IJARCCE ISSN (O) 2278-1021, ISSN (P) 2319-5940

International Journal of Advanced Research in Computer and Communication Engineering

Impact Factor 8.102 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 13, Issue 6, June 2024

DOI: 10.17148/IJARCCE.2024.13618

“DEEPFAKE MULTIMEDIA DETECTION USING DEEP LEARNING”

Sandya P[1], Moksha B Anekar[2], Nithya SS[3], Sagar S[4], Dr. Suma R[5]

UG Students, Dept. of I.S.E., SSIT, Sri Siddhartha Academy of Higher Education, Tumkur, Karnataka[1-4]

Assistant Professor, Dept. of I.S.E., SSIT, Sri Siddharth Academy of Higher Education, Tumkur, Karnataka[5]

Abstract: As the proliferation of deep fake content continues to pose a growing threat to the integrity of multimedia, this paper introduces a robust approach for deepfake detection leveraging a hybrid architecture. The proposed framework seamlessly integrates the power of Residual Networks (ResNet) for spatial feature extraction and Long Short-Term Memory (LSTM) with Convolutional Neural Networks (CNN) for modeling temporal dependencies. The ResNet component adeptly captures intricate patterns in facial and contextual information, while the LSTM-CNN module focuses on discerning dynamic facial expressions and movements over sequential frames. Transfer learning strategies are employed to bolster model generalization, combining pre-training on a large-scale dataset with fine-tuning on deepfake-specific data. Experimental evaluations on diverse deepfake datasets demonstrate superior performance in accuracy, precision, and recall, establishing the efficacy of the hybrid architecture in addressing the evolving challenges posed by increasingly sophisticated deepfake generation techniques.

Keywords: Resnet, LSTM (Long Short Term Memory), CNN (Convolutional Neural Network), Deep learning, Tensor flow.

I.INTRODUCTION

The rapid advancement of artificial intelligence, particularly in the realm of deep learning, has led to the emergence of deepfake technology, allowing the creation of hyper-realistic multimedia content that can deceive human perception. The consequences of maliciously deployed deepfakes range from misinformation and reputation damage to potential threats to national security. As a countermeasure to this escalating challenge, the development of effective deepfake detection methods has become imperative. This paper introduces a novel approach to deepfake detection, combining the strengths of Residual Networks (ResNet) and Long Short-Term Memory (LSTM) with Convolutional Neural Networks (CNN) to create a hybrid architecture that excels in capturing both spatial and temporal features.

The proposed hybrid model aims to address the limitations of existing deepfake detection techniques by leveraging the comprehensive spatial understanding offered by ResNet and the nuanced temporal dependencies modeled through LSTM-CNN fusion. With the exponential growth in deepfake sophistication, conventional methods often struggle to discern subtle manipulations in facial features and fail to capture the temporal dynamics inherent in video sequences. In response, our approach not only integrates these two powerful neural network architectures but also employs transfer learning strategies to enhance generalization, enabling the model to adapt effectively to the diverse and evolving landscape of deepfake creation methods. The subsequent sections detail the architecture, methodology, and experimental results, illustrating the efficacy of our proposed solution in the challenging task of deepfake detection.

II.RELATED WORK

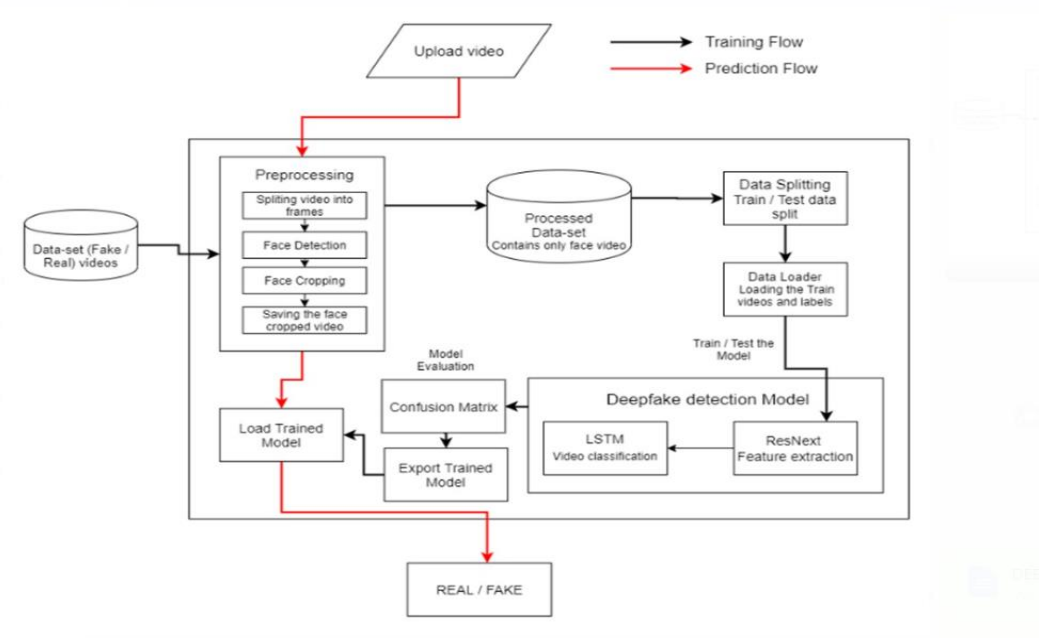
The current landscape of deepfake detection methods primarily relies on traditional convolutional neural networks (CNNs) or isolated recurrent neural networks (RNNs) for spatial and temporal information processing, respectively. While these approaches exhibit a degree of effectiveness, they often fall short in capturing the nuanced interplay between spatial and temporal features inherent in deepfake content. Additionally, many existing systems lack robustness against adversarial attacks, compromising their real-world applicability. The drawbacks of these methods underscore the need for a more comprehensive and adaptive approach.

III.PROPOSRD WORK

The proposed methodology involves the integration of Residual Networks (ResNet) for spatial feature extraction and Long Short-Term Memory (LSTM) with Convolutional Neural Networks (CNN) for modeling temporal dependencies. This hybrid architecture aims to synergistically combine the strengths of ResNet's ability to capture intricate spatial patterns with LSTM-CNN's proficiency in discerning dynamic temporal changes in video sequences. Transfer learning strategies will be employed to pre-train the model on a diverse dataset, ensuring a robust foundation, followed by fine-tuning on deepfake-specific datasets to enhance its adaptability to evolving deepfake generation techniques. The proposed methodology seeks to overcome the limitations of existing systems by providing a more comprehensive and adaptive solution to the intricate challenges posed by deepfake detection.

MODEL OF PROPOSED WORK

BLOCK DIAGRAM


Upload video
Training Flow
Prediction Flow
Data-set (Fake / Real) videos
Preprocessing
Spliting video into frames
Face Detection
Face Cropping
Saving the face cropped video
Processed Data-set Contains only face video
Data Splitting Train / Test data split
Data Loader Loading the Train videos and labels
Train / Test the Model
Model Evaluation
Confusion Matrix
Deepfake detection Model
LSTM Video classification
ResNext Feature extraction
Load Trained Model
Export Trained Model
REAL / FAKE

1. **Preprocessing:**

Face Detection and Alignment:

Utilize a pre-trained face detection model (e.g., MTCNN or Haarcascades) to locate faces in each image or frame. Align detected faces to a standard reference frame to ensure consistent positioning.

Normalization and Resizing:

Normalize pixel values to a standard range for better convergence during training.Resize images or frames to a fixed resolution to ensure uniformity across the dataset.

Data Augmentation:

Apply data augmentation techniques such as rotation, flipping, and slight variations in brightness and contrast to increase model robustness.
Generate augmented images to expand the dataset for improved generalization.

2. **Data Splitting between Train and Test:**

Random Split:

Randomly split the preprocessed dataset into training and testing set. Common split ratios are 80-20 or 70-30 for training and testing, respectively.

Stratified Split:

DOI: 10.17148/IJARCCE.2024.13618

Class distribution in both sets. If the dataset has imbalanced classes (more genuine than deepfake images, or vice versa), consider using a stratified split to maintain

3. Feature Extraction by ResNet:

Loading Pre-trained ResNet Model:

Utilize a pre-trained ResNet model (e.g., ResNet50) from a deep learning framework like TensorFlow or PyTorch. Load the pre-trained weights to leverage knowledge gained from a large dataset.

Extracting Features:

Pass preprocessed images through the modified ResNet to obtain feature maps.These feature maps represent high-level spatial features learned by ResNet.

4. Model Building (CNN + LSTM):

Designing LSTM-CNN Fusion:

Stack convolutional layers with pooling and normalization for spatial analysis.Integrate LSTM layers to capture temporal dependencies across sequential frames.

Combining ResNet and LSTM-CNN:

Connect the output of the ResNet (feature maps) to the input of the LSTM-CNN fusion module. Ensure compatibility in the number of features between the ResNet and LSTM-CNN components.

5. Testing Input Image:

Single Image Test:

For testing a single input image, follow the same preprocessing steps as for the training data.Pass the preprocessed image through the ResNet and subsequent LSTM-CNN module in the trained model.

Result Interpretation:

Obtain the model's output, which represents the probability of the input being a deepfake.Set a threshold to classify the image as either genuine or a deepfake based on the output probability

IV.PSEUDO CODE

```
import Flask, render_template, url_for, request
import sqlite3
import os
from image_test import *
from video_test import *
from audio_test import *
connection = sqlite3.connect('user_data.db')
cursor = connection.cursor()
command = """CREATE TABLE IF NOT EXISTS user(name TEXT, password TEXT, mobile TEXT, email TEXT)"""
cursor.execute(command)
app = Flask(_name_)
@app.route('/')
def index():
    return render_template('index.html')
@app.route('/home')
```

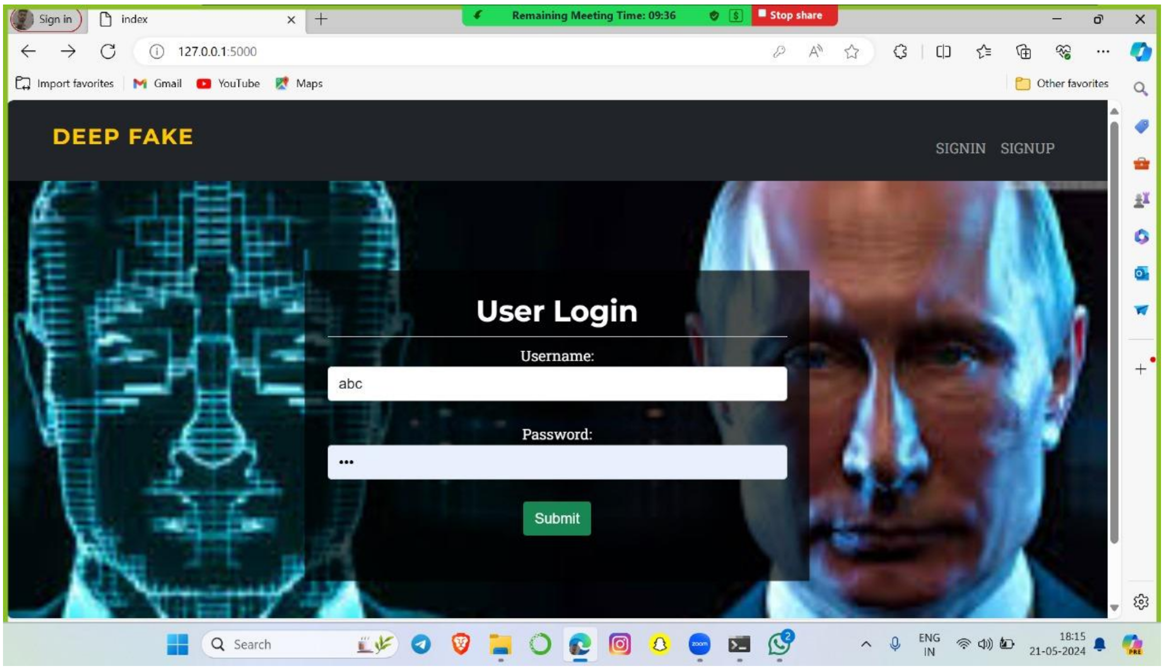
DOI: 10.17148/IJARCCE.2024.13618

```
def home():
  return render_template('home.html')
@app.route('/userlog', methods=['GET', 'POST'])
def userlog():
  if request.method == 'POST':

    connection = sqlite3.connect('user_data.db')
    cursor = connection.cursor()
    name = request.form['name']
    password = request.form['password']
    query = "SELECT name, password FROM user WHERE name = '"+name+"' AND password= '"+password+"'"
    cursor.execute(query)
    result = cursor.fetchall()
    if len(result) == 0:
      return render_template('index.html', msg='Sorry, Incorrect Credentials Provided,  Try Again')
    else:
      return render_template('home.html')
  return render_template('index.html')
@app.route('/userreg', methods=['GET', 'POST'])
def userreg():
  if request.method == 'POST':
   connection = sqlite3.connect('user_data.db')
    cursor = connection.cursor()
    name = request.form['name']
    password = request.form['password']
    mobile = request.form['phone']
    email = request.form['email']
    print(name, mobile, email, password)

 command = """CREATE TABLE IF NOT EXISTS user(name TEXT, password TEXT, mobile TEXT, email TEXT)"""
    cursor.execute(command)
    cursor.execute("INSERT INTO user VALUES ('"+name+"', '"+password+"', '"+mobile+"', '"+email+"')")
    connection.commit()
   return render_template('index.html', msg='Successfully Registered'
  return render_template('index.html')
@app.route('/detectimage', methods=['GET', 'POST'])
def detectimage():
  if request.method == 'POST':
    src = "static/imgs/"+request.form['src']
    dst = "static/imgs/"+request.form['dst']
    out = "static/image_output/output.jpg"
    command = f"python main.py --src {src} --dst {dst} --out {out} --correct_color"
    os.system(command)
    return render_template('Image.html', inputimage=dst, outputimage=out)
  return render_template('Image.html')
@app.route('/detectvideo', methods=['GET', 'POST'])
def detectvideo():
  if request.method == 'POST':
    src = "static/imgs/"+request.form['src']
    dst = "static/videos/"+request.form['dst']
    out = "static/video_output/output.mp4"

 command = f"python main_video.py --src_img {src} --video_path {dst} --show --correct_color --save_path {out}"
    os.system(command)
   return render_template('Video.html', inputvideo=dst, outputvideo=out)
  return render_template('Video.html')
@app.route('/detectlive', methods=['GET', 'POST'])
def detectlive():
  if request.method == 'POST':
    src = "static/imgs/"+request.form['src']
    out = "static/video_output/output.mp4"
    command = f"python main_video.py --src_img {src} --show --correct_color --save_path {out}"
    os.system(command)
    return render_template('Live.html', inputvideo='static/videos/input.mp4', outputvideo=out)
  return render_template('Live.html')
```

DOI: 10.17148/IJARCCE.2024.13618

```
@app.route('/detection')
def detection():
    return render_template('testimage.html')
@app.route('/testimage', methods=['GET', 'POST'])
def testimage():
    if request.method == 'POST':
        src = "static/imgs/"+request.form['src']
        out = "static/testimage_output/output.jpg"
        process_image(src, out)

 return render_template('testimage.html', inputimage=src, outputimage=out)
    return render_template('testimage.html')
@app.route('/testvideo', methods=['GET', 'POST'])
def testvideo():
    if request.method == 'POST':
        src = "static/videos/"+request.form['src']
        out = "static/testvideo_output/output.mp4"
        process_video(src, out)
        return render_template('testvideo.html', inputvideo=src, outputvideo=out)
    return render_template('testvideo.html')
@app.route('/testaudio', methods=['GET', 'POST'])
def testaudio():
    if request.method == 'POST':
        src = "static/audio/"+request.form['src']
        out = runtest(src)
        return render_template('testaudio.html', inputaudio=src, output=out)
    return render_template('testaudio.html')
@app.route('/logout')
def logout():
    return render_template('index.html')
if _name_ == "_main_":
    app.run(debug=True, use_reloader=False)
```

V.SIMULATION RESULT



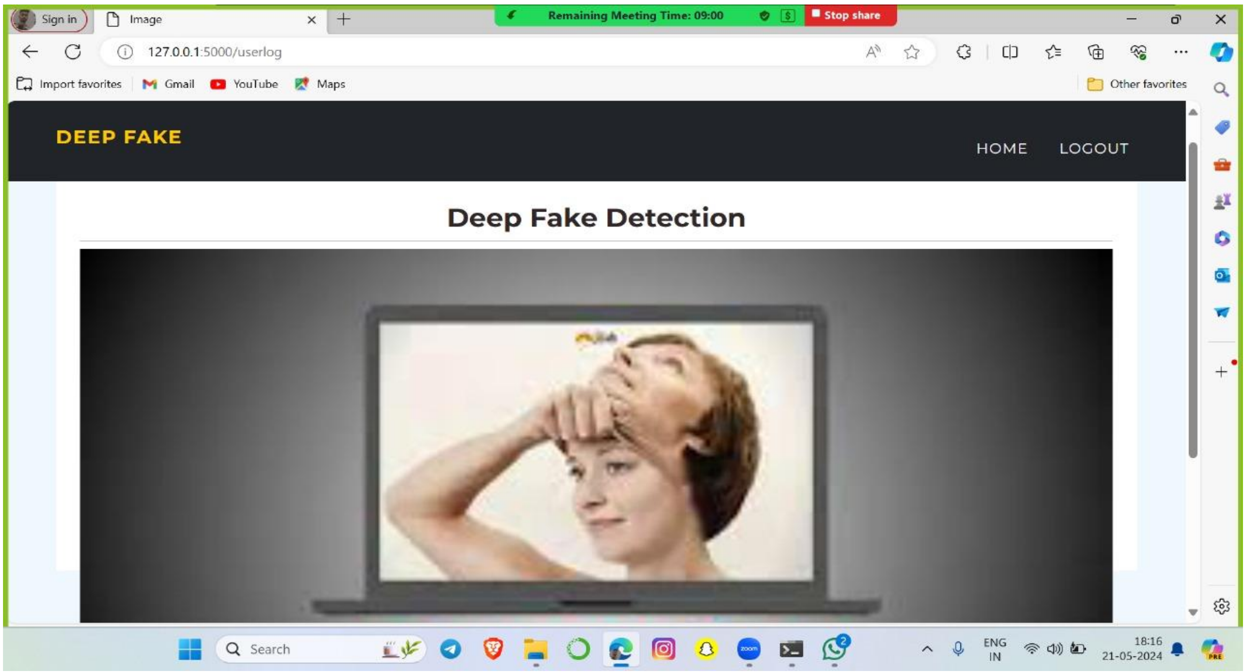
User Login

Username:

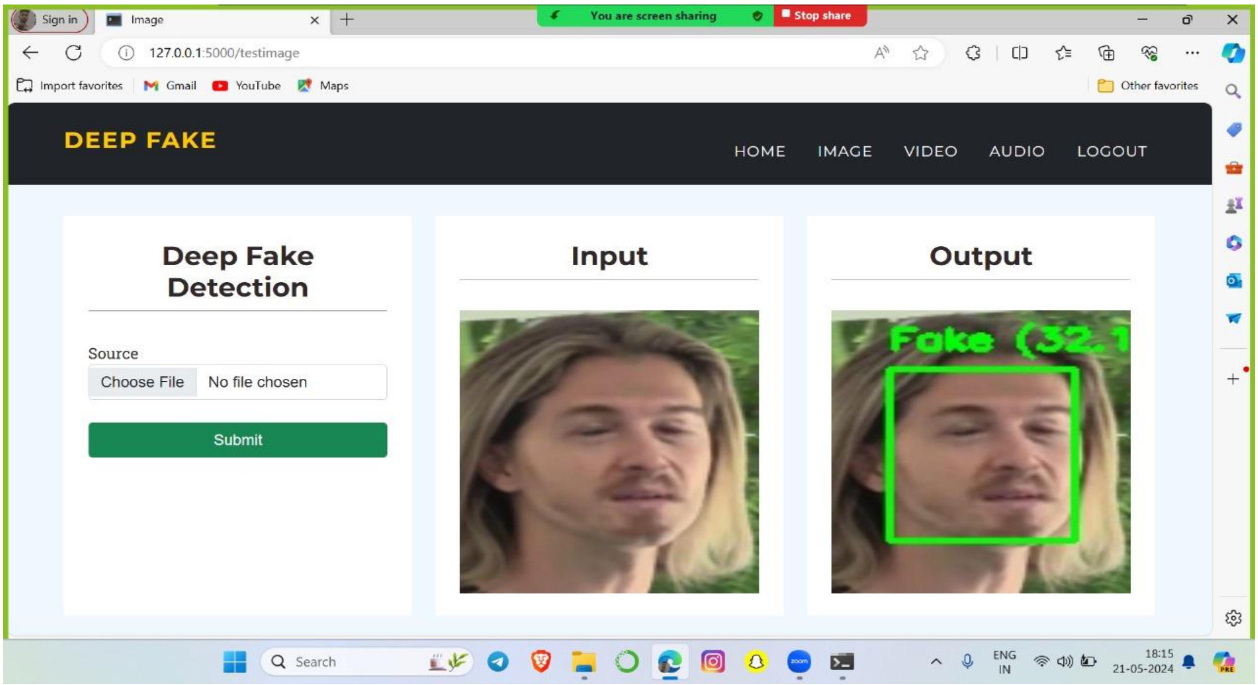
abc

Password:

•••



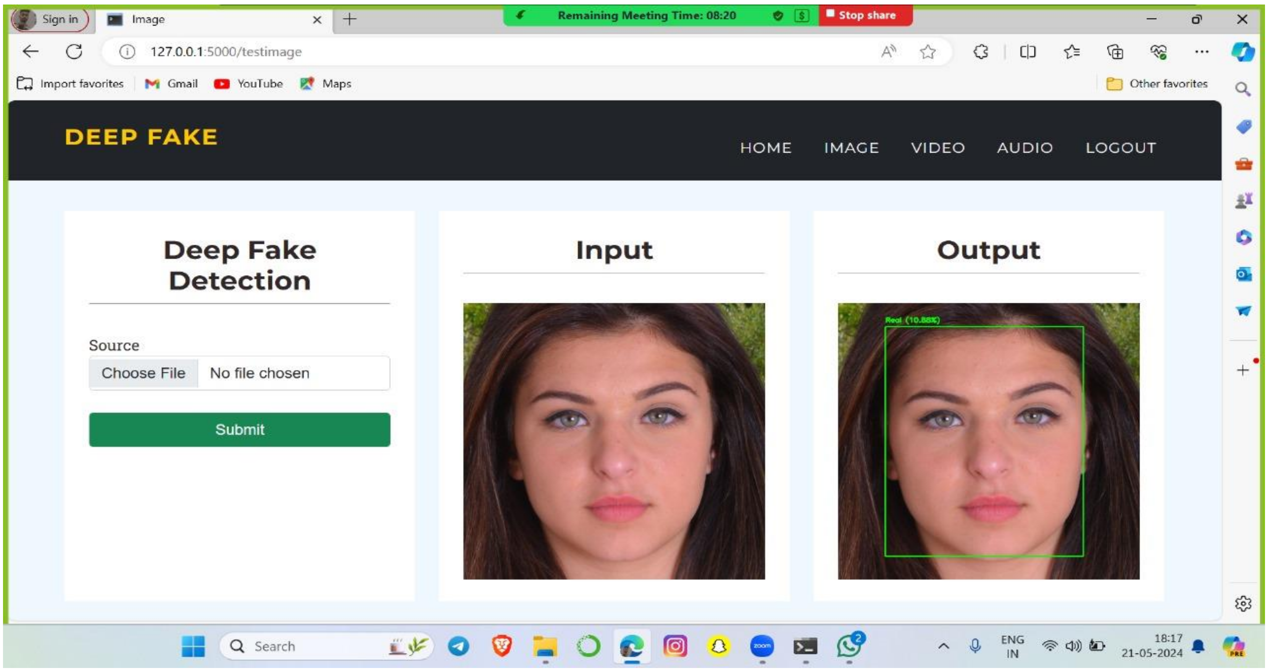
Deep Fake Detection



Deep Fake Detection

Input

Output



Deep Fake Detection

Source

Choose File No file chosen

Submit

VI. CONCLUSION AND FUTURE WORK

Deepfakes have started to dissolve the trust of individuals in media substance as observing them is not, at this point proportionate with putting stock in them. They could make the pain and negative impacts those focused on, increase disinformation and abhor discourse, and even could animate political strain, excite general society, savagery, or war. This is particularly basic these days as the advancements for making deepfakes are progressively agreeable furthermore, online networking stages can spread those phony substances rapidly. Sometimes deepfakes do not need to be spread to the massive audiences to cause detrimental effects. People who create deepfakes with a malicious purpose only need to deliver them to target audiences as part of their sabotage strategy without using social media.

People need to be more careful about what they see and they should have the ability to understand whether the content is fake or real. Recordings and photographs have been broadly utilized as confirmations in police examination and equity cases. They might be presented as confirmations in an official courtroom by computerized media criminology specialists who have a foundation in PC or law implementation and involvement with gathering, looking at, and breaking down advanced data. The improvement of ML and AI advances may have been utilized to adjust this computerized substance and in this manner, the specialists' suppositions may not be sufficient to confirm these confirmations in light of the fact that even specialists can't perceive controlled substances.

REFERENCES

[1]. Oleg Alexander, Mike Rogers, William Lambeth, Jen-Yuan Chiang, Wan-Chun Ma, Chuan-Chang Wang, and Paul Debevec. The Digital Emily project: Achieving a photorealistic digital actor. IEEE Computer Graphics and Applications, 30(4):20–31, 2021.

[2]. Antreas Antoniou, Amos J. Storkey, and Harrison Edwards. Augmenting image classifiers using data augmentation generative adversarial networks. In Artificial Neural Networks and Machine Learning - ICANN, pages 594–603, 2021.

[3]. A. K. Jain, A. Ross, and S. Prabhakar, ‘‘An introduction to biometric recognition,’’ IEEE Trans. Circuits Syst. Video Technol., vol. 14, no. 1, pp. 4–20, Jan. 2022.

[4]. DeepFake Detection Using Deep Learning: A Review **by** Zhang, X., Zhuang, Y., Ding, L., & Shi, Year: 2020.

[5]. FaceForensics++: Learning to Detect Manipulated Facial Images **,**Rossler, A., Cozzolino, D., Verdoliva, L., Riess, C., Thies, J., & Nießner, M.Year: 2021.

[6]. Towards Automatic Detection of Deepfake Videos: A Review**,** Al-Qershi, O. M., Khoo, B. E., & See, J.,**Year**: 2020.

[7]. Deep Residual Learning for Image Recognition**,** He, K., Zhang, X., Ren, S., & Sun, J.Year: 2022.

[8]. Detecting Deepfake Videos from Gaze Analysis, Nguyen, P. H., Tran, D. A., & Duong, D. A.Year: 2021