ENHANCING DEEPFAKE DETECTION: A MULTIMODAL APPROACH FOR IMPROVED ACCURACY

A PROJECT REPORT

Submitted by

KISHOREKUMAR V [211420104135]

MONNIESH B [211420104168]

NIVESHKUMAR S [211420104186]

in partial fulfillment for the award of the degree of

BACHELOR OF ENGINEERING

IN

COMPUTER SCIENCE AND ENGINEERING

PANIMALAR ENGINEERING COLLEGE

(An Autonomous Institution, Affiliated to Anna University, Chennai)

APRIL 2024

PANIMALAR ENGINEERING COLLEGE

(An Autonomous Institution, Affiliated to Anna University, Chennai)

BONAFIDE CERTIFICATE

Certified that this project report **"ENHANCING DEEPFAKE DETECTION: A MULTIMODAL APPROACH FOR IMPROVED ACCURACY"** is the bonafide work of **KISHOREKUMAR V [211420104135], MONNIESH B [211420104168], NIVESHKUMAR S [211420104186]** who carried out the project work under my supervision

SIGNATURE

Dr. L. JABASHEELA, M.E., Ph.D.,

HEAD OF THE DEPARTMENT

Department of CSE,
Panimalar Engineering College,
Nazarathpettai,
Poonamallee,
Chennai-600 123.

SIGNATURE

Mr.A. KARTHIKEYAN, M.Tech.,

SUPERVISOR

ASSISTANT PROFESSOR

Department of CSE,
Panimalar Engineering College,
Nazarathpettai,
Poonamallee,
Chennai-600 123.

Certified that the above candidate(s) was examined in the End Semester Project Viva-Voce Examination held on.....

INTERNAL EXAMINER

EXTERNAL EXAMINER

DECLARATION BY THE STUDENT

We **KISHOREKUMAR V [211420104135], MONNIESH B [211420104168], NIVESHKUMAR S [211420104186]** hereby declare that this project report titled "**ENHANCING DEEPFAKE DETECTION: A MULTIMODAL APPROACH FOR IMPROVED ACCURACY"**, under the guidance of **Mr. A. KARTHIKEYAN** is the original work done by us and we have not plagiarized or submitted to any other degree in any university by us

KISHOREKUMAR V

MONNIESH B

NIVESHKUMAR S

ACKNOWLEDGEMENT

We would like to express our deep gratitude to our respected Secretary and Correspondent **Dr. P. CHINNADURAI, M.A., Ph.D.** for his kind words and enthusiastic motivation, which inspired us a lot in completing this project.

We express our sincere thanks to our beloved Directors **Tmt. C. VIJAYARAJESWARI**, **Dr. C. SAKTHI KUMAR, M.E., Ph.D** and **Dr. SARANYASREE SAKTHI KUMAR B.E., M.B.A., Ph.D.,** for providing us with the necessary facilities to undertake this project.

We also express our gratitude to our Principal **Dr. K. MANI, M.E., Ph.D.** who facilitated us in completing the project.

We thank the Head of the CSE Department, **Dr. L. JABASHEELA, M.E., Ph.D.,** for the support extended throughout the project.

We would like to express our sincere thanks to **Project Coordinator** **Dr. G. SENTHILKUMAR, M.C.A., M.Phil., M.B.A, M.E., Ph.D.,** and **Project Guide** **Mr. A. KARTHIKEYAN, M.Tech.** and all the faculty members of the Department of CSE for their unwavering support for the successful completion of the project.

KISHOREKUMAR V

MONNIESH B

NIVESHKUMAR S

TRIOS TECHNOLOGIES PVT LTD

We There Wherever The Technology Is

PROJECT COMPLETION LETTER

This is to certify that the following students of **B.E-CSE. PANIMALAR ENGINEERING COLLEGE** has successfully completed their project in our company. The Topic Titled **"ENHANCING DEEPFAKE DETECTION: A MULTIMODAL APPROACH FOR IMPROVED ACCURACY"**.

The names are follows:

1. **KISHOREKUMAR V (REG NO: 211420104135)**
2. **MONNIESH B (REG NO: 211420104168)**
3. **NIVESHKUMAR S (REG NO: 211420104186)**

TRIOS TECHNOLOGIES PVT LTD. Chennai

HR MANAGER

VENKATESAN RAMAMOORTHY

TRIOS TECHNOLOGIES PVT LTD.,

Rani Arcade, #24/5, 1st Avenue, Ashok Nagar, Chennai -83. (Near Ashok Pillar)
www.triostechnology.com support@triostechnology.com 044 4854 0077

TABLE OF CONTENTS

ABSTRACT

As deepfake techniques become more sophisticated, the demand for fake facial image detection continues to increase. Various deepfake detection techniques have been introduced but detecting all types of deepfake images with a single model remains challenging. We propose a technique for detecting various types ofdeepfake images using three common traces generated by deepfakes: residual noise, warping artifacts, and blur effects. We adopted a network designed for steganalysis to detect pixel-wise residual-noise traces.

We also consider landmarks, which are the primary parts of the face where unnatural deformations often occur in deepfake images, to capture high-level features. Finally, because the effect of a deepfake is similar to that of blurring, weapply features from various image quality measurement tools that can capture traces of blurring. The results demonstrate that each detection strategy is efficient,and that the performance of the proposed network is stable and superior to that of existing detection networks on datasets of various deep fake types.

Several deepfake detection approaches have been presented in this effort on picture forensics of generic image alterations. This used noise due to erroneous geometry and light predictions, as well as colour mismatch in two eyes. Deepfake employing the 2D direction discrepancy between the head's general circumferenceand restricted facial region. The two basic fake-face detection networks that are suggested make use of macroscopic characteristics. Since training uses an RGB colour space distribution, the colour space was converted to HSV, and deepfake was identified by utilising the statistical difference between the two colour spaces.

LIST OF FIGURE

CHAPTER-1
INTRODUCTION

CHAPTER-1
INTRODUCTION

1.1 OVERVIEW

The increasing sophistication of mobile camera technology and the ever-growing reach of social media and media sharing portals have made the creation and propagation of digital videos more convenient than ever before. Until recently, the number of fake videos and their degrees of realism has been limited by the lack of sophisticated editing tools, the high demand on domain expertise, and the complex and time-consuming process involved. However, the time of fabrication and manipulation of videos has decreased significantly in recent years, thanks to the accessibility to large-volume training data and high-through put computing power, but more to the growth of machine learning and computer vision techniques that eliminate the need for manual editing steps.

In particular, a new vein of AI-based fake video generation methods known as Deep Fake has attracted a lot of attention recently. It takes as input a video of a specific individual (’target’), and outputs another video with the target’s faces replaced with those of another individual (’source’). The back bone of deep fake is deep neural networks trained on face images to automatically map the facial expressions of the source to the target. With proper post-processing, the resulting videos can achieve a high level of realism.

Our approach is based on the same procedure that GAN uses to construct the DF. The technique is based on the features of the DF videos. The DF algorithm can only synthesise face pictures of a certain size owing to production time and computational resource constraints; thus, the face images must undergo an affinal warp to fit the source's facial configuration. The back bone of deep fake is deep neural networks trained on face images to automatically map the facial expressions of the source to the target.

1.2 PROBLEM DEFINITION

The objective of this study is to proliferation of deep fake content poses a serious threat to various aspects of society, including politics, journalism, and personal privacy. The problem lies in the difficulty of distinguishing between genuine and manipulated media, which can lead to the dissemination of false information and manipulation of public opinion. Thus, there is a critical need for reliable deep fake detection methods to identify and mitigate the impact of manipulated content. In today's digital age, the proliferation of deep fake technology poses a significant threatto the authenticity and integrity of media content.

Deep fake refers to the use of artificial intelligence and machine learning techniques to create highly realistic but fabricated images, videos, or audio recordings that can deceive viewers into believing false narratives or events. This technology has the potential to manipulate public opinion, spread misinformation, and even disrupt political processes. Hence, there is an urgent need for robust deep fake detection systems to mitigate these risks and safeguard the credibility of digital media.

The Challenge lies in developing effective algorithms and methodologies capable of accurately detecting deep fake content amidst the vast volume of digital media circulating online. Deep fake techniques are constantly evolving, becoming increasingly sophisticated and difficult to detect using traditional methods. Moreover, the accessibility of deep fake tools and the ease of creating convincing forgeries make it imperative to stay ahead in the arms race between creators and detectors. One of the key issues in deep fake detection is the lack of large-scale labeled datasets for training and evaluation purposes. Collecting diverse and representative datasets that encompass various deep fake manipulation techniques and scenarios is crucial for enhancing the robustness and generalization capabilities of detection models.

CHAPTER-2
LITERATURE SURVEY

CHAPTER-2
LITERATURE SURVEY

1. Title: Domain General Face Forgery Detection by Learning to Weight

Year: 2021

Authors: Li H, Wang X., Zhang Y.

Abstract: In this paper, we propose a domain-general model, termed learning-to-weight (LTW) that guarantees face detection performance across multiple domains, particularly the target domains that are never seen before. However, various face forgery methods cause complex and biased data distributions, making it challenging to detect fake faces in unseen domains. We argue that different faces contribute differently to a detection model trained on multiple domains, making the model likely to fit domain-specific biases. As such, we propose the LTW approach based on the meta-weight learning algorithm, which configures different weights for face images from different domains. The LTW network can balance the model's generalizability across multiple domains. Then, the meta-optimization calibrates the source domain's gradient enabling more discriminative features to be learned. The detection ability of the network is further improved by introducing an intra-class compact loss. Extensive experiments on several commonly used deep fake datasets to demonstrate the effectiveness of our method in detecting synthetic faces

2. Title: Exploring Frequency Adversarial Attacks for face forgery Detection

Year: 2022

Authors: Shuai Jia, Chao Ma, Taiping Yao, Bangjie Yin, Shouhong Ding

Abstract: Various facial manipulation techniques have drawn serious public concerns in morality, security, and privacy. Although existing face forgery classifiers achieve promising performance on detecting fake images, these methods are vulnerable to adversarial examples with injected imperceptible perturbations on the pixels. Meanwhile, many face forgery detectors always utilize the frequency diversity between real and fake faces as a crucial clue. In this paper, instead of injecting adversarial

perturbations into the spatial domain, we propose a frequency adversarial attack method against face forgery detectors. Concretely, we apply discrete cosine transform (DCT) on the input images and introduce a fusion module to capture the salient region of adversary in the frequency domain. Compared with existing adversarial attacks (e.g. FGSM, PGD) in the spatial domain, our method is more imperceptible to human observers and does not degrade the visual quality of the original images. Moreover, inspired by the idea of meta-learning, we also propose a hybrid adversarial attack that performs attacks in both the spatial and frequency domains. Extensive experiments indicate that the proposed method fools not only the spatial-based detectors but also the state-ofthe-art frequency-based detectors effectively. In addition, the proposed frequency attack enhances the transferability across face forgery detectors as black-box attacks

3. **Title:** NAS-FAS: Static-Dynamic Central Diffrence Network Search For Face Anti-Spoofing

Year:2020

Authors Zitong Yu, Jun Wan, Yunxiao Qin, Xiaobai Li, Stan Z.Li, Guoying Zhao.

Abstract: Face anti-spoofing (FAS) plays a vital role in securing face recognition systems. Existing methods heavily rely on the expert-designed networks, which may lead to a sub-optimal solution for FAS task. Here we propose the first FAS method based on neural architecture search (NAS), called NAS-FAS, to discover the well-suited task-aware networks. Unlike previous NAS works mainly focus on developing efficient search strategies in generic object classification, we pay more attention to study the search spaces for FAS task. The challenges of utilizing NAS for FAS are in two folds: the networks searched on 1) a specific acquisition condition might perform poorly in unseen conditions, and 2) particular spoofing attacks might generalize badly for unseen attacks. To overcome these two issues, we develop a novel search space consisting of central difference convolution and pooling operators. Moreover, an efficient static-dynamic representation is exploited for fully mining the FAS-aware spatio-temporal discrepancy.

4. Title: Learning Meta Pattern for Face Anti-Spoofing

Year: 2022

Authors: Rizhao cai, zhi Li, Renjie Wan, Haoliang Li, Yongjian Hu, Alex C.Kot

Abstract: Face Anti-Spoofing (FAS) is essential to secure face recognition systems and has been extensively studied in recent years. Although deep neural networks (DNNs) for the FAS task have achieved promising results in intra-dataset experiments with similar distributions of training and testing data, the DNNs' generalization ability is limited under the cross-domain scenarios with different distributions of training and testing data. To improve the generalization ability, recent hybrid methods have been explored to extract task-aware handcrafted features (e.g., Local Binary Pattern) as discriminative information for the input of DNNs. However, the handcrafted feature extraction relies on experts' domain knowledge, and how to choose appropriate handcrafted features is underexplored. To this end, we propose a learnable network to extract Meta Pattern (MP) in our learningto-learn framework. By replacing handcrafted features with the MP, the discriminative information from MP is capable of learning a more generalized model. Moreover, we devise a two-stream network to hierarchically fuse the input RGB image and the extracted MP by using our proposed Hierarchical Fusion Module (HFM).

CHAPTER-3
SYSTEM ANALYSIS

CHAPTER-3
SYSTEM ANALYSIS

3.1 EXISTING SYSTEM

The existing system is current context revolves around the rapid development of face forgery generating algorithms that create manipulated images and videos, contributing to the proliferation of fake information that's challenging to detect. Due to the significant concerns posed by these facial manipulation technologies, the field of computer vision is increasingly focusing on face forgery detection. However, existing detection systems often struggle in real-world applications due to poor generalization across unseen domains. To address this, the paper introduces a deep fake detection method called Meta Deepfake Detection (MDD), which employs meta-learning principles to establish a model capable of directly detecting unseen domains without requiring frequent updates. MDD utilizes meta-weight learning to adapt information from source domains to target domains, facilitating the generation of effective domain representations.

DISADVANTAGES:

- The utilization of meta-learning, multi-domain strategies, and additional loss functions could potentially increase the complexity of the detection method. This complexity might lead to higher computational requirements, making the approach less efficient in resource-constrained environments
- The success of meta-learning approaches, including meta-weight learning, is often influenced by the characteristics of the training datasets. If the dataset distribution or quality is not representative of real-world scenarios, the method's generalization might still be limited.

3.2 PROPOSED SYSTEM

In this study, we proposed a generalized detection method to detect of deepfake techniques. We exploited three types of common traces residual noise, warping artifacts, and blur effects generated by the deepfake process. We applied them to the proposed network for deepfake detection. First, a network designed for steganalysis was adopted as the base network to detect residual noise. Second, landmark patches were extracted from the semantic facial region to detect warping artifacts, which are unnatural high-level features. Finally, we applied IQM features to capture the statistical characteristics of the blur-like effects of a deepfake. The results revealed that each detection strategy is effective, and the performance of the proposed network is superior to that of existing networks. Because a deepfake video inherits residual features from image operations, our approach can be directly adopted for deepfake video detection pipelines based on frame-by-frame detection. Based on the proposed method, we plan to expand this study to include a deepfake video detection method. We hope this method is robust against signal- and time-based attacks.

Advantages:

- We propose a generalized detection method using traces to detect three types of deepfake: face swap, puppet master, and attribute change.
- We developed a network based on image quality measurement (IQM) features and warping artifacts extracted from facial landmarks.
- We propose using a network designed for steganalysis to capture residual noise traces in deepfake images.

3.3 PROJECT REQUIREMENTS

General:

Requirements are the basic constrains that are required to develop a system. Requirements are collected while designing the system. The following are the requirements that are to be discussed.

1. Functional Requirements
2. Non-Functional Requirements
3. Environment Requirements
    A. Hardware Requirements
    B. Software Requirements

1. Functional Requirements:

The software requirements specification is the first step in the requirements analysis process. It lists requirements of a particular software system. The following details to follow the special libraries like Python, Numpy, Pandas, Matplotlib ,Vscode, Jupyter notebook.

2. Non-Functional Requirements:

1. Data Input
2. Data Preprocessing
3. Machine Learning Model Development
4. Model Training and Evaluation
5. Real-time Prediction
6. User Interface
7. Integration
8. Security and Privacy
9. Scalability
10. Documentation and Reporting

3. Environmental Requirements:

A. Hardware system configuration:

Processor	-	Intel i3, i5, i7, AMD
ProcessorRAM	-	Above 8 GB
Hard Disk	-	Above 500 GB

B. Software system configuration:

Operating System	-	Windows 7/8/10/11
Front End	-	HTML, CSS
Scripts	-	Anaconda (Jupiter)
Tool	-	Python (3.10)

SOFTWARE DESCRIPTION

INTRODUCTION TO PYTHON

Python is a high-level object-oriented programming language that was created by Guido van Rossum. It is also called general-purpose programming language as it is used in almost every domain we can think of as mentioned below:

- Web Development
- Software Development
- Game Development
- AI & ML
- Data Analytics

WHY PYTHON PROGRAMMING?

You guys might have a question in mind that, why python? why not another programming language? So let me explain:Every Programming language serves some purpose or use-case according to a domain. for e.g., Javascript is the most popular language amongst web developers as it gives the developer the power to handle applications via different frameworks like react, angular which are used to build beautiful User Interfaces. Similarly, they have pros and cons at the same time. so if we consider python it is general-purpose which means it is widely used in every domain the reason is it's very simple to understand, scalable because of which the speed of development is so fast. Now you get the idea why besides learning python it doesn't require any programming background so that's why it's popular amongst developers as well. Python has simpler syntax similar to the English language and also the syntax allows developers to write programs with fewer lines of code. Since it is open-source there are many libraries available that make developers' jobs easy ultimately results in high productivity. They can easily focus on business logic and its demanding skills in the digital era where information is available in large data sets.

INSTALLING PYTHON PACKAGES

To install Python packages, you can use pip, the package installer for Python. Here are the steps to install Python packages using pip:

- Open a command prompt (Windows) or terminal (Mac/Linux).
- Type pip install <package_name> and press Enter. Replace <package_name>with the name of the package you want to install.
- Wait for the installation to complete. pip will automatically download and install the package and its dependencies.

CHAPTER-4
SYSTEM DESIGN

CHAPTER-4
SYSTEM DESIGN

4.1 UML DIAGRAMS

Unified Modeling Language (UML) is a general-purpose modelling language. The main aim of UML is to define a standard way to visualize the way a system has been designed. It is quite similar to blueprints used in other fields of engineering.

4.1.1 USE CASE DIAGRAM

Use case diagrams are considered for high level requirement analysis of a system. When the requirements of a system are analyzed, the functionalities are captured in use cases. So, it can say that use cases are system functionalities written in an organized manner.

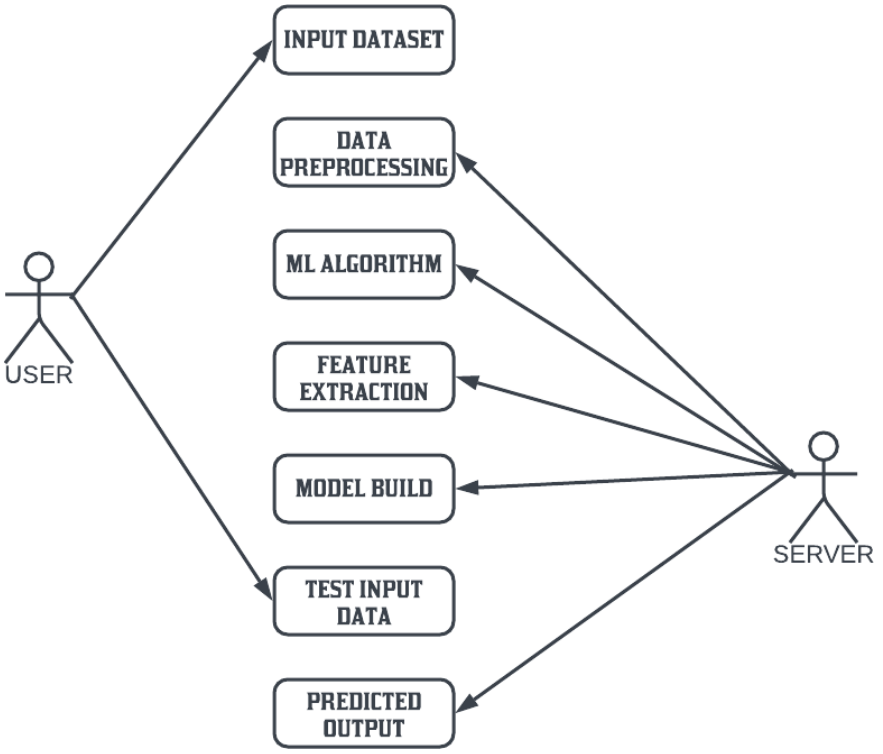


Figure 4.1.1 Use Case Diagram

4.1.2 ACTIVITY DIAGRAM:

A graphical representation of an executed set of procedural system activities and considered a state chart diagram variation. Activity diagrams describe parallel and conditional activities, use cases and system functions at a detailed level.

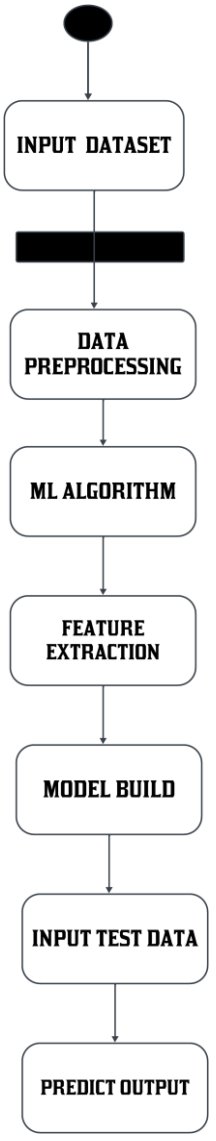


Figure 4.1.2 Activity Diagram

4.1.3 CLASS DIAGRAM:

Class diagram is basically a graphical representation of the static view of the system and represents different aspects of the application. The name of the class diagram should be meaningful to describe the aspect of the system. Each element and their relationships should be identified in advance.

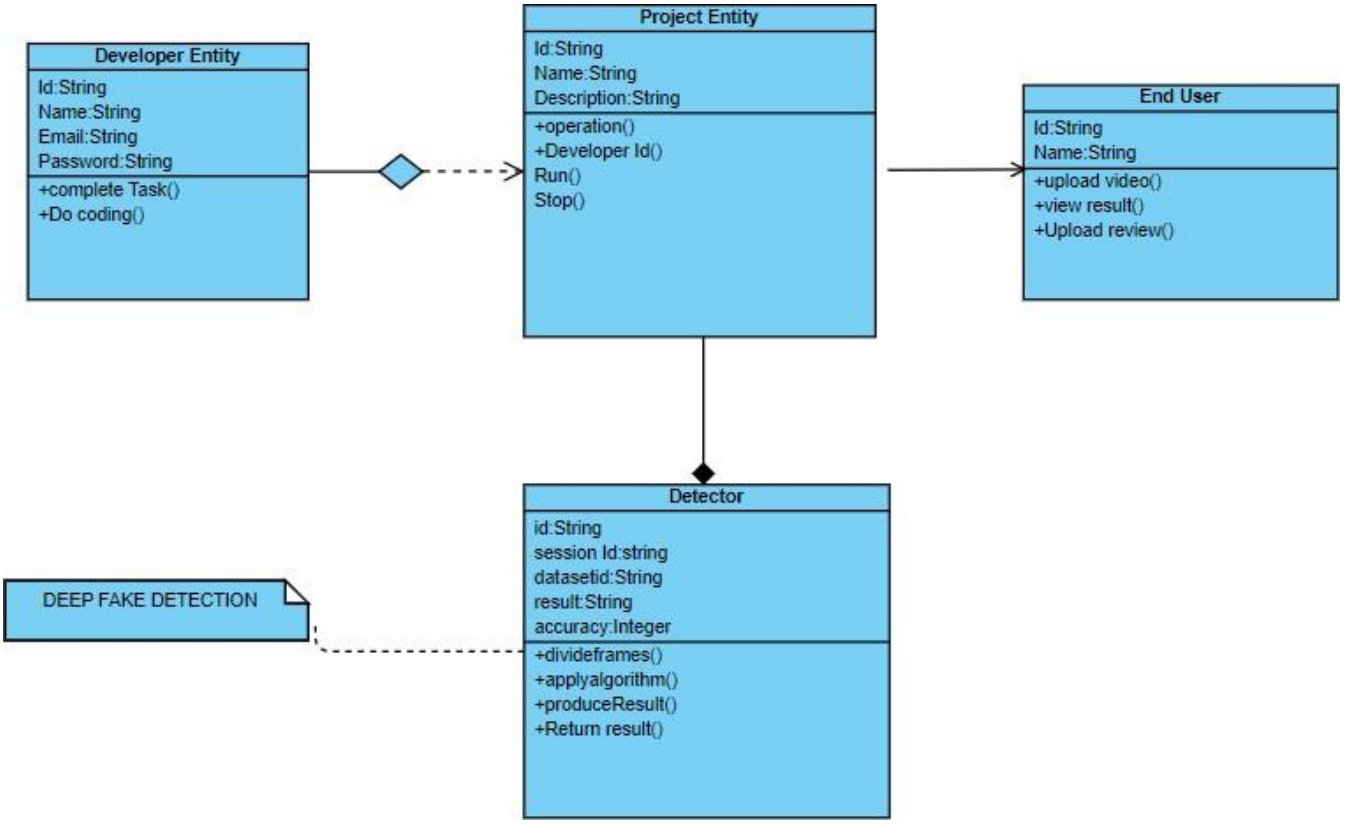


Figure 4.1.3 Class Diagram

4.2 Data Flow Diagram:

A data flow diagram (DFD) is a graphical representation of the "flow" of data through an information system, modeling its process aspects. It can be used for the visualization of data processing (structured design).Data flow diagrams are also known as bubble charts. DFD is a designing tool used in the top-down approach to Systems Design. DFD levels are numbered 0, 1 or 2, and occasionally go to even Level 3 or beyond. DFD Level 0 is also called a Context Diagram.

Level 0:

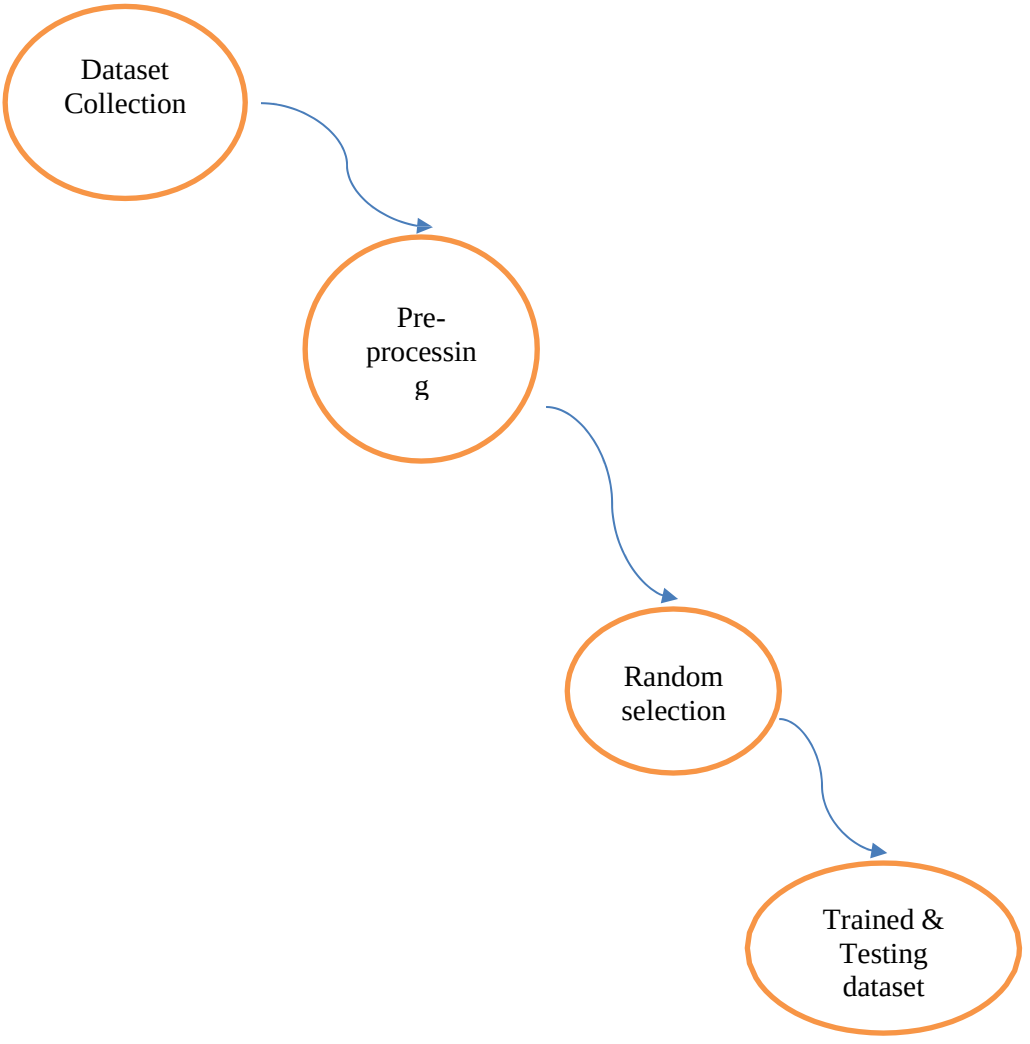


Figure 4.2.1 Level 0 DFD Diagram

Level 1:

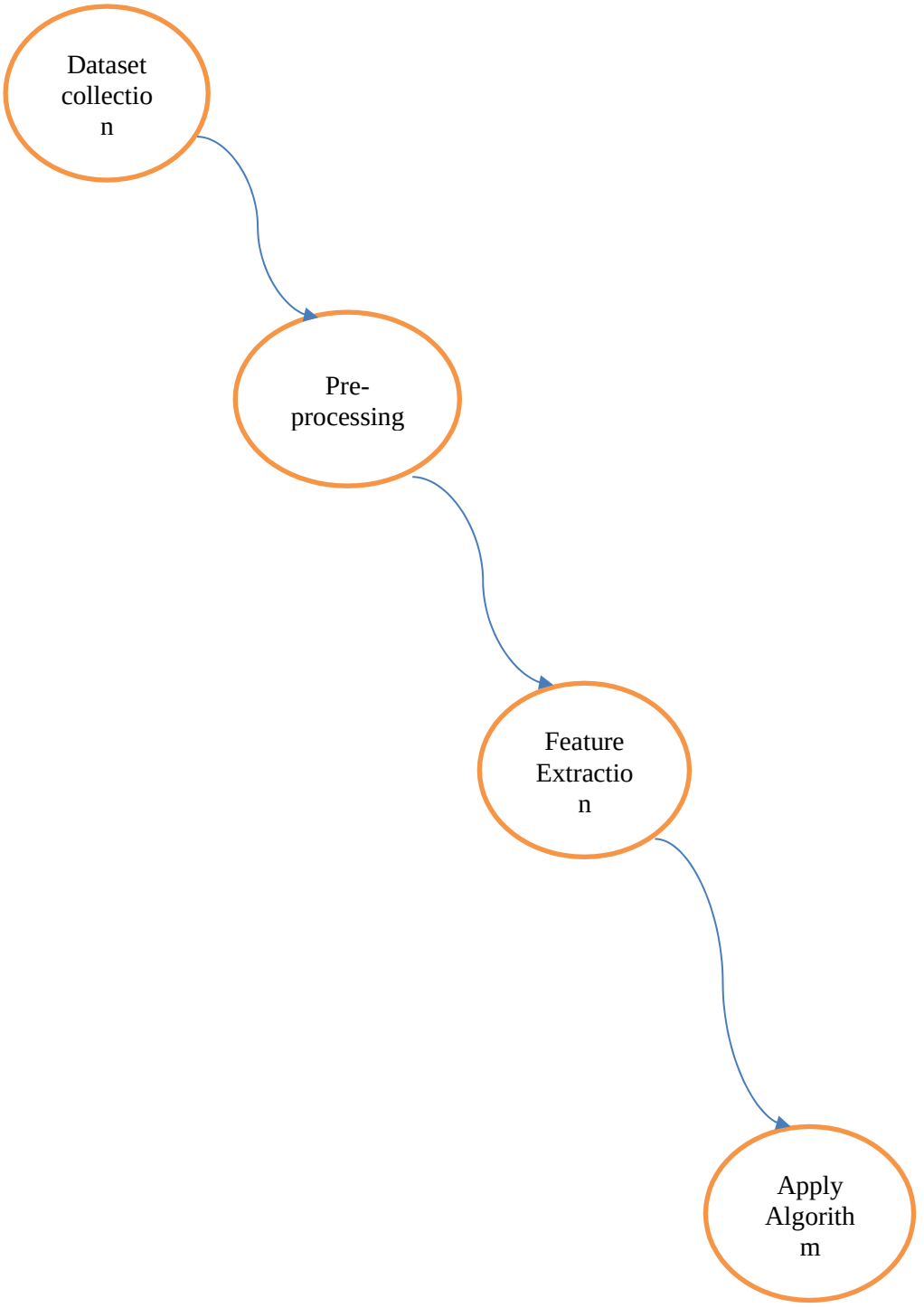


Figure 4.2.2 Level 1 DFD Diagram

LEVEL 2

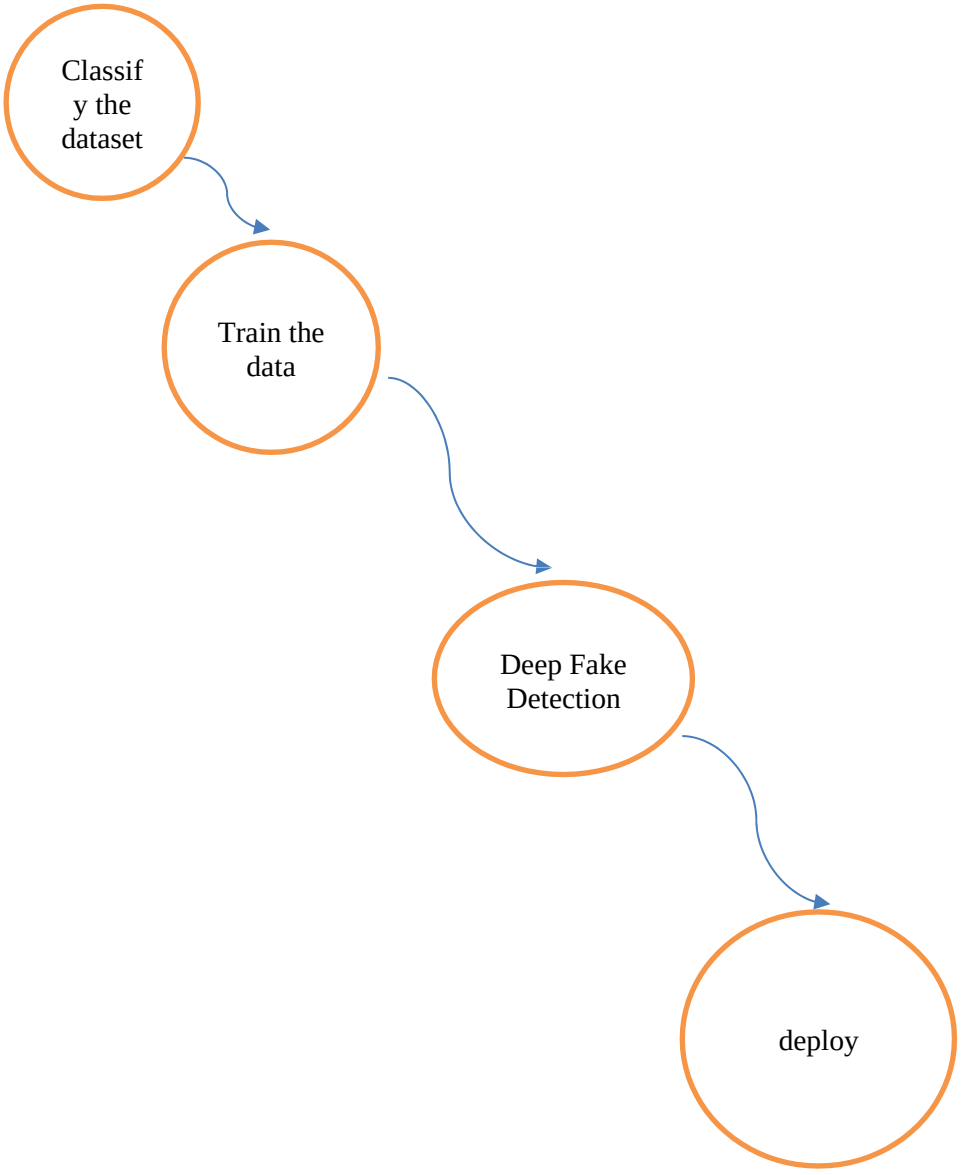


Figure 4.2.3 Level 2 DFD Diagram

CHAPTER-5
SYSTEM ARCHITECTURE

CHAPTER-5

SYSTEM ARCHITECTURE

5.1 SYSTEM ARCHITECTURE OVERVIEW

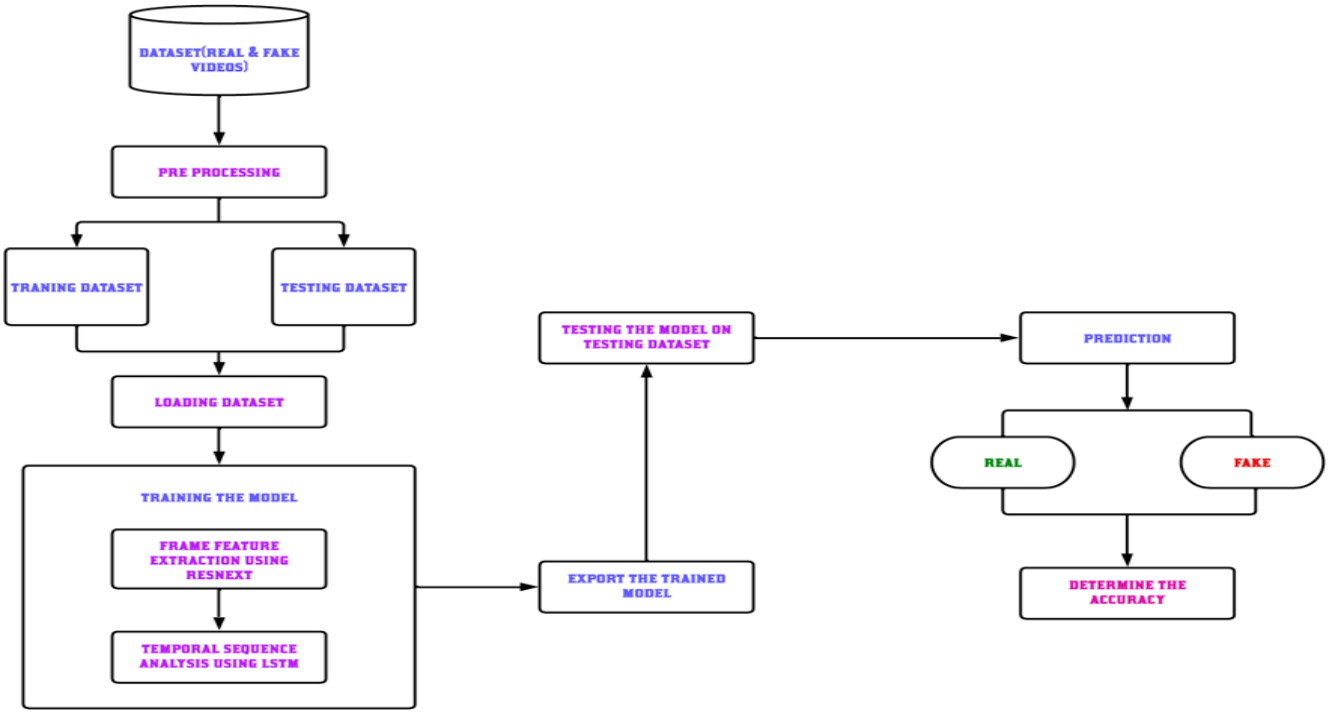


Figure 5.1 System Architecture

A robust system architecture for deep fake detection combines Long Short-Term Memory (LSTM) networks and Residual Networks (ResNet). LSTM models analyze temporal patterns in video sequences, capturing contextual information over time. Meanwhile, ResNet efficiently extracts spatial features from individual frames. By integrating these components, the system can effectively discern inconsistencies between real and manipulated content, leveraging both temporal and spatial cues for accurate deep fake detection.

The architecture diagram shows the processes involved for building the project. It involves collecting dataset from website, the processing it to remove the noisy data, visualizing it and then implementing algorithms and finding the best model based on accuracy and then deploying it in the form of webpage.

CHAPTER-6

SYSTEM IMPLEMENTATION

CHAPTER-6

SYSTEM IMPLEMENTATION

6.1 ALGORITHMS:

1. LSTM Algorithm
2. RESNET Algorithm

6.1.1 LSTM ALGORITHM:

The Long Short-Term Memory (LSTM) algorithm is a specialized form of recurrent neural networks designed to overcome challenges in learning and retaining long-term dependencies within sequential data. LSTMs are equipped with memory cells, gates, and a unique architecture that allows them to capture intricate patterns over extended sequences. Three gates, namely the input gate, forget gate, and output gate, regulate the flow of information, enabling effective learning and retention of temporal dependencies. Widely used in various domains, LSTMs are particularly valuable in time-series prediction tasks, such as meteorological forecasting. In this project, LSTMs are employed to model the complex relationships in historical weather data, providing a powerful tool for accurate monsoon predictions by effectively addressing the sequential nature of meteorological variables. The algorithm's versatility and ability to handle both short-term and long-term dependencies make it a key component in enhancing the accuracy of forecasting models in applications requiring a deep understanding of sequential data.

WORKING OF LSTM ALGORITHM:

The working of the Long Short-Term Memory (LSTM) algorithm involves a sophisticated architecture designed to capture and learn patterns in sequential data, making it particularly effective for tasks such as time-series prediction. The following steps outline the key components and operations of the LSTM algorithm:

1. Initialization:

- The LSTM network begins with the initialization of parameters, including weights and biases, which are adjusted during the training process to minimize prediction errors.

2. Sequential Input Processing:

- For each element in the sequential input data (e.g., time-series data), the LSTM processes the information one step at a time.
- At each time step, the input is passed through the network, and the LSTM makes predictions based on the current input and its internal memory.

3. Memory Cells:

- LSTMs have memory cells that store information over time. These cells can retain information for long durations, addressing the vanishing gradient problem encountered in traditional recurrent neural networks (RNNs).

4. Gates:

- LSTMs have three types of gates that regulate the flow of information:
  - Input Gate: Controls the input into the memory cell, determining which information to store.
  - Forget Gate: Decides what information to discard from the memory cell, helping the network focus on relevant data.
  - Output Gate: Determines the information to be output based on the current input and the content of the memory cell.

5. Information Flow: - The input gate decides how much of the new information should be added to the memory cell.
- The forget gate decides how much of the existing information in the memory cell should be retained.

6. Backpropagation Through Time (BPTT):

- The training of LSTMs involves using a supervised learning approach and backpropagation through time (BPTT).
- During training, the algorithm adjusts the weights and biases to minimize the difference between the predicted output and the actual output at each time step.

7. Learning Temporal Dependencies:

- LSTMs excel at capturing long-term dependencies in sequential data. The memory cells and gates allow the network to learn and retain relevant information over extended periods, making them effective for tasks where understanding the temporal context is crucial.

8. Model Optimization:

- The LSTM model is optimized through iterative training, where the parameters are adjusted to enhance predictive accuracy.
- Performance metrics, such as mean squared error or other relevant metrics, are used to assess and improve the model's effectiveness.

In summary, the LSTM algorithm's working involves a dynamic interplay of memory cells and gates to selectively process, store, and retrieve information over time, enabling it to capture and learn complex patterns in sequential data for accurate predictions. working of the Long Short-Term Memory (LSTM) algorithm involves a sophisticated architecture designed to capture and learn patterns in sequential data, making it particularly effective for tasks such as time-series prediction. For each element in the sequential input data (e.g., time-series data), the LSTM processes the information one step at a time.

PSEUDOCODE FOR LSTM ALGORITHM:

- function lstm_cell(input_x, prev_hidden_state, prev_cell_state, weights, biases):
- Concatenate input and previous hidden state

  concat_input = concatenate(input_x, prev_hidden_state)
- Input gate

  input_gate = sigmoid(dot_product(concat_input, weights['input_gate']) biases['input_gate'])

- Forget gate

  forget_gate = sigmoid(dot_product(concat_input, weights['forget_gate']) + biases['forget_gate'])

- Output gate

  output_gate = sigmoid(dot_product(concat_input, weights['output_gate']) + biases['output_gate'])

- Cell state update

  cell_state_candidate = tanh(dot_product(concat_input, weights['cell_state']) + biases['cell_state'])

  cell_state = forget_gate * prev_cell_state + input_gate * cell_state_candidate

- Hidden state update

  hidden_state = output_gate * tanh(cell_state)

  return hidden_state, cell_state

- Pseudocode for a simple LSTM network

  function lstm_network(inputs, initial_hidden_state, initial_cell_state, weights, biases):

  current_hidden_state = initial_hidden_state

  current_cell_state = initial_cell_state

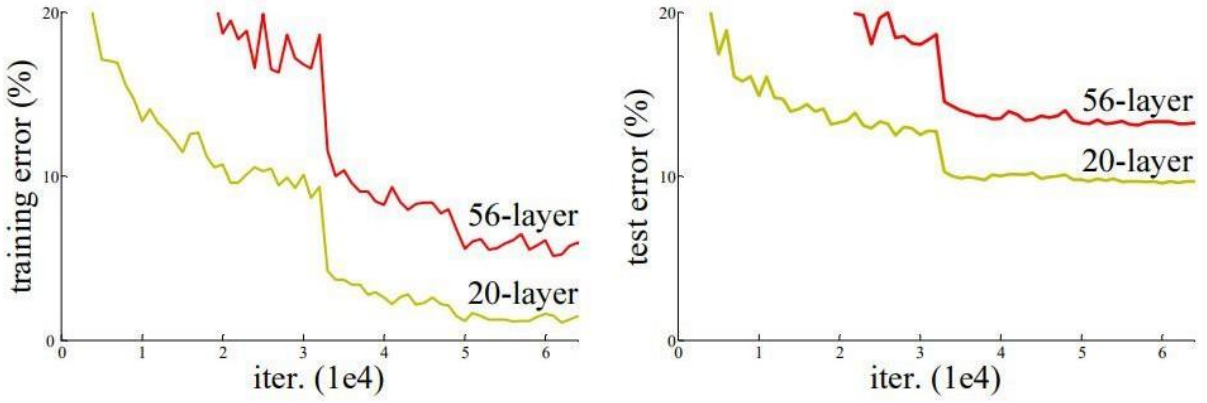
```
for each time step t:
    current_input = inputs[t]
    current_hidden_state, current_cell_state = lstm_cell(current_input,
current_hidden_state, current_cell_state, weights, biases)

return current_hidden_state
```

ADVANTAGES OF LSTM:

- LSTMs are designed to address the vanishing gradient problem, enabling them to capture and remember long-term dependencies in sequential data. This is crucial for tasks where understanding the context over extended periods is essential.
- The gating mechanisms (input gate, forget gate, and output gate) provide LSTMs with control over the flow of information. This adaptability allows the network to decide which information to store, forget, or output, enhancing its flexibility.
- LSTMs have been successfully applied to a wide range of tasks, including machine translation, sentiment analysis, speech recognition, and financial time-series prediction. Their versatility makes them a popular choice for different applications.

6.1.2 RESNET ALGORITHM

After the first CNN-based architecture (AlexNet) that win the ImageNet 2012 competition, Every subsequent winning architecture uses more layers in a deep neural network to reduce the error rate. This works for less number of layers, but when we increase the number of layers, there is a common problem in deep learning associated with that called the Vanishing/Exploding gradient. This causes the gradient to become 0 or too large.



Residual Network: In order to solve the problem of the vanishing/exploding gradient, this architecture introduced the concept called Residual Blocks. In this network, we use a technique called ***skip connections***. The skip connection connects activations of a layer to further layers by skipping some layers in between. This formsa residual block. Resnets are made by stacking these residual blocks together.

The approach behind this network is instead of layers learning the underlying mapping, we allow the network to fit the residual mapping. So, instead of say H(x), initial mapping, let the network fit,

$F(x) := H(x) - x$ which gives $H(x) := F(x) + x$.

The advantage of adding this type of skip connection is that if any layer hurt the performance of architecture then it will be skipped by regularization. So, this results in training a very deep neural network without the problems caused by vanishing/exploding gradient. The authors of the paper experimented on 100-1000 layers of the CIFAR-10 dataset.

There is a similar approach called "highway networks", these networks also use skip connection. Similar to LSTM these skip connections also use parametric gates. These gates determine how much information passes through the skip connection. This architecture however has not provided accuracy better than ResNet architecture.

6.2 MODULE DESIGN SPECIFICATION

1.Data set & Data preprocessing

2.Feature extraction

3.Experimental setup

4.Evaluation metrics

6.3MODULE DESCRIPTION

6.3.1 DATASET AND PREPROCESSING:

The proposed idea is to detect deep fake videos using human-pose estimation. The publicly available datasets are not appropriate for this research as those do not fit the proposed face and body language landmark requirements. The only videos that we can train and test the established hypothesis are the videos of world leaders (i.e., president, vice president, etc.). These videos are widely available with a good quality pixel ratio, and we can also test these videos against deepfakes very quickly. Therefore, we manually downloaded online videos to generate a customized dataset. These videos were annotated and labeled according to the proposed requirements. We need two kinds of videos, the original and the synthetic, in order to test the proposed hypothesis.The original videos were downloaded from the official website of Miller Center to ensure that the videos have not tampered. This website streams US presidential speeches. Initially, we chose videos of the four most recent United States (US) presidents George W. Bush, Barack Obama, Donald Trump, and Joe Biden. The videodownload satisfy conditions (a) the file formats are MP4 with 30 frame-per-second (fps) quality, (b) a similar video frames size for the whole dataset, (c) the person of interest was in the middle of the frame, (d) there is no one else in the background, and
(e) the cameras were relatively stable the dataset samples of real and fake videos for deep fake training.

6.3.2 FEATURE EXTRACTION:

The deep learning model was trained to learn the spatial and temporal data in order to address fake videos using body language analysis. The proposed method will be an automated system that can determine whether the input pose movement belongs to the target person or not. To implement the proposed method, the RNN is the best candidate to capture the dataset's spatial and temporal features. Furthermore, it can be applied to learn the dependencies of sequential data along with graphical features. LSTMs are advanced RNNs, which can learn long-term dependencies without gradient vanishing. Figure 6 shows a structural comparison between RNN and LSTM. LSTM is very effective for learning spatial and temporal features. Thus, the proposed model is a many-to-one LSTM that requires continuous poses as the input, and it predicts a fixed size output.

6.3.3 EXPERIMENTAL SETUP:

To test the proposed hypothesis that body languages can identify and expose deep fakes, a many-to-one LSTM model was designed using PyTorch. PyTorch provides flexible DNN implementation and GPU support. Two kinds of objects are necessary for setting up the training and testing experiment: (a) the object for data loading and (b) the object for model creation. The custom dataset was divided into an 80% training set, a 10% validation set, and a 10% testing set. Moreover, the values were normalized to the range. A specially designed (according to needs of our custom dataset) Data Loader that was used for passing the datasets to the LSTM model for training. The data only contain the body languages with 24 key points, the model to be implemented with a binary classification model having an input layer with 24 features. It also contains a fully connected output layer with a single output unit. The many-to-one LSTM model outputs the prediction at the last time step. It is observed that every 150 poses are a continuous movement; therefore, the time step of the model is 150. The hidden size and the number of hidden layers should be decided based on the experiment. The proper depth and width of the network can prevent under fitting and over fitting problems.

6.3.4 EVALUATION METRICS:

Loss and Accuracy metrics were used to evaluate the model's performance. The binary cross-entropy loss function was applied for experiments. A lower testing loss value projects a more similar predicted distribution to the target distribution. We plotted the loss and accuracy over time to check whether the model was under fitted or over fitted. Whereas, under fitting means that the model fails to converge and over fitting refers to a model having poor generalization performance. This circumstance may happen due to inadequate training data. The generated dataset used for this project is small; high confidence may lead to over fitting. Therefore, we adjusted the hyper-parameters according to both losses and accuracy, while choosing the model with the highest validation accuracy to be the best model. Metrics like loss and accuracy were used to assess the model's performance. For the experiments, the binary cross-entropy loss function was used. A predicted distribution that is closer to the target distribution is projected by a smaller testing loss value. To determine if the model was over- or under-fitted, we plotted the accuracy and loss with time. Conversely, overfitting denotes a model with poor generalisation performance, while underfitting denotes a model that fails to converge. It is possible that insufficient training data led to this situation. This project's produced dataset is limited, and a high degree of confidence might cause overfitting. As a result, we chose the model with the highest validation accuracy to be the optimal model and modified the hyper-parameters based on both losses and accuracy.

CHAPTER-7

PERFORMANCE EVALUATION

CHAPTER-7

PERFORMANCE EVALUATION

7.1. RESULTS AND DISCUSSIONS:

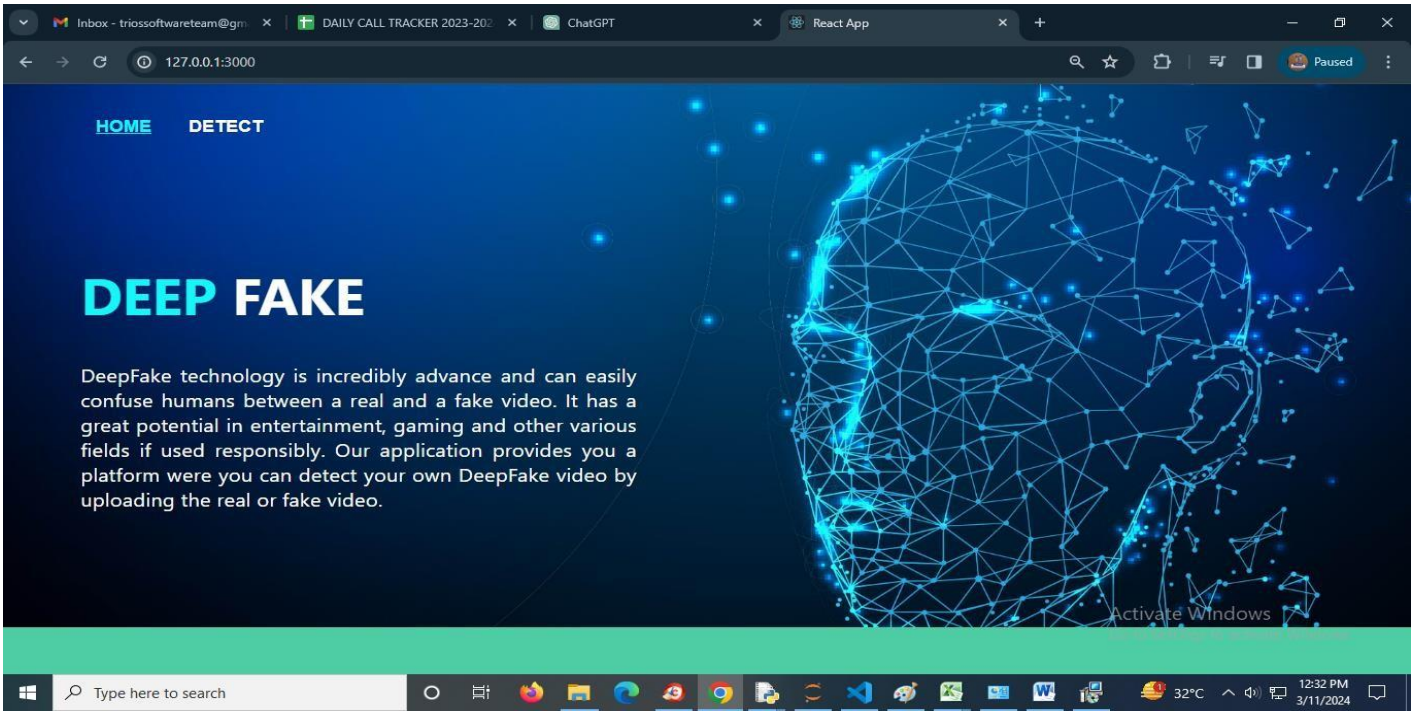


Figure 7.1.1 Home page

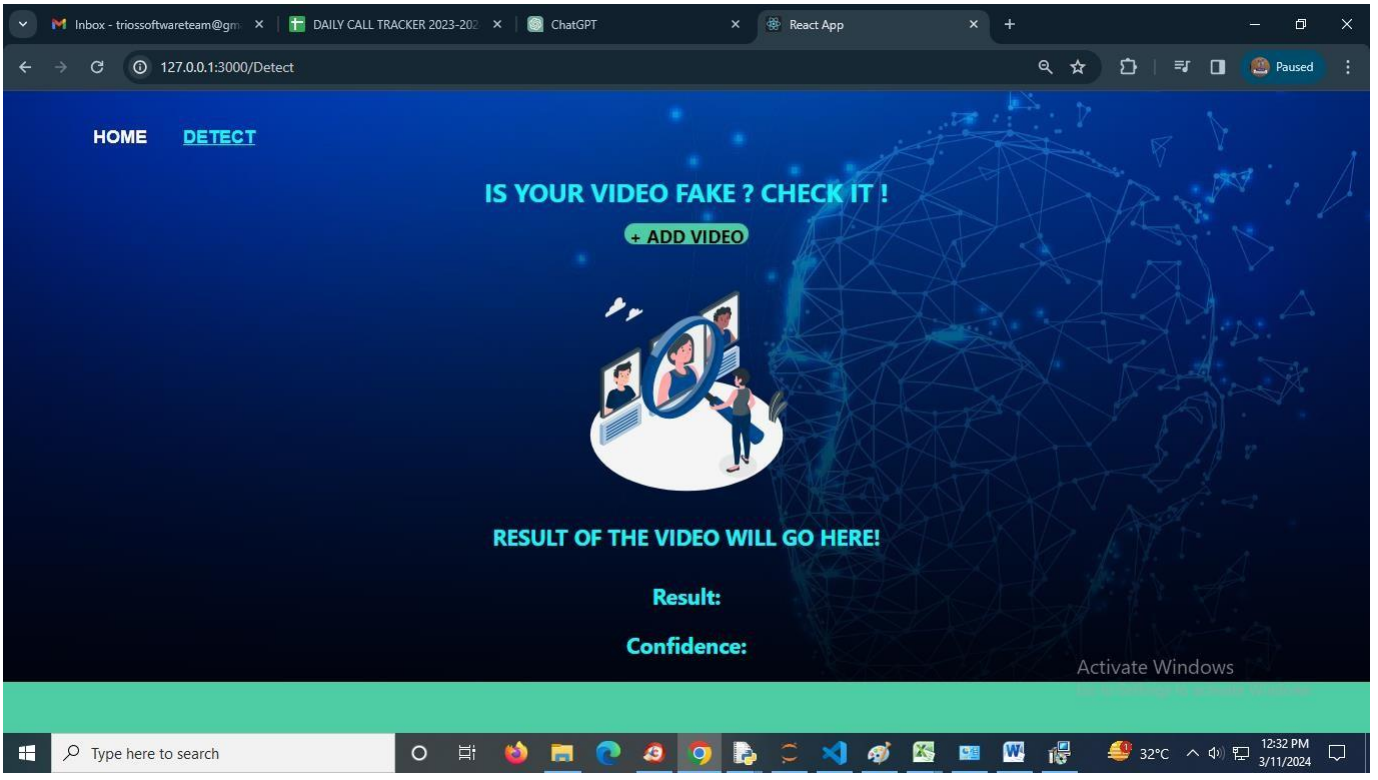


Figure 7.1.2 Detect page

Figure 7.1.3 Output (Real)

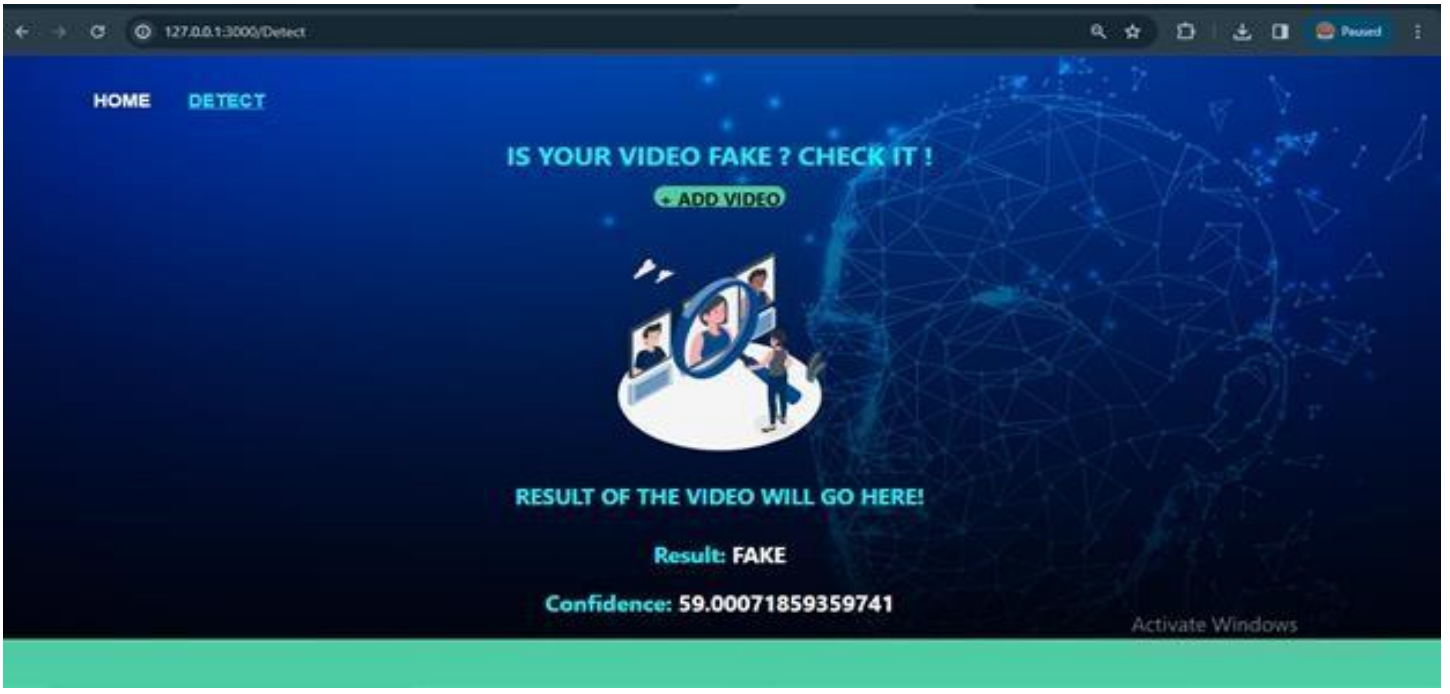


Figure 7.1.4 Output (Fake)

CHAPTER-8
CONCLUSION

8.1 CONCLUSION

This project is used as a deepfake creation, and detection methods. Deepfake creates forged images or videos that persons cannot differentiate from real images or videos. Deepfakes are created using generative adversarial networks, in which two machine learning models exit. One model trains on a dataset and the other model tries to detect the deepfakes. The forger creates fakes until the other model can't detect the forgery. Deepfakes creating fake news, videos, images, and terrorism events that can cause social and financial fraud. It is increasing affects religions, organizations, individuals and communities', culture, security, and democracy. When deepfake videos and images increase on social media people will ignore to trust the truth. So, deepfake datasets and cross-platform detection techniques need to be developed in the future. This needs efficient, reliable and robust mobile detectors to detect deepfakes in widely used mobile devices. Moreover, will improve deepfake detection by integrating deepfake detection and object detection algorithms.

8.2 FUTURE ENHANCEMENT

1. Multi-Modal Detection: Incorporating various modalities such as audio, video, and metadata analysis can provide a more comprehensive understanding of the content, making it harder for deepfake creators to evade detection.
2. Deep Learning Advancements: Continued advancements in deep learning models, such as more robust architectures and improved training techniques, can lead to better detection accuracy and generalization across different types of deepfakes.
3. Generative Model Analysis: Developing methods to analyze the artifacts and inconsistencies generated by deep learning models can help in distinguishing between real and fake content more effectively.
4. Biometric Verification: Integrating biometric verification techniques, such as facial recognition and voice analysis, can add an extra layer of authentication to identify if the person in the media matches their known biometric data.

5. Blockchain and Watermarking: Implementing blockchain technology and digital watermarking can help in tracking the origin and authenticity of media content, making it more difficult for deepfake creators to spread malicious content undetected.

6. Adversarial Training: Training deepfake detection models using adversarial learning techniques can make them more robust against adversarial attacks and manipulation attempts.

7. Real-Time Detection Systems: Developing real-time deepfake detection systems that can quickly analyze and flag suspicious content as it is being uploaded or shared online can help in preventing the rapid spread of harmful deepfakes.

8. Collaborative Efforts: Encouraging collaboration among researchers, industry experts, and policymakers can facilitate the sharing of resources, datasets, and best practices, accelerating the development and deployment of effective deepfake detection solutions.

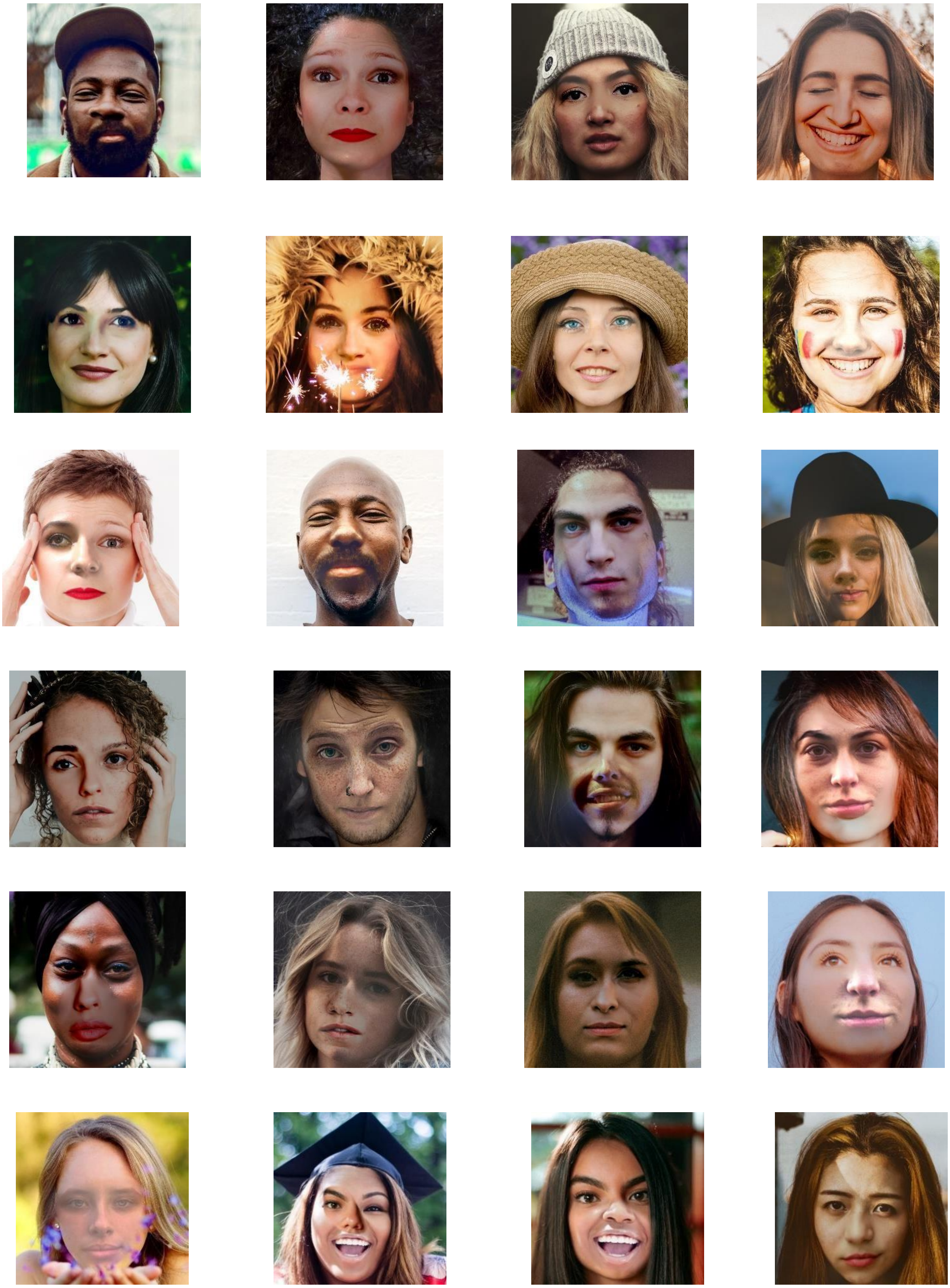
9. Ethical Considerations: Addressing the ethical implications of deepfake detection technologies, such as privacy concerns and potential misuse, is essential to ensure that these technologies are developed and deployed responsibly.

10. User Education and Awareness: Educating users about the existence of deepfakes, how to recognize them, and the potential risks associated with them can empower individuals to be more discerning consumers of media content and less susceptible to manipulation.

CHAPTER-9
APPENDICES

APPENDICES

9.1 SAMPLE DATASET: (FAKE)



SAMPLE DATASET: (REAL)









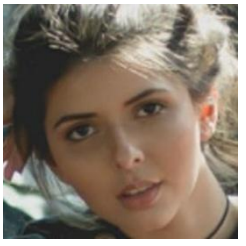










9.2 SAMPLE CODING:

```
import numpy as np
import tensorflow as tf
import pandas as pd
import matplotlib.pyplot as plt
import warnings
warnings.filterwarnings("ignore")

data = tf.keras.utils.image_dataset_from_directory('data/')

class_names = data.class_names
class_names

data_iterator = data.as_numpy_iterator()
batch = data_iterator.next()

plt.figure(figsize=(5, 5))
for images, labels in data.take(1):
  for i in range(9):
    ax = plt.subplot(3, 3, i + 1)
    plt.imshow(images[i].numpy().astype("uint8"))
    plt.title(class_names[labels[i]])
    plt.axis("off")

data = data.map(lambda x,y: (x/255, y)) # x/255
data.as_numpy_iterator().next()
train_size = int(len(data)*.7)
val_size = int(len(data)*.2)
test_size = int(len(data)*.1)

train = data.take(train_size)
val = data.skip(train_size).take(val_size)
test = data.skip(train_size+val_size).take(test_size)

from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import LSTM, Dense, Flatten, Reshape
from tensorflow.keras.applications import ResNet50
from tensorflow.keras import layers
```

```
resnet_model = ResNet50(weights='imagenet', include_top=False,
input_shape=(256, 256, 3))

for layer in resnet_model.layers:
    layer.trainable = False

model = Sequential()
model.add(resnet_model)
model.add(layers.GlobalAveragePooling2D())
model.add(Reshape((1, -1)))
model.add(LSTM(128))
model.add(Dense(1, activation='sigmoid'))

# Compile the model
loss = 'binary_crossentropy'
model.compile(loss=loss, optimizer='adam', metrics=["accuracy"])

model.summary()

logdir='logs'
tensorboard_callback = tf.keras.callbacks.TensorBoard(log_dir=logdir)

hist = model.fit(train, epochs=5, validation_data=val, callbacks=[tensorboard_callback])

import matplotlib.pyplot as plt

# Assuming you have the 'history' object from model.fit
history = hist.history

# Plot training & validation accuracy values
plt.plot(history['accuracy'])
plt.plot(history['val_accuracy'])
plt.title('Model Accuracy')
plt.xlabel('Epoch')
plt.ylabel('Accuracy')
plt.legend(['Train', 'Validation'], loc='upper left')
plt.show()

fig = plt.figure()
plt.plot(hist.history['loss'], color='teal', label='loss')
```

```
plt.plot(hist.history['val_loss'], color='orange', label='val_loss')
fig.suptitle('Loss', fontsize=20)
plt.legend(loc="upper left")
plt.show()

fig = plt.figure()
plt.plot(hist.history['accuracy'], color='teal', label='accuracy')
plt.plot(hist.history['val_accuracy'], color='orange', label='val_accuracy')
fig.suptitle('Accuracy', fontsize=20)
plt.legend(loc="upper left")
plt.show()

loss, accuracy = model.evaluate(test)

print("Loss: ", loss)
print("Accuracy: ", accuracy)
```

CLIENT-SIDE CODING:

server.py

```
from flask import Flask, render_template, redirect, request, url_for, send_file
from flask import jsonify, json
from werkzeug.utils import secure_filename

import os
os.environ['KMP_DUPLICATE_LIB_OK']='True'

import torch
import torchvision

from torchvision import transforms

from torch.utils.data import DataLoader
from torch.utils.data.dataset import Dataset

import numpy as np
import cv2

import face_recognition

from torch.autograd import Variable

import time

import sys

from torch import nn

from torchvision import models

from skimage import img_as_ubyte
import warnings
warnings.filterwarnings("ignore")

UPLOAD_FOLDER = 'Uploaded_Files'
video_path = ""

detectOutput = []

app = Flask("__main__", template_folder="templates")
app.config['UPLOAD_FOLDER'] = UPLOAD_FOLDER
```

```
class Model(nn.Module):
  def__init__(self, num_classes, latent_dim= 2048, lstm_layers=1,
hidden_dim=2048 bidirectional=False):,
    super(Model, self).__init__()

    model = models.resnext50_32x4d(pretrained= True)

    self.model = nn.Sequential(*list(model.children())[:-2])

    self.lstm = nn.LSTM(latent_dim, hidden_dim, lstm_layers, bidirectional)

    self.relu = nn.LeakyReLU()

    self.dp = nn.Dropout(0.4)

    self.linear1 = nn.Linear(2048, num_classes)

    self.avgpool = nn.AdaptiveAvgPool2d(1)



  def forward(self, x):
    batch_size, seq_length, c, h, w = x.shape

    x = x.view(batch_size*seq_length, c, h, w)

    fmap = self.model(x)
    x = self.avgpool(fmap)
    x = x.view(batch_size, seq_length, 2048)
    x_lstm,_ = self.lstm(x, None)
    return fmap, self.dp(self.linear1(x_lstm[:,-1,:]))




im_size = 112

mean = [0.485, 0.456, 0.406]

std = [0.229, 0.224, 0.225]

sm = nn.Softmax()

inv_normalize = transforms.Normalize(mean=-1*np.divide(mean, std),
std=np.divide([1,1,1], std))
```

```
def im_convert(tensor):
  image = tensor.to("cpu").clone().detach()
  image = image.squeeze()
  image = inv_normalize(image)
  image = image.numpy()
  image = image.transpose(1,2,0)
  image = image.clip(0,1)
  cv2.imwrite('./2.png', image*255)
  return image

def predict(model, img, path='./'):
  fmap, logits = model(img.to())
  params = list(model.parameters())
  weight_softmax = model.linear1.weight.detach().cpu().numpy()
  logits = sm(logits)
  _, prediction = torch.max(logits, 1)
  confidence = logits[:, int(prediction.item())].item()*100
  print('confidence of prediction: ', logits[:, int(prediction.item())].item()*100)
  return [int(prediction.item()), confidence]


class validation_dataset(Dataset):
  def__init__(self, video_names, sequence_length = 60, transform=None):
    self.video_names = video_names
    self.transform = transform
    self.count = sequence_length

  def__len__(self):
    return len(self.video_names)

  def__getitem__(self, idx):
    video_path = self.video_names[idx]
    frames = []
    a = int(100 / self.count)
    first_frame = np.random.randint(0,a)
    for i, frame in enumerate(self.frame_extract(video_path)):
      faces = face_recognition.face_locations(frame)
      try:
        top,right,bottom,left = faces[0]
        frame = frame[top:bottom, left:right, :]
      except:
        pass
      frames.append(self.transform(frame))
      if(len(frames) == self.count):
```

```
            break
        frames = torch.stack(frames)
        frames = frames[:self.count]
        return frames.unsqueeze(0)

    def frame_extract(self, path):
        vidObj = cv2.VideoCapture(path)
        success = 1
        while success:
            success, image = vidObj.read()
            if success:
                yield image


def detectFakeVideo(videoPath):
    im_size = 112
    mean = [0.485, 0.456, 0.406]
    std = [0.229, 0.224, 0.225]

    train_transforms = transforms.Compose([
                                    transforms.ToPILImage(),
                                    transforms.Resize((im_size,im_size)),
                                    transforms.ToTensor(),
                                    transforms.Normalize(mean,std)])
    path_to_videos= [videoPath]

            video_dataset = validation_dataset(path_to_videos,sequence_length =
20,transform = train_transforms)


    model = Model(2)
    path_to_model = 'model/df_model.pt'
    model.load_state_dict(torch.load(path_to_model, map_location=torch.device('cpu')))
    model.eval()
    for i in range(0,len(path_to_videos)):
        print(path_to_videos[i])
        prediction = predict(model,video_dataset[i],'./')
        if prediction[0] == 1:
            print("REAL")
        else:
            print("FAKE")
    return prediction


@app.route('/', methods=['POST', 'GET'])
```

```
def homepage():
  if request.method == 'GET':
          return render_template('index.html')
  return render_template('index.html')


@app.route('/Detect', methods=['POST', 'GET'])
def DetectPage():
    if request.method == 'GET':
        return render_template('index.html')
    if request.method == 'POST':
        video = request.files['video']
        print(video.filename)
        video_filename = secure_filename(video.filename)
        video.save(os.path.join(app.config['UPLOAD_FOLDER'], video_filename))
        video_path = "Uploaded_Files/" + video_filename
        prediction = detectFakeVideo(video_path)
        print(prediction)
        if prediction[0] == 0:
            output = "FAKE"
        else:
            output = "REAL"
        confidence = prediction[1]

        data = {'output': output, 'confidence': confidence}
        data = json.dumps(data)
        os.remove(video_path);
        return render_template('index.html', data=data)


app.run(port=3000);
```

9.3 SAMPLE SCREENSHOTS

Figure 9.3.1 Sample Output (REAL)

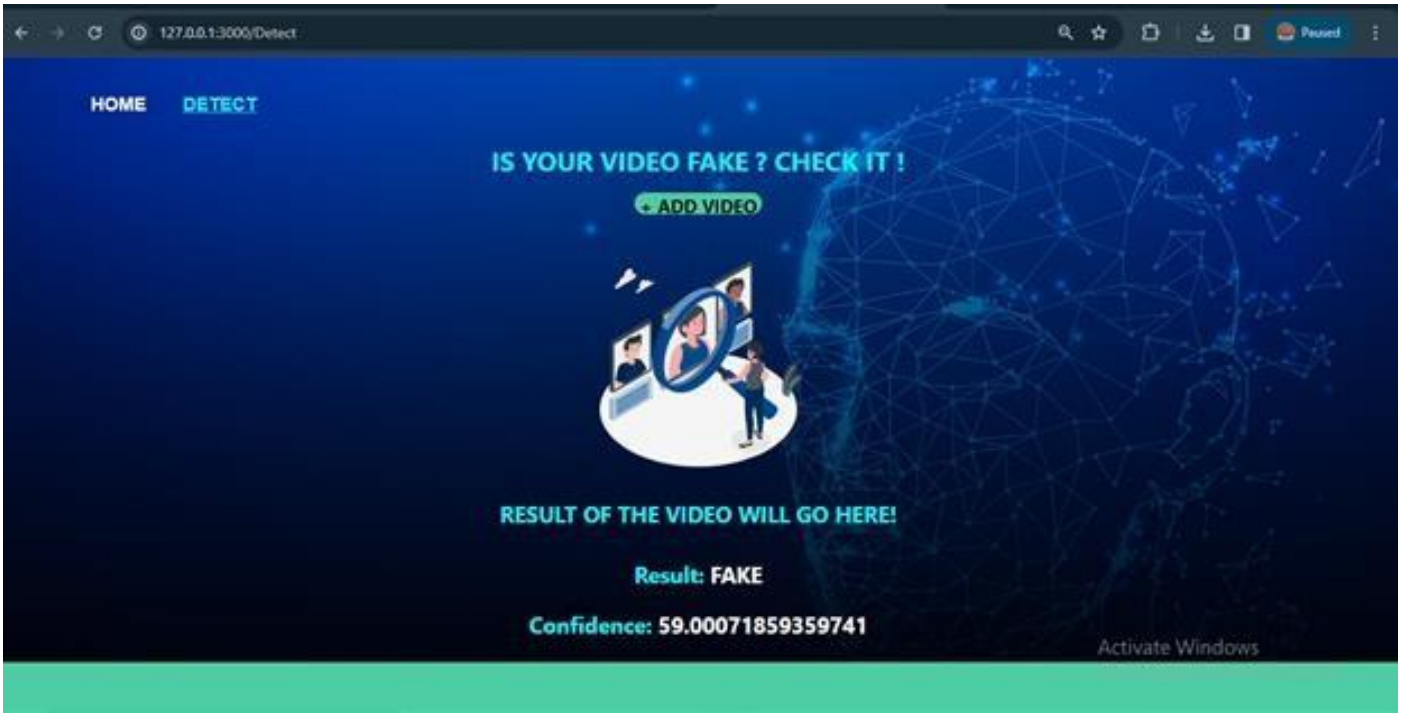


Figure 9.3.2 Sample Output (FAKE)

9.4 SDG GOALS

The Sustainable Development Goals (SDGs) are a collection of 17 global goals set by the United Nations General Assembly, intended to be a "blueprint to achieve a better and more sustainable future for all" by 2030. Deep fake detection intersects with several SDGs, particularly:

SDG 3 (Good Health and Well-being): Deep fake detection can help combat health misinformation, which is crucial for public health, especially during pandemics.

SDG 9 (Industry, Innovation, and Infrastructure): Advancements in deep fake detection technologies contribute to resilient infrastructure and foster innovation.

SDG 16 (Peace, Justice, and Strong Institutions): By preventing the spread of misinformation, deep fake detection supports peaceful societies and accountable institutions.

SDG 5 (Gender Equality): Deepfake detection helps combat gender-based cybercrimes, protecting women from digital exploitation and ensuring equal rights online.

SDG 9 (Industry, Innovation, and Infrastructure): By fostering innovation in AI and cybersecurity, deepfake detection contributes to building resilient infrastructure and promoting inclusive and sustainable industrialization.

1374

by KISHOREKUMAR V

Submission date: 18-Mar-2024 10:18AM (UTC+0530)
Submission ID: 2323375144
File name: 1374\Submission\ENHANCING DEEPFAKE DETECTION CONFERENCE PAPER.pdf (180.8K)
Word count: 3609
Character count: 20441

ENHANCING DEEPFAKE DETECTION: A MULTIMODAL APPROACH FOR IMPROVED ACCURACY

Karthikeyan A
Computer science dept
Panimalar Engineering college
Chennai, India

Kishorekumar V
Computer Science Dept
Panimalar Engineering College
Chennai, India

Monniesh B
Computer Science dept.
Panimalar Engineering College
Chennai, India

Niveshkumar S
Computer Science Dept
Panimalar Engineering College
Chennai, India

Abstract -Fake facial image identification is becoming more and more necessary as deepfake methods advance. Although a number of deepfake detection methods have been developed, it is still difficult to identify every kind of deepfake image using a single model. We provide a method for identifying different kinds of deepfake photos by utilising three typical traces that are produced by deepfakes: blur effects, residual noise, and warping artefacts. To find pixel-wise residual-noise traces, we used a network built for steganalysis. In order to capture high-level characteristics, we also take landmarks into consideration. These are the main facial features that frequently exhibit strange deformations in deepfake photos. Lastly, we use features from several picture quality evaluation tools that can catch traces of blurring, since the impact of a deepfake is comparable to that of blurring. The outcomes show that every detection method is effective, and that the suggested network performs steadily and better than other detection networks when tested on a variety of deepfake datasets.

Several deepfake detection approaches have been presented in this effort on picture forensics of generic image alterations. This used noise due to erroneous geometry and light predictions, as well as colour mismatch in two eyes. Deepfake employing the 2D direction discrepancy between the head's general circumference and restricted facial region. The two basic fake-face detection networks that are suggested make use of macroscopic characteristics. Since training uses an RGB colour space distribution, the colour space was converted to HSV, and deepfake was identified by utilising the statistical difference between the two colour spaces.

Keywords: Deepfakes, Machine Learning, Adversarial media detection, Machine learning for video forgery analysis, GAN, LSTM

I. INTRODUCTION

Digital video creation and distribution are now easier than ever because to the advancement of mobile camera technology and the expanding influence of social media and media sharing websites. Until recently, the number of fake videos and their degrees of realism has been limited by the lack of sophisticated editing tools, the high demand on domain expertise, and the complex and time-consuming process involved. However, the time of fabrication and manipulation of videos has decreased significantly in recent years, thanks to the accessibility to large-volume training data and high-through put computing power, but greater advancements in computer vision and machine learning methods, which do away with the necessity of manual editing processes. In particular, a new vein of AI-based fake video generation methods known as Deep Fake has attracted a lot of attention recently. It receives a video of a certain person (referred to as the "target") as input and generates a different movie in which the faces of the target are swapped out with those of the source). The video was divided into frames by the GAN, which then changed the input picture in each frame. It goes on to rebuild the video. By dividing the video into frames, extracting the features with a ResNext Convolutional Neural Network (CNN), and using a Recurrent Neural Network (RNN) with Long Short Term Memory (LSTM) to capture the temporal inconsistencies between frames introduced by GAN during the reconstruction of the Deepfake, our method detects such artefacts by comparing the generated face areas and

their surrounding regions. We streamline the procedure by directly modelling the resolution discrepancy in affine face wrappings in order to train the ResNext CNN model. Autoencoders are typically used to accomplish this operation. We present a novel deep learning approach that efficiently separates authentic films from Deepfake videos. Our approach is based on the same procedure that GAN uses to construct the Deepfake. The technique is based on the features of the Deepfake videos. The Deepfake algorithm can only synthesise face pictures of a certain size owing to production time and computational resource constraints; thus, the face images must undergo an affinal warp to fit the source's facial configuration Deep fake is based on automatically mapping the source's and target's Facial expressions using deep neural networks that have been trained on face pictures. With proper post-processing, the resulting videos can achieve a high level of realism The ever-expanding reach has been boosted by the sophistication of smartphone cameras and the global availability of reliable internet connections.

Digital video creation and transmission are now easier than ever thanks to social media and media sharing websites. Deep learning is becoming so strong that it was unthinkable just a few years ago due to the increase in processing power. Similar to every revolutionary invention, this has brought up new difficulties. The term "DeepFake" refers to the manipulation of audio and video footage by deep generative adversarial models. The DeepFake is now widely disseminated via social media platforms, which encourages spamming and the propagation of false information.

II. RELATED WORKS

1. Domain General Face Forgery Detection by Learning to Weight by Qixiang Ye, Yue Gao, Jianzhuang Liu, Ling Shao, Rongraong Ji, Ke Sun, Hong Liu. (2021): a domain-general model, termed learning-to-weight(LTW) that guarantees face detection performance across multiple domains.

2. Exploring Frequency Adversarial Attacks for face forgery Detection Shouhong by Ding, Xiaokang Yang (2022): an adversarial attack technique using frequency directed at face forgery detectors.Concretely, we apply discrete cosine transform (DCT) on the input images and introduce a fusion module to capture the salient region of adversary in the frequency domain

3. NAS-FAS: Static-Dynamic Central Diffrence Network Search For Face Anti-Spoofing By Zitong Yu, Jun Wan, Yunxiao Qin, Xiaobai Li, Stan Z. Li, and Guoying Zhao (2020): provide the first neural architecture search (NAS)-based FAS technique, known as NAS-FAS, to identify the most appropriatet ask-aware networks.

4. Learning Meta Pattern for Face Anti-Spoofing By Rizhao cai, zhi Li, Renjie Wan, Haoliang Li, Yongjian Hu, Alex C.Kot (2022): learnable network to extract Meta Pattern (MP) in our learningto-learn framework. The MP can train a more generalised model by substituting its discriminative information for handmade features.

5. Towards Generalaizable Deepfake Detection With Locality-Aware AutoEncoder By Mengnam Du, Shiva Pentyala, Yuening Li, Xia Hu (2020): The paper proposes Locality-Aware AutoEncoder (LAE) to bridge the generalization gap. Rather than collecting artefacts in the training set and learning superficial correlations to achieve detection, we apply a pixel-wise mask to regularise local interpretation of LAE during the training process, forcing the model to learn intrinsic representation from the forgery region.

6. A Robust Approach to Multimodal Deepfake Detection by Davide Salvi, Honggu Liu (2023): The peculiarity of the proposed method is that we never train on multimodal deepfake data, but using disjunct monomodal datasets with deepfakes that are simply auditory or visually presented. As multimodal datasets are scarce in the literature, it is desired that this spares us from using them during training.

7. Deepfake Generation and Detection: case Study and Challenges (2023): A comprehensive review of deepfake generation and detection and the different ML/DL approaches to synthesize deepfake contents and A special case study called IBMM is examined, providing a multi-modal summary of deepfake detection.

8. Deepfake Detection in Media Files-Audios, Images and Videos (2023): A system is proposed that can automatically detect the deepfake in media files such as images, videos, and audios which uses an image processing approaches combined with deep learning which detects the inconsistency that exists in fake media

9. DeepFake Videos Detection by Using recurrent Neural Network (RNN) By Ali Abdulzahra Mohsin Albazony (2023): The results show that the proposed model based on combining Convolutional Neural Networks, In comparison to the prior model, recurrent neural networks and image preprocessing approaches have been able to provide improved results in the classification and identification ofthe false video from the actual video.

III. EXISTING SYSTEM

The current system's backdrop is on the quick development of algorithms for face forgery, which produce altered photos and videos and aid in the spread of difficult-to-detect fraudulent material. The identification of face forgeries is becoming a more important area of emphasis for computer vision because to the serious concerns raised by these facial modification technologies. However, because of their inadequate generality across unknown domains, current detection algorithms frequently fail in practical applications. In order to solve this, the study presents Meta Deepfake Detection (MDD), a deep fake detection technique that uses meta-learning principles to create a model that can directly detect hidden domains that don't need to be updated frequently. To enable the creation of efficient domain representations, MDD applies meta-weight learning to adapt data from source domains to target domains. The approach involves creating meta-train and meta-test sets using a multi-domain strategy, and employs techniques like pair-attention loss and average-center alignment loss to enhance detection capabilities. The efficacy and universality of the approach are assessed against widely used deepfake datasets in relation to baseline techniques.

Digital forensics: This is the process of examining the media file's digital footprint to find changes or anomalies. For example, looking at metadata, compression artifacts, or irregularities in shadows and lighting can reveal information concerning tampering. Face Forensics: Disparities between authentic and fraudulent videos can be found using methods designed to examine facial characteristics such lip movements, facial emotions, eye blinking.

Audio analysis: As synthetic speech is a common component of deepfakes, examining aspects of audio such as intonation, cadence, and voice patterns might aid in the identification of altered audio. Human Expertise: To visually examine media content for indications of tampering, human specialists are sometimes hired. Although this method can be expensive and time-consuming, it can be useful for identifying complex deepfakes.

In conclusion, a range of techniques and technologies are employed by current deepfake detection algorithms to recognize altered media. These systems examine visual and audio clues suggestive of manipulation using a combination of cutting-edge machine learning algorithms and conventional forensic procedures. Modeling the temporal patterns found in video sequences, LSTM algorithms have been investigated for deepfake detection due to their capacity to identify temporal dependencies in sequential data. Though LSTM algorithms may work well for some deepfakes, they might not be enough to identify more complex alterations on their own.

IV. PROPOSED SYSTEM

We suggested a broad detection approach in this work to identify deepfake methods. We took use of the warping distortions, blur effects, and common traces of noise produced by the deepfake technique. We implemented them in the deepfake detection network that was suggested. To find residual noise, a steganalysis-specific network was first used as the basic network. Secondly, the semantic facial region was used to derive landmark patches. in order to identify warping artifacts, which are abnormal high-level characteristics. In order to capture the statistical properties of the blur-like effects of a deepfake, we finally utilized IQM features. The outcomes showed that every detection method works, and the suggested network's performance is better than the current networks. Our method may be immediately applied to deepfake video detection pipelines based on frame-by-frame detection since a deepfake video inherits residual characteristics from picture processes. Using the suggested methodology, we want to extend this investigation. to integrate a technique for detecting deepfake videos. We hope that this approach is resistant to time- and signal-based assaults.

We provide a broad detection technique that uses traces to identify three different kinds of deepfakes: attribute changes, puppet masters, and face swapping. We created a network using characteristics from image quality measurement (IQM) and distortion artifacts taken from landmarks on the face. We suggest that residual noise traces in deepfake pictures be captured using a network specifically engineered for steganalysis.

In summary, detect altered material, the suggested deepfake detection system combines cutting-edge machine learning algorithms with conventional forensic methodologies. Digital forensics, which examines information and compression artifacts, machine learning algorithms, which look for patterns suggestive of tampering, and face and audio analysis, which examines facial and speech characteristics, are important components. For verification reasons, the system may also use watermarking or blockchain technology. In order to guarantee the efficacy of the system and adjust to new deepfake approaches, stakeholders must collaborate and monitor it continuously.

V. SYSTEM ARCHITECTURE

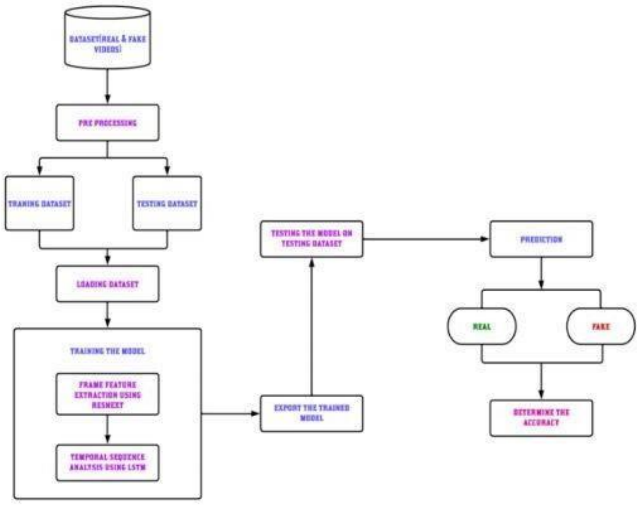


Fig. 1. System Architecture

VI. METHODOLOGY

1. Data Processing Module: Using human-pose estimation, deepfake video detection is the suggested method. Since the publicly accessible datasets do not meet the suggested facial and body language marker requirements, they are not suitable for use in this study. The only movies of world leaders (presidents, vice presidents, etc.) that we may use to train and test the stated hypothesis. These films have a decent pixel ratio and are generally accessible. We can test these videos as well. fast to combat deepfakes. Thus, in order to create a bespoke dataset, we manually retrieved movies from the internet. The suggested standards were followed in the annotation and labeling of these films. To evaluate the, we need two types of videos: the synthetic and the original. To make sure the videos haven't been altered, the original files were taken from Miller Center's official website. US presidents' speeches are streamed on this website. First, we selected films featuring the last four US presidents, George W. Bush, Joe Biden, Donald Trump, and Barack Obama. The prerequisites are satisfied by the video download. The following factors were present in the dataset samples of real and fake videos for deep fake training: (a) the file formats are MP4 with 30 frames per second (fps); (b) the video frames are similar throughout the dataset; (c) the subject of interest was in the middle of the frame; (d) there is no one else in the background; and (e) the cameras were reasonably stable.

2.Experimental setup: The notion that body languages may detect and reveal deep fakes was tested by building a many-to-one LSTM model with PyTorch. GPU support and customizable DNN implementation are offered by PyTorch. To set up the training and testing experiment, two types of objects are required: (a) an object for loading data, and (b) an object for creating a model. Ten percent was used for validation, ten percent for testing, and eighty percent was used for training the custom dataset. Additionally, the values were adjusted to match the range. A uniquely created (based on the requirements of the datasets to the LSTM model for training through the usage of our proprietary dataset) Data Loader. The model to be constructed uses a binary classification model with an input layer including 24 features, and the data only contains the body languages with 24 key points. Additionally, it has a single output unit and a completely linked output layer. At the last time step, the prediction is produced by the many-to-one LSTM model. Since every 150 poses are seen to be continuous movements, the model's time step is 150. The amount and the concealed size of The experiment should be used to determine hidden layers. Underfitting and overfitting issues may be avoided with the right network depth and width. The original LSTM design, which has only one hidden layer, served as the foundation for the first configuration. A dropout layer cannot come after a single concealed layer, resulting in a zero initial dropout.

3.Evaluation metrics: Metrics like loss and accuracy were used to assess the model's performance. For the experiments, the binary cross-entropy loss function was used. A predicted distribution that is closer to the target distribution is projected by a smaller testing loss value. To determine if the model was over- or under-fitted, we plotted the accuracy and loss with time. Conversely, overfitting denotes a model with poor generalization performance, while underfitting denotes a model that fails to converge. It is possible that insufficient training data led to this situation. The produced little dataset utilized for this research might cause overfitting if the confidence level is raised. As a result, we selected the model with the highest validation accuracy as the optimal model and modified the hyper-parameters based on both losses and accuracy.

In summary, preprocessing the video data by removing frames, features are retrieved using methods such as CNNs or optical flow. To represent temporal relationships in these sequences, LSTM algorithms are used. Metrics including accuracy, precision, recall, and F1 score are used to evaluate performance on an independent validation dataset. These metrics measure how well the model generalizes to new data. Real and fraudulent videos make up the diversified dataset that the system is trained on. To keep up with new developments in deepfake methods, upgrades and constant monitoring are carried out. To create reliable detection systems, researchers and industry professionals must work together.

VII. RESULT AND DISCUSSION

Whether the uploaded video is a real or deepfake video is the model's output. Following each pre-processing stage, the face-cropped frames are sent into the LSTM and ResNext neural networks. These chopped frames are employed to determine the authenticity of a video.A specific type of recurrent neural network called the Long Short-Term Memory (LSTM) algorithm was created to address difficulties in identifying and remembering long-term relationships in sequential input. Because of its

special design, memory cells, gates, and other features, long sequences of complex patterns may be recorded by LSTMs. Temporal dependencies may be learned and retained efficiently thanks to the regulation of information flow provided by three gates: the input gate, forget gate, and output gate. LSTMs are widely employed in many different sectors, but they are especially useful for time-series prediction problems like weather forecasting. This study uses long short-term memory (LSTM) to simulate the intricate associations seen in historical weather data. By addressing the sequential structure of meteorological variables, LSTMs offer a valuable tool for accurate monsoon forecasts. One of the preparation steps for the dataset is dividing the movie into frames. Face detection and cropping the frame with the detected face come next. The mean of the dataset video is determined in order to preserve the regularity in the number of frames, and a new processed face cropped dataset with the mean number of frames is constructed. During preprocessing, the frames without any faces are disregarded. It will take a lot of computing power to produce a 10-second video at 30 frames per second, or 300 total frames. Therefore, we are suggesting that the model be trained using only the first 100 frames for experimental purposes

VIII. CONCLUSION

In conclusion, this paper is used to create deepfakes and develop detection techniques. People are unable to distinguish fabricated photos or movies produced by Deepfake from authentic ones. Two machine learning models exit generative adversarial networks, which are used to construct deepfakes. While one model learns on a dataset, a second model looks for deepfakes. Up until the other model is unable to recognise the fake, the forger produces fakes. Fake news, photos, videos, and terrorist events are produced using deepfakes, which can lead to financial and social fraud. It is having a rising impact on democracy, security, culture, and religions as well as organisations, people, and communities. When the number of deepfake photos and videos on social media rises, people will stop believing the real thing. Therefore, cross-platform detection methods and deepfake datasets must be built in the future. The criteria given in the study may be used by the suggested approach to determine if the video is a deep fake or real. It is our belief that it will offer a extremely good accuracy with real-time data. We demonstrated an LSTM-based method that processes a one-second video clip accurately to determine if the footage is deepfake or real.

IX. FUTURE WORK

Future technological developments in areas like deep learning and machine learning will likely accelerate the democratisation of artificial intelligence, replacing manual procedures and processes with intelligent algorithms. AI democratisation will fundamentally change the way data is gathered and handled. The working algorithms will get smarter and more efficient with more extensive and nuanced data. This will make fraud detection less unclear and more exact. Deepfakes and other forms of fraud will be eliminated by extremely advanced AI systems. Because AI algorithms will develop to not only explain the fraud detection process but also provide evidence-based justifications for its judgements, it will create new paradigms in the field of fraud detection.

REFERENCES

[1]. K. Sun, H. Liu, Q. Ye, Y. Gao, J. Liu, L. Shao, and R. Ji, ‘‘Domain general face forgery detection by learning to weight,’’ in Proc. AAAI Conf. Artif. Intell., 2021, vol. 35, no. 3, pp. 2638–2646.

.

[2]. S. Jia, C. Ma, T. Yao, B. Yin, S. Ding, and X. Yang, ‘‘Exploring frequency adversarial attacks for face forgery detection,’’ in Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR), Jun. 2022, pp. 4103–4112.

[3]. Z. Yu, J. Wan, Y. Qin, X. Li, S. Z. Li, and G. Zhao, ‘‘NAS-FAS: Static-dynamic central difference network search for face anti-spoofing,’’ IEEE Trans. Pattern Anal. Mach. Intell., vol. 43, no. 9, pp. 3005–3023, Sep. 2021.

[4]. R. Cai, Z. Li, R. Wan, H. Li, Y. Hu, and A. C. Kot, ‘‘Learning meta pattern for face anti-spoofing,’’ IEEE Trans. Inf. Forensics Security, vol. 17, pp. 1201–1213, 2022.

[5]. M. Du, S. Pentyala, Y. Li, and X. Hu, ‘‘Towards generalizable DeepFake detection with locality-aware AutoEncoder,’’ in Proc. 29th ACM Int. Conf. Inf. Knowl. Manag., Oct. 2020, pp. 325–334.

[6]. M. Maung, A. Pyone, and H. Kiya, ‘‘Encryption inspired adversarial defense for visual classification,’’ in Proc. IEEE Int. Conf. Image Process. (ICIP), Oct. 2020, pp. 1681–1685.

[7]. Z. Wang, Y. Guo, and W. Zuo, ‘‘DeepFake forensics via an adversarial game,’’ IEEE Trans. Image Process., vol. 31, pp. 3541–3552, 2022

1374

ORIGINALITY REPORT

12%	10%	6%	1%
SIMILARITY INDEX	INTERNET SOURCES	PUBLICATIONS	STUDENT PAPERS

PRIMARY SOURCES

1	www.arxiv-vanity.com Internet Source	5%
2	Jihyeon Kang, Sang-Keun Ji, Sangyeong Lee, Daehee Jang, Jong-Uk Hou. "Detection Enhancement for Various Deepfake Types Based on Residual Noise and Manipulation Traces", IEEE Access, 2022 Publication	1%
3	www.ijitee.org Internet Source	1%
4	Robail Yasrab, Wanqi Jiang, Adnan Riaz. "Fighting Deepfakes Using Body Language Analysis", Forecasting, 2021 Publication	1%
5	www.semanticscholar.org Internet Source	1%
6	www.ijiemr.org Internet Source	1%
7	www.irjmets.com Internet Source	2%

8	Meta Deepfake Detection Model", IEEE Access, 2023 Publication	<1%
9	Submitted to Panimalar Engineering College Student paper	<1%
10	www.hindawi.com Internet Source	<1%
11	www.ijert.org Internet Source	<1%
12	Xu-Yao Zhang, Guo-Sen Xie, Xiuli Li, Tao Mei, Cheng-Lin Liu. "A Survey on Learning to Reject", Proceedings of the IEEE, 2023 Publication	<1%
13	export.arxiv.org Internet Source	<1%
14	oar.a-star.edu.sg Internet Source	<1%
15	www.researchgate.net Internet Source	<1%
16	Sepp Hochreiter, Jürgen Schmidhuber. "Long Short-Term Memory", Neural Computation, 1997 Publication	<1%
17	Yogesh Patel, Sudeep Tanwar, Rajesh Gupta, Pronaya Bhattacharya et al. "Deepfake	<1%

Generation and Detection: Case Study and Challenges", IEEE Access, 2023
Publication

Exclude quotes	On	Exclude matches	Off
Exclude bibliography	On		

REFERENCES

[1]. K. Sun, H. Liu, Q. Ye, Y. Gao, J. Liu, L. Shao, and R. Ji, ‘‘Domain general face forgery detection by learning to weight,’’ in Proc. AAAI Conf. Artif. Intell., 2021, vol. 35, no. 3, pp. 2638–2646.

[2]. S. Jia, C. Ma, T. Yao, B. Yin, S. Ding, and X. Yang, ‘‘Exploring frequency adversarial attacks for face forgery detection,’’ in Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR), Jun. 2022, pp. 4103–4112.

[3]. Z. Yu, J. Wan, Y. Qin, X. Li, S. Z. Li, and G. Zhao, ‘‘NAS-FAS: Static-dynamic central difference network search for face anti-spoofing,’’ IEEE Trans. Pattern Anal. Mach. Intell., vol. 43, no. 9, pp. 3005–3023, Sep. 2021.

[4]. R. Cai, Z. Li, R. Wan, H. Li, Y. Hu, and A. C. Kot, ‘‘Learning meta pattern for face anti-spoofing,’’ IEEE Trans. Inf. Forensics Security, vol. 17, pp. 1201–1213, 2022.

[5]. M. Du, S. Pentyala, Y. Li, and X. Hu, ‘‘Towards generalizable DeepFake detection with locality-aware AutoEncoder,’’ in Proc. 29th ACM Int. Conf. Inf. Knowl. Manag., Oct. 2020, pp. 325–334.

[6]. M. Maung, A. Pyone, and H. Kiya, ‘‘Encryption inspired adversarial defense for visual classification,’’ in Proc. IEEE Int. Conf. Image Process. (ICIP), Oct. 2020, pp. 1681–1685.

[7]. Z. Wang, Y. Guo, and W. Zuo, ‘‘DeepFake forensics via an adversarial game,’’ IEEE Trans. Image Process., vol. 31, pp. 3541–3552, 2022

[8]. B. Chen, T. Li, and W. Ding, ‘‘Detecting DeepFake videos based on spatiotemporal attention and convolutional LSTM,’’ Inf. Sci., vol. 601, pp. 58–70, Jul. 2022

[9]. Y. Ru, W. Zhou, Y. Liu, J. Sun, and Q. Li, ‘‘Bita-Net: Bi-temporal attention network for facial video forgery detection,’’ in Proc. IEEE Int. Joint Conf. Biometrics (IJCB), Aug. 2021, pp. 1–8

[10]. B. Dolhansky, J. Bitton, B. Pflaum, J. Lu, R. Howes, M. Wang, and C. C. Ferrer, ‘‘The DeepFake detection challenge (DFDC) dataset,’’ 2020, arXiv:2006.07397

[11] Agarwal, Shruti et al. “Watch Those Words: Video Falsification Detection Using Word-Conditioned Facial Motion.” 2023 IEEE/CVF Winter Conference on Applications of Computer Vision (WACV) (2021): 4699-4708.

[12] N. Khatri, V. Borar, and R. Garg, “A Comparative Study: Deepfake Detection Using Deep-learning,” 2023, doi: 10.1109/Confluence56041.2023.10048888.

[13] V. N. Tran, S. H. Lee, H. S. Le, B. S. Kim, and K. R. Kwon, “Learning Face Forgery Detection in Unseen Domain with Generalization Deepfake Detector,” in Digest of Technical Papers - IEEE International Conference on Consumer Electronics, 2023, vol. 2023-January, doi: 10.1109/ICCE56470.2023.10043436.

[14] V. H and T. G, “Antispoofing in face biometrics: a comprehensive study on software-based techniques,” Comput. Sci. Inf. Technol., vol. 4, no. 1, 2023, doi: 10.11591/csit. v4i1.p1-13.

[15] P. Gupta, C. Singh Rajpoot, and A. Professor, “A Deep Learning Technique based on Generative Adversarial Network for Heart Disease Prediction,” doi: 10.4186

[16] J. Peng, M. Sun, Z. Zhang, T. Tan, and J. Yan, “Efficient neural architecture transformation search in channel-level for object detection,” in Advances in Neural Information Processing Systems, 2019, vol. 32.

[17] Z. Shang, H. Xie, L. Yu, Z. Zha, and Y. Zhang, “Constructing Spatio-Temporal Graphs for Face Forgery Detection,” ACM Trans. Web, 2023, doi: 10.1145/3580512.

[18] A. Maclaughlin, J. Dhamala, A. Kumar, S. Venkatapathy, R. Venkatesan, and R. Gupta, Evaluating the Effectiveness of Efficient Neural Architecture Search for Sentence-Pair Tasks.

[19] A. Hesham, Y. Omar, E. El-fakharany, and R. Fatahillah, “A Proposed Model for Fake Media Detection Using Deep Learning Techniques,” in Lecture Notes on Data Engineering and Communications Technologies, vol. 152, 2023.