butakoff, and A. F. Frangi, “Automatic assessment of eye blinking patterns through statistical shape models,” in International Conference on Computer Vision Systems. Springer, 2009, pp. 33–42.

[21] D. Torricelli, M. Goffredo, S. Conforto, and M. Schmid, “An adaptive blink detector to initialize and update a view-basedremote eye gaze tracking system in a natural scenario,” Pattern Recognition Letters, vol. 30, no. 12, pp. 1144–1150, 2009.

[22] M. Divjak and H. Bischof, “Eye blink based fatigue detection for prevention of computer vision syndrome.” in MVA, 2009, pp. 350–353.

[23] F. Yang, X. Yu, J. Huang, P. Yang, and D. Metaxas, “Robust eyelid tracking for fatigue detection,” in Image Processing (ICIP), 2012 19th IEEE International Conference on. IEEE, 2012, pp. 1829–1832.

[24] T. Drutarovsky and A. Fogelton, “Eye blink detection using variance of motion vectors,” in European Conference on Computer Vision. Springer, 2014, pp. 436–448.

[25] T. Soukupova and J. Cech, “Real-time eye blink detection using facial landmarks,” in 21st Computer Vision Winter Workshop (CVWW2016), 2016, pp. 1–8.

[26] K. W. Kim, H. G. Hong, G. P. Nam, and K. R. Park, “A study of deep cnn-based classification of open and closed eyes using a visible light camera sensor,” Sensors, vol. 17, no. 7, p. 1534, 2017.

[27] K. He, X. Zhang, S. Ren, and J. Sun, “Deep residual learning for image recognition,” in CVPR, 2016, pp. 770–778.

[28] J. G. Lawrenson, R. Birhah, and P. J. Murphy, ‘‘Tear-film lipid layer morphology and corneal sensation in the development of blinking in neonates and infants,’’ J. Anatomy, vol. 206, no. 3, pp. 265–270, Mar. 2005.

[29] A. J. Zametkin, J. R. Stevens, and R. Pittman, ‘‘Ontogeny of spontaneous blinking and of habituation of the blink reflex,’’ Ann. Neurol., vol. 5, no. 5,pp. 453–457, May 1979.

[30] P. J. De Jong and H. Merckelbach, ‘‘Eyeblink frequency, rehearsal activity,and sympathetic arousal,’’ Int. J. Neurosci., vol. 51, nos. 1–2, pp. 89–94,Jan. 1990. 14

[31] J. Oh and J. Jeong, ‘‘Potential significance of eyeblinks as a behavior marker of neuropsychiatric disorders,’’ Korean J. Biol. Psychiatry, vol. 19, no. 1, pp. 9–20, 2012.

[32] Yuezun Li, Xin Yang, Pu Sun, Honggang Qi, and Siwei Lyu. Celeb-df: A new dataset for deepfake forensics. arXiv preprint arXiv:1909.12962, 2019.

[33] Xin Yang, Yuezun Li, and Siwei Lyu. Exposing deep fakes using inconsistent head poses. In ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), pages 8261–8265. IEEE, 2019.

[34] G. Bradski. The OpenCV Library. Dr. Dobb’s Journal of Software Tools, 2000.

[35] Bruce D Lucas, Takeo Kanade, et al. An iterative image registration technique with an application to stereo vision. 1981.

[36] Gunnar Farneb¨ack. Two-frame motion estimation based on polynomial expansion. In Scandinavian conference on Image analysis, pages 363–370. Springer, 2003.{For Gunnar Farneback in Optical Flow}

[37] Sepp Hochreiter; Jürgen Schmidhuber (1997). "Long short-term memory". Neural Computation. 9 (8): 1735–1780. doi:10.1162/neco.1997.9.8.1735. PMID 9377276. S2CID 1915014.

[38] S. Suwajanakorn, S. M. Seitz, I. Kemelmacher-Shlizerman, "Synthesizing Obama: learning lip sync from audio", Jul. 2017

[39] F. Chollet, “Xception: Deep learning with depthwise separable convolutions,” in Proc. IEEE Conf. Comput. Vision Pattern Recognit., 2017, pp. 1800–1807.

[40] Fazle Karim, Somshubra Majumdar, Houshang Darabi and Shun Chen, "LSTM Fully Convolutional Networks for Time Series Classification" IEEE Access, vol. 6, pp. 1662-1669, 2017.

[41] J. Donahue, L., “Long-term recurrent convolutional networks for visual recognition and description,” in CVPR, 2015, pp. 2625–2634.

[42] Rösler, O., and Suendermann, D., "A first step towards eye state prediction using eeg", in Proc. AIHLS, Istanbul, Turkey,2013.

[43] R. Ranjan, V. M. Patel, R. Chellappa, “HyperFace: A deep multitask learning framework for face detection landmark localization pose estimation and gender recognition”, 2016.

[44] Tackhyun Jung , Sangwon Kim And Keecheon Kim“DeepVision: Deepfakes Detection Using Human Eye Blinking Pattern” –April 2020.

[45] T. Karras, S. Laine, and T. Aila, “A Style-Based Generator Architecture for Generative Adversarial Networks,” in Proc. IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2019.

[46] E. Gonzalez-Sosa, J. Fierrez, R. Vera-Rodriguez, and F. AlonsoFernandez, “Facial Soft Biometrics for Recognition in the Wild: Recent Works, Annotation and COTS Evaluation,” IEEE Transactions on Information Forensics and Security, vol. 13, no. 8, pp. 2001–2014, 2018.

[47] Y. Choi, M. Choi, M. Kim, J. Ha, S. Kim, and J. Choo, “StarGAN: Unified Generative Adversarial Networks for Multi-Domain Imageto-Image Translation,” in Proc. IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2018.

[48] M. Liu, Y. Ding, M. Xia, X. Liu, E. Ding, W. Zuo, and S. Wen, “STGAN: A Unified Selective Transfer Network for Arbitrary Image Attribute Editing,” in Proc. IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2019.

[49] J. Thies, M. Zollhofer, M. Stamminger, C. Theobalt, and M. Nießner, “Face2face: Real-Time Face Capture and Reenactment of RGB Videos,” in Proc. IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2016.

[50] J. Thies, M. Zollhofer, and M. Nießner, “Deferred Neural Rendering: ¨Image Synthesis using Neural Textures,” ACM Transactions on Graphics, vol. 38, no. 66, pp. 1–12, 2019