DEEPFAKE DETECTION

FOR HUMAN FACE IMAGES AND VIDEOS

GROUP MEMBERS

ASMI FAISEL

JOSIN JOSE

NANDHANA A

V H PRANAV

GUIDED BY

Ms. Nimitha Mary Mohan

Asst.Professor

Dept.of CSE

INTRODUCTION

- Deepfake is a technique for fake media synthesis based on AI.
- Deepfakes are created by combining and superimposing existing images and videos onto source images or videos using a deep learning technique , GAN.
- This project explores the technology behind deepfakes, the growing threat they pose, and the innovative methods and tools developed to combat this formidable challenge.

ABSTRACT

- The growing computation power has made creating an indistinguishable synthesized video called as deepfakes very simple.

- Scenarios where deepfakes are used to create political distress, fake news, revenge porn, financial fraud are becoming common.

- In response to the growing concern over deepfake technology's impact on media credibility, the project presents a comprehensive DFD system.

LITERATURE REVIEW

Sl No.	Paper	Published Year	Source	Advantages and Disadvantages
1	**Detecting GAN generated Fake Images using Co-occurrence matrices**	2019	**arXiv**	**Advantages:** • Shows **robustness** against JPEG compression. • provides an **end-to-end framework** for detecting GAN-generated fake images, integrating co-occurrence matrices directly into a deep learning architecture**.** **Disadvantages:** • may not be able to detect deepfakes that are generated using a different GAN architecture. • Implementing such a system may require significant computational resources and expertise.
2	**Deepfake Video Detection Using Recurrent Neural Networks**	2020	IEEE EXPLORE	**Advantages:** the system can capture temporal inconsistencies introduced by face-swapping processes**.**It is robust to variations in illumination and pose. **Disadvantages:**The performance of the system is highly dependent on the quality and diversity of the training data**.**

Sl No.	Paper	Published Year	Source	Advantages and Disadvantages
3	**Deepfake Detection Through Deeplearning**	2020	IEEE EXPLORE	**Advantages:** 1)Xception has a high performance based On its benchmarking performance on the FaceForensics test Environment 2)Xception shown a relatively good performance over four different datasets **Disadvantages:** This method is useful when the corresponding detection model is used for Detection of each type of fake video. If the correspondence Between the model and the testing data was not followed, it is imapcted worsly.
4	**Exposing Deep Fakes Using Inconsistent Head Poses**	2020	IEEE EXPLORE	**Advantages: *** It is relatively fast and efficient.It is robust to variations in facial appearance. Since the method is based on analyzing facial landmarks and head poses, it can potentially be applied to a wide range of media formats, including images and videos **Disadvantages**:* It may not be able to detect deepfakes that are generated using very sophisticated methods.It may not work well on deepfakes that are edited to make the head pose consistent

Sl No.	Paper	Published Year	Source	Advantage and Disadvantage
5	**FaceForensics++: Learning to Detect Manipulated Facial Images**	2019	IEEE EXPLORE	**Advantages:** • The FaceForensics++ dataset is the largest publicly available dataset for facial manipulation detection. • The model is robust to different types of facial manipulations, including Face2Face, FaceSwap, DeepFakes, and NeuralTextures. **Disadvantages:** • The model may not be able to detect deepfakes that are generated using new or unknown methods.
6	**Methods of Deepfake Detection Based on Machine Learning**	2020	IEEE EXPLORE	**Advantages:** Artifacts on small moving parts such as hairs, eyebrows, eyelashes can help identify Deepfake manipulation better. **Disadvantages:** indicators such as skin smoothness or color mismatches, may have limitations in detecting Deepfake manipulation.

SYSTEM REQUIREMENTS

Hardware Specification

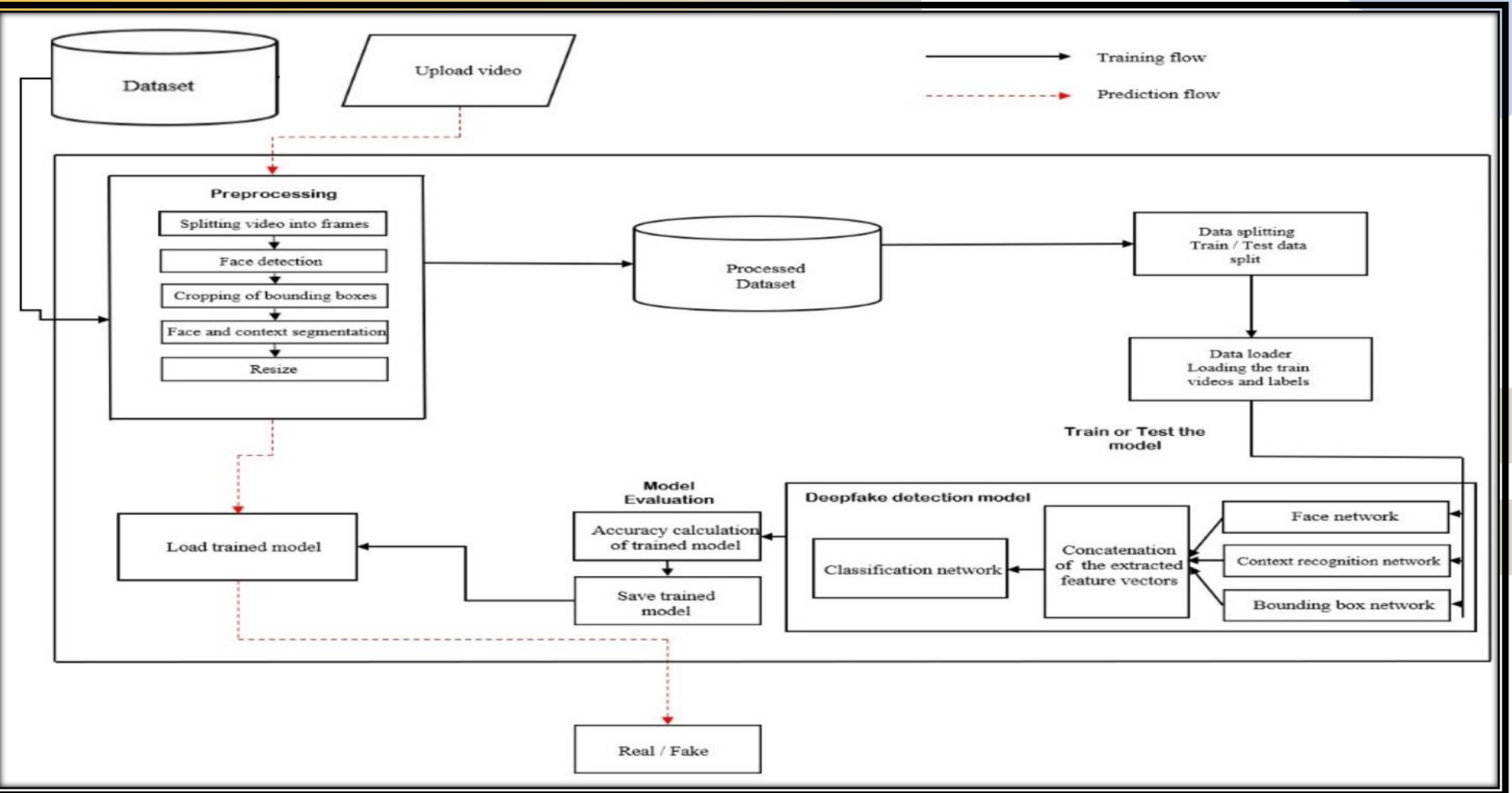
- Processor: i5 or i7
- RAM: 8GB (Minimum)
- Storage: 500GB SSD or above

Software Specification

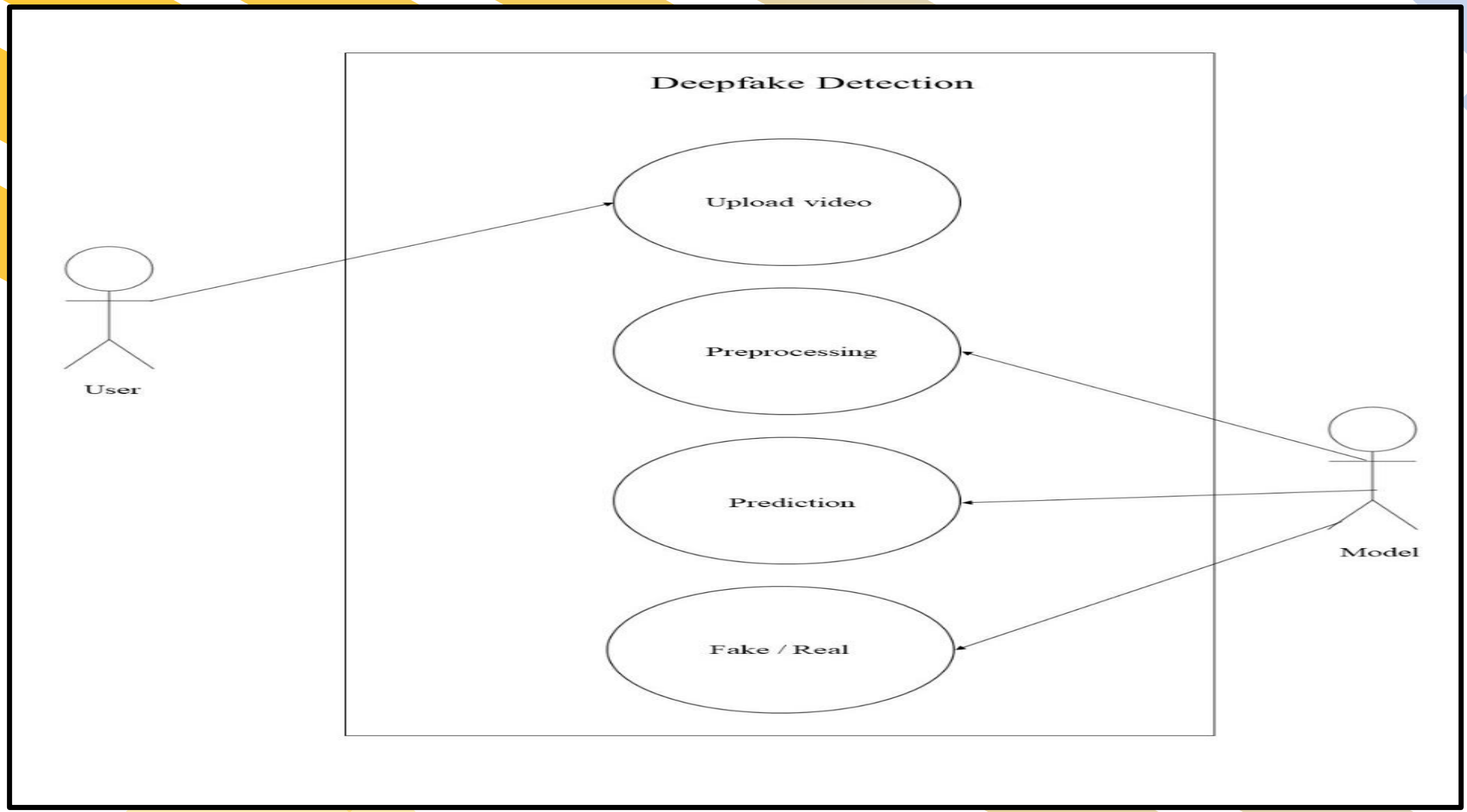
- Tool: Python IDLE, Anaconda
- Python: version3
- Libraries : Tensorflow, OpenCV, Numpy, Tkinter

DESIGN DIAGRAMS

BLOCK DIAGRAM

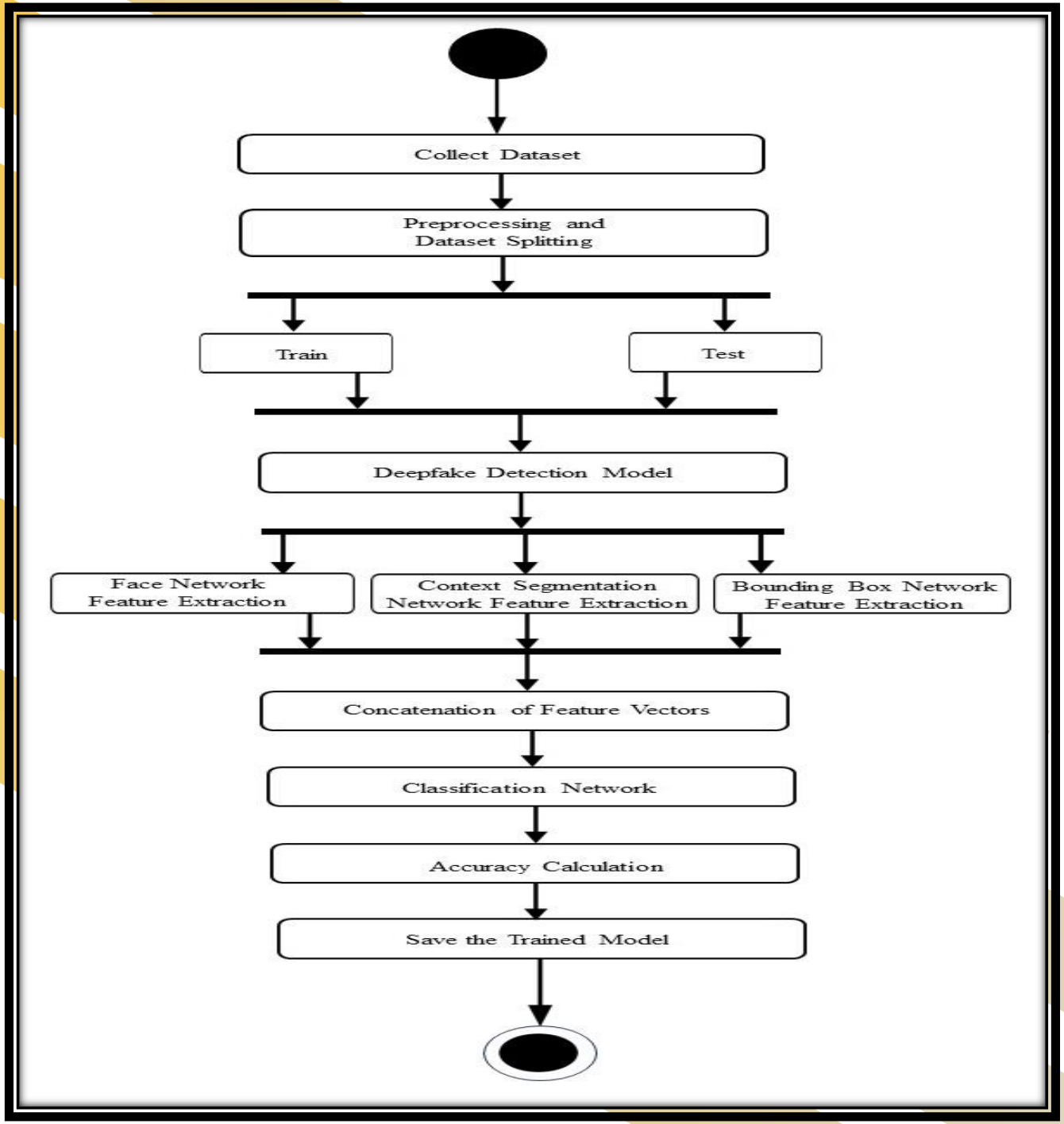


USE CASE DIAGRAM



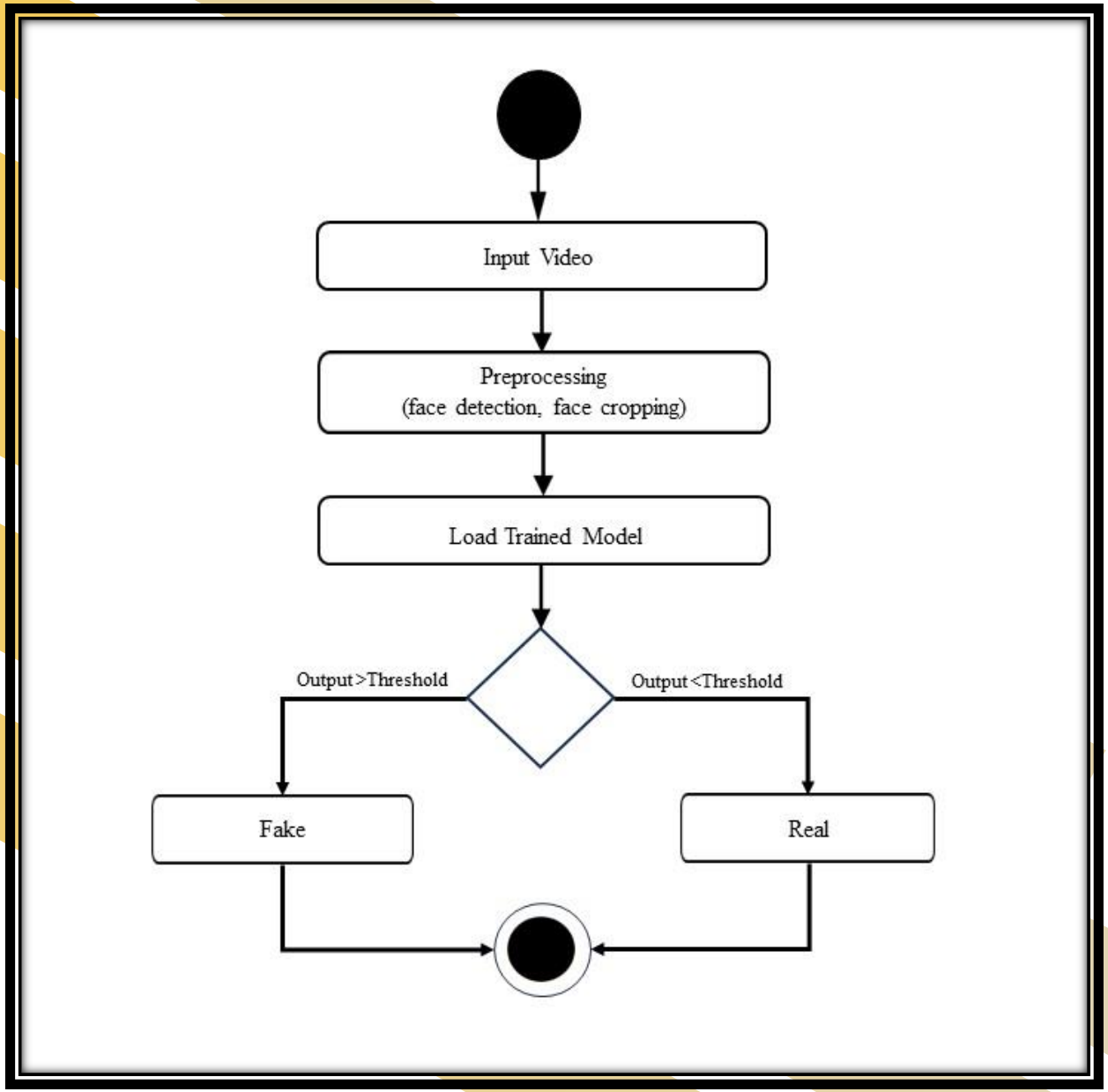
ACTIVITY DIAGRAM

1) **Training Workflow**

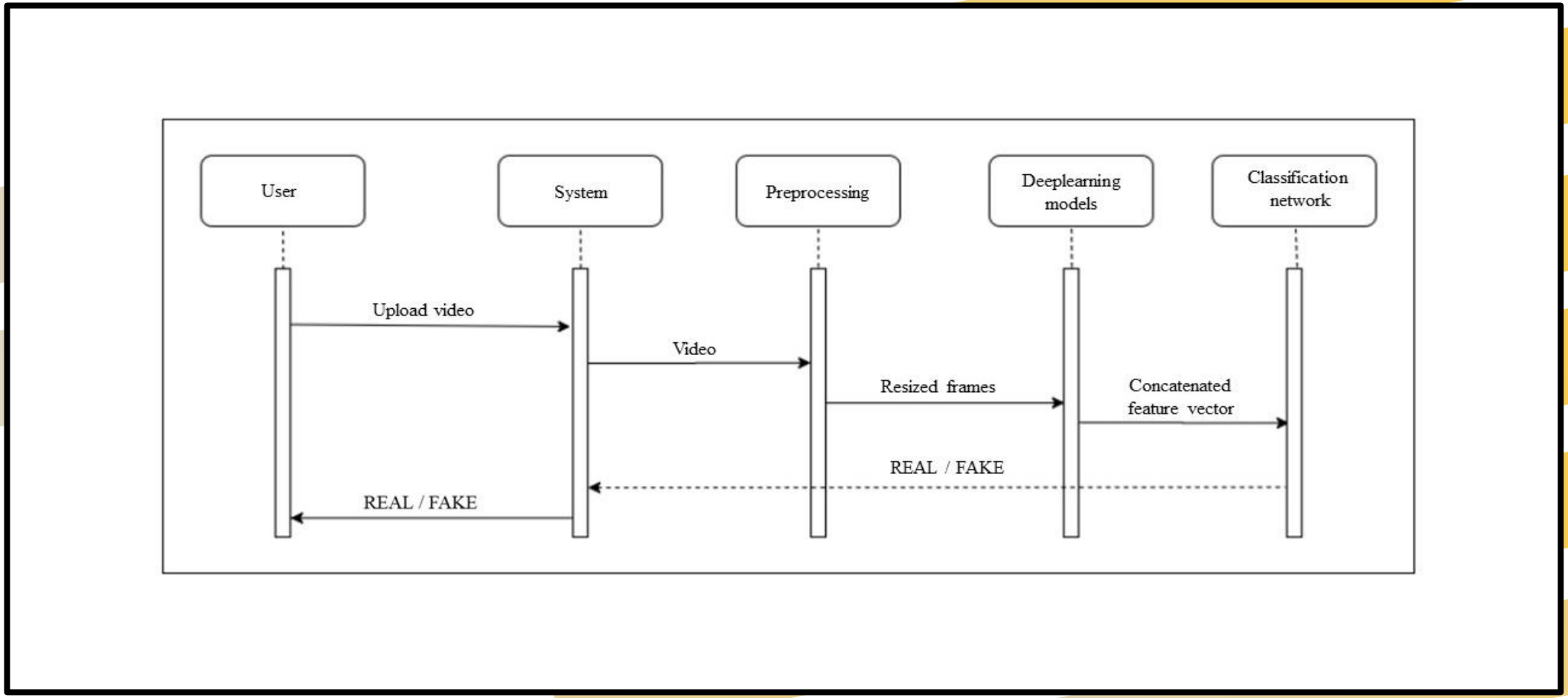


ACTIVITY DIAGRAM

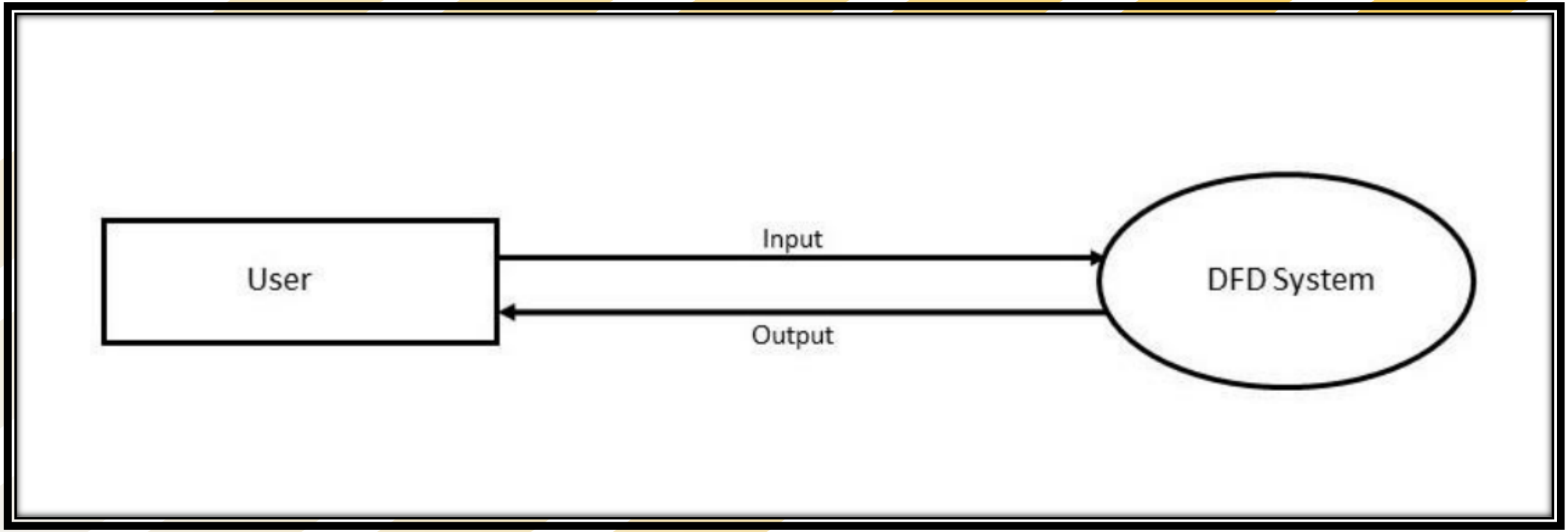
2) **Prediction workflow**



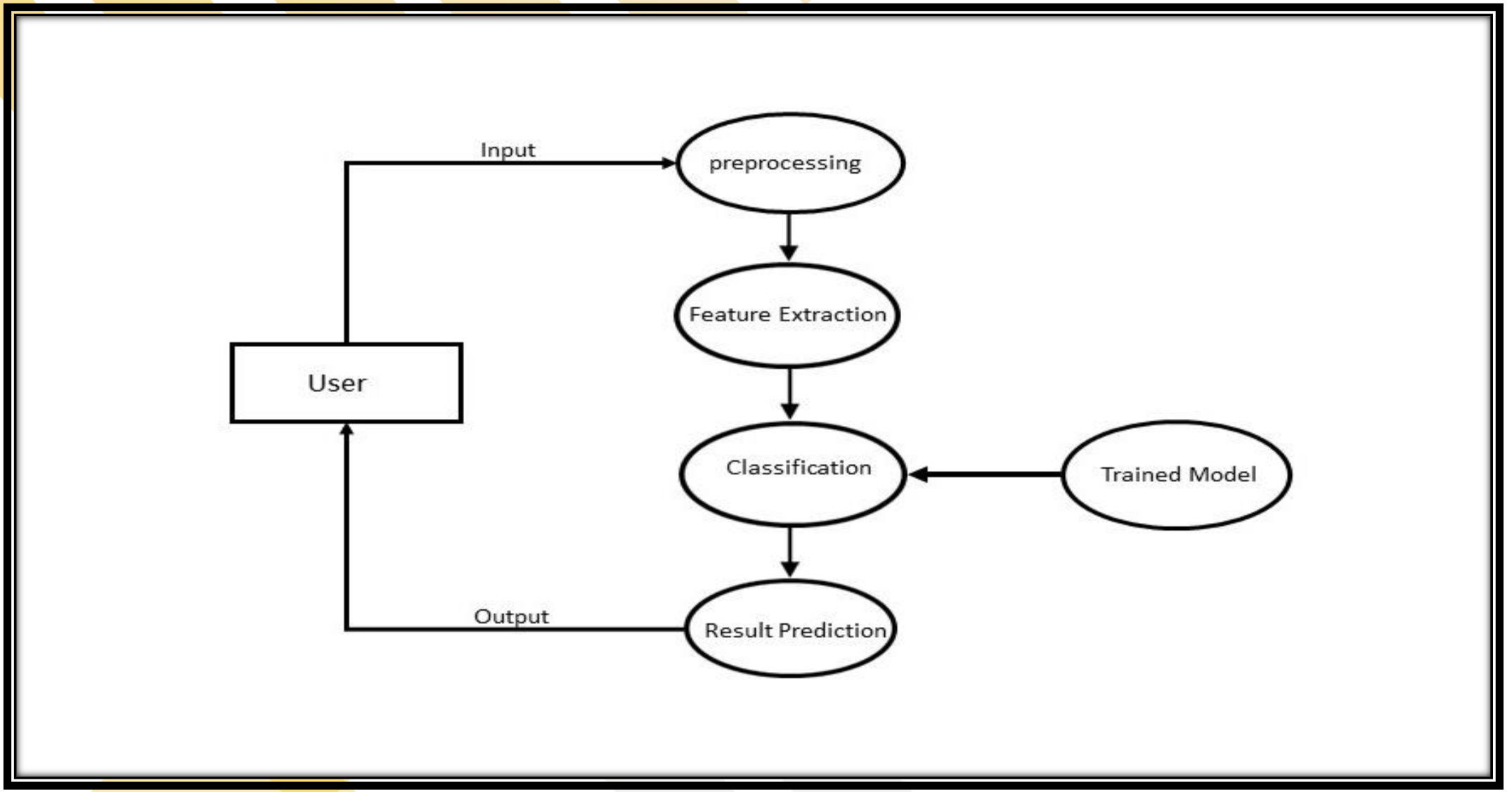
SEQUENCE DIAGRAM



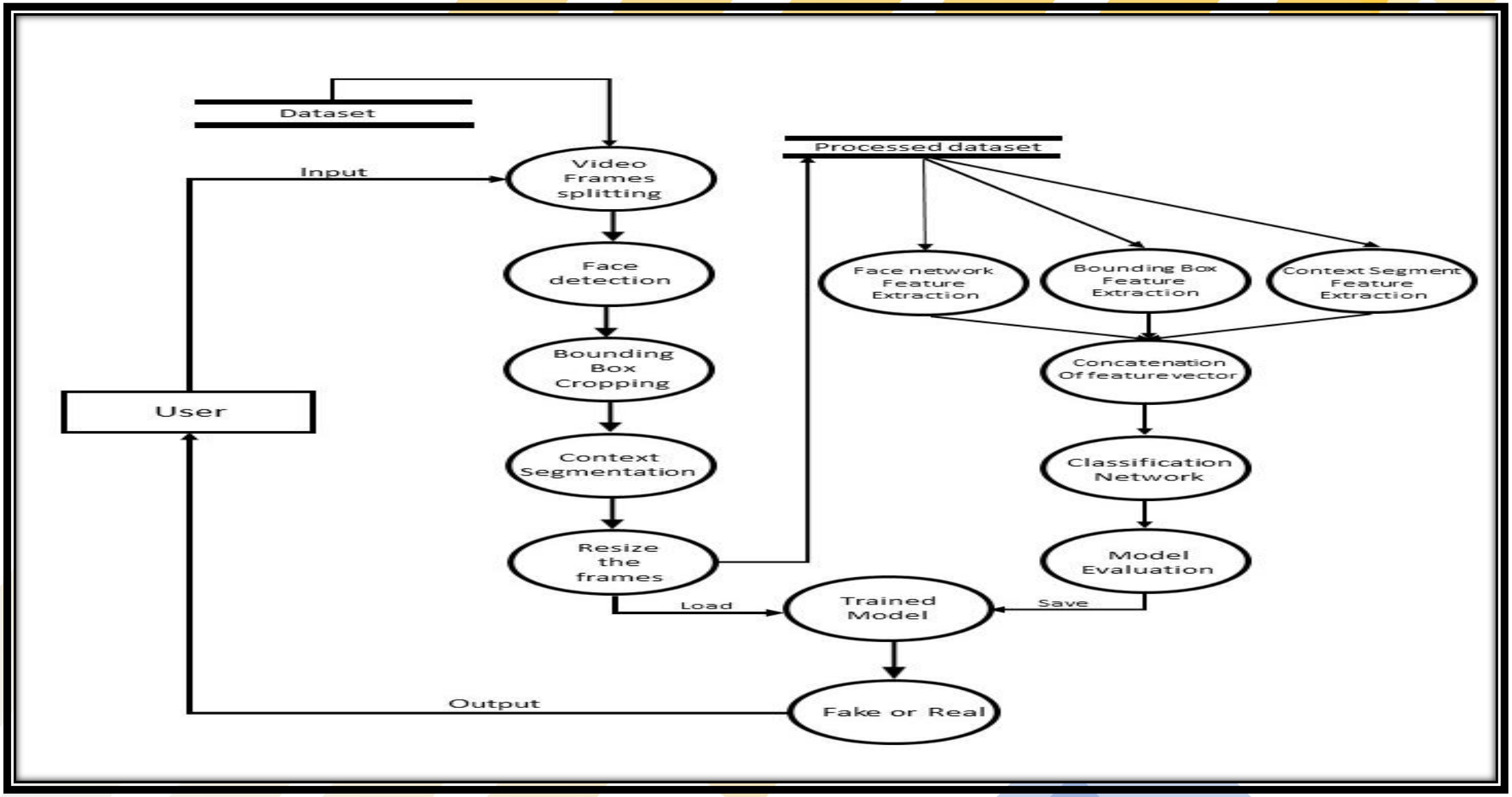
DFD LEVEL 0



DFD LEVEL 1


Input
preprocessing
Feature Extraction
User
Classification
Trained Model
Output
Result Prediction

DFD LEVEL 2


Dataset
Input
Video Frames splitting
Face detection
Bounding Box Cropping
Context Segmentation
Resize the frames
User
Load
Trained Model
Output
Fake or Real
Processed dataset
Face network Feature Extraction
Bounding Box Feature Extraction
Context Segment Feature Extraction
Concatenation Of feature vector
Classification Network
Model Evaluation
Save

IMPLEMENTATION

1. DATASET LOADING

- FaceForensics++ dataset which is publicly available is used.

- It contains videos and images manipulated using deepfakes, face2face and faceswap methods .

2. PREPROCESSING OF DATASET

- Extraction of video frames
- Perform face detection
- Cropping of bounding boxes
- Perform Face Segmentation
- Perform Context Segmentation

2.1 EXTRACTION OF FRAMES

- Video File Access:

  -Video files are accessed using the OpenCV library's VideoCapture object.

  -20 frames are read from each file.

- Iterative Extraction:

  frames are extracted iteratively from each video file.

- Frame Storage:

  Extracted frames are stored as individual image files in predefined directories.

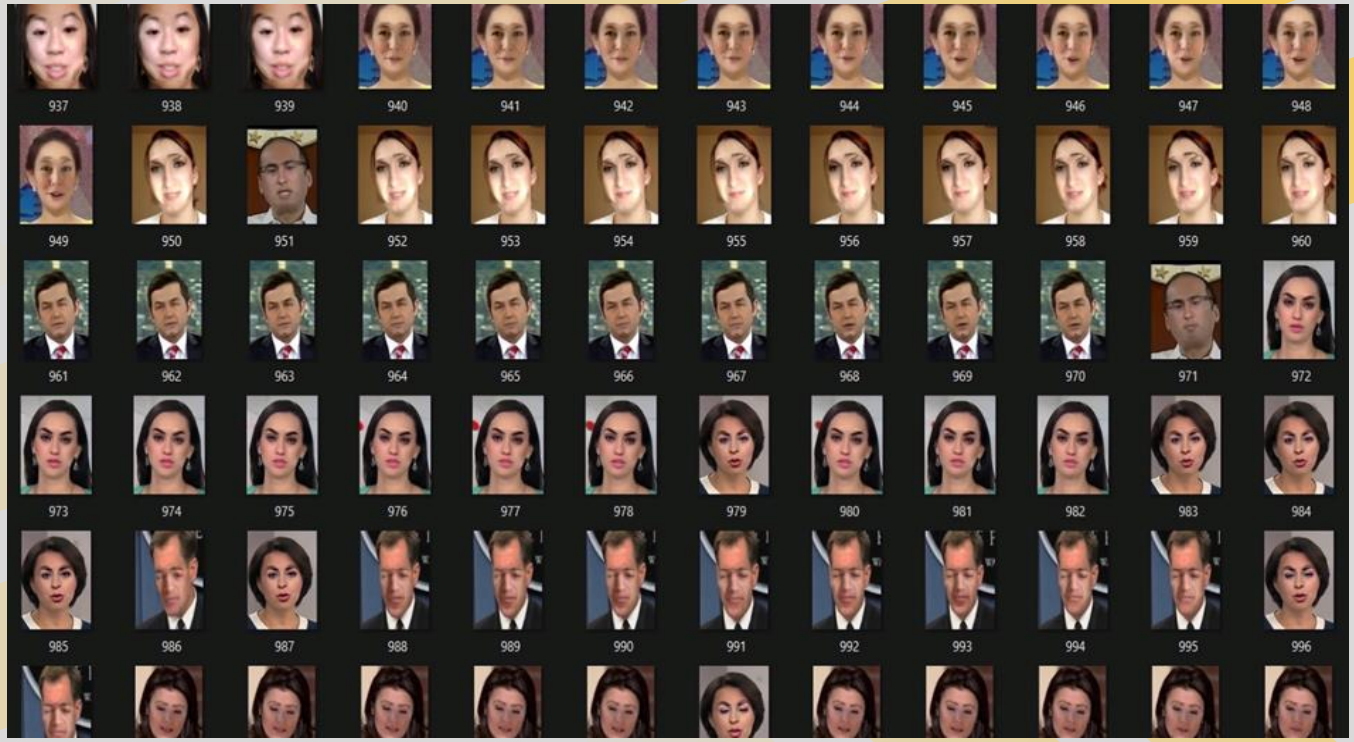
2.2 BOUNDING BOX FACE DETECTION

- For each frame in the 'fake' and 'real' directories ,attempt for face detection using **Haar cascades** is done.
- If a face is detected in the image, a **rectangle is drawn** around the detected face
- The **ROI** containing the detected face is **cropped** .
- The cropped face image is then **saved** in a new directory

2.2 BOUNDING BOX FACE DETECTION

- 2000+ frames obtained in both real and fake directories

SAMPLE FRAMES EXTRACTED



PREPROCESSING OF DATASET

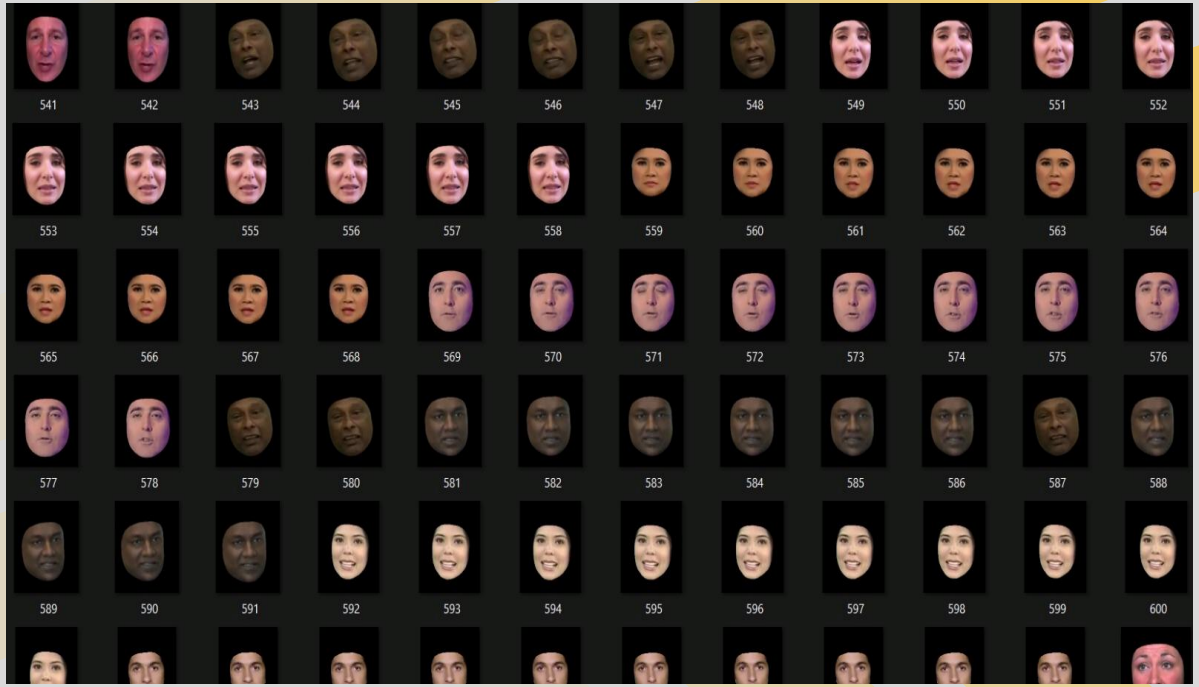
2.3 FACE SEGMENTATION

- Use MediaPipe's **FaceMesh** model for **detecting facial landmarks** in each image.
- Process each image to detect facial landmarks.
- Extract the facial landmarks **coordinates**.
- **Creating Face Mask**:
  - Define the **face oval** using predefined landmarks from MediaPipe.
  - Draw **lines** between the landmark points to form the face mask.
- **Applying Mask**:
  - Apply the face mask to the original image to **retain** only the **facial region**.
  - **Save** the processed image with the face mask applied.

2.3 FACE SEGMENTATION

- 2000+ frames in both fake and real directories

SAMPLE FRAMES EXTRACTED



2.4 CONTEXT SEGMENTATION

❑ **Face Detection and Landmark Extraction**

- Uses MediaPipe's Face Mesh model to detect facial landmarks in the image.

❑ **Creating the Mask**

- Once the facial landmarks are detected, a binary mask is created. The facial landmarks are used to define a polygonal region covering the face.

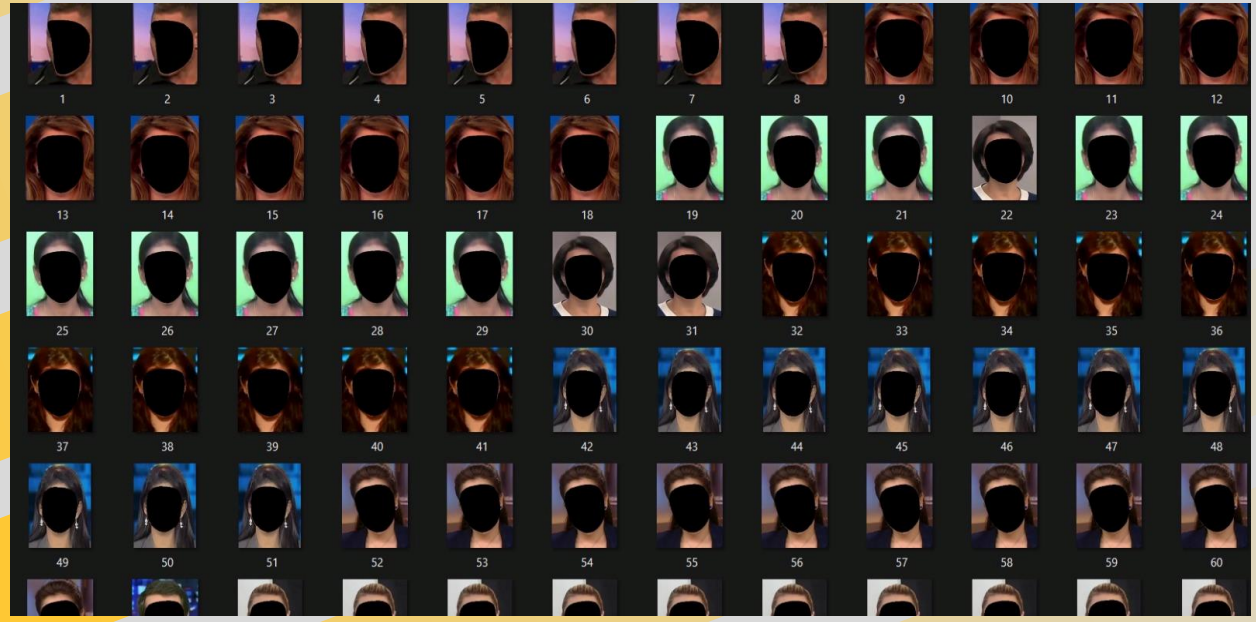
❑ **Applying Mask**

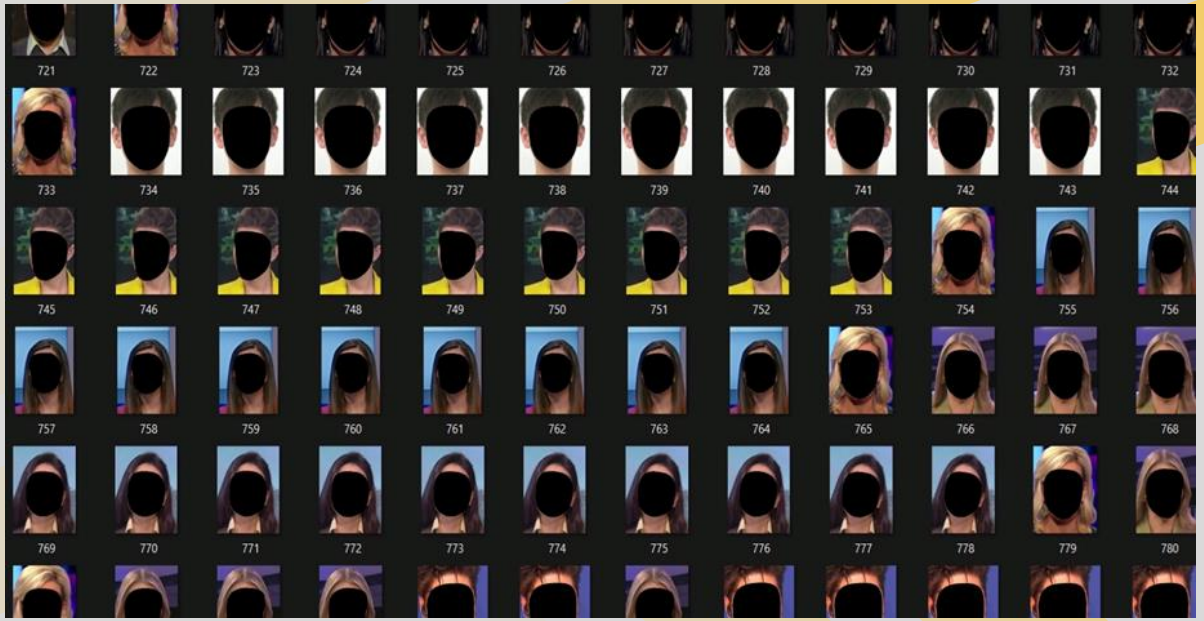
- The resulting binary mask will have the facial region masked and the surrounding context is extracted.
- **Save** the processed image with the face mask applied.

2.4 CONTEXT SEGMENTATION

- 2000+ frames in both fake and real folders combined.

SAMPLE FRAMES EXTRACTED





3. DEEPFAKE MODEL ARCHITECTURE CREATION

- Three customized networks are created for feature extraction –
  - **Bounding Box Network**
  - **Face Network**
  - **Context Recognition Network**
- The resulted extracted features are concatenated and fed to
  - **Classification Network**.

3.1 ARCHITECTURE FOR THE 3 FEATURE EXTRACTION NETWORKS

❖ **Input Layer**

- The input layer implicitly defines the input shape of images as (128, 128, 3), indicating images of height 128, width 128, and 3 color channels (RGB).

❖ **Hidden Layers**

- Multiple convolutional layers are stacked to extract hierarchical features from the input images.
- These convolutional layers are followed by activation functions (ReLU), batch normalization, max-pooling, and dropout layers.

❖ **Output Layer**

- Single output node representing the probability of an input image.
- A sigmoid activation function is applied to the output node to squash the output values between 0 and 1.

3.2 MULTI-MODAL CLASSIFICATION MODEL ARCHITECTURE

❖ **Input Layer**:

- In the sequential model used, the Flatten layer is the first layer in the model.
- It flattens the input data into a one-dimensional array.

❖ **Hidden Layer** :

- A single Dense layer
- It receives the flattened input data and applies a set of weights to transform the input.
- The layer also applies the sigmoid activation function to introduce non-linearity into the model.

❖ **Output Layer**:

- The output layer of the model is also a Dense layer with a single neuron.
- It utilizes the sigmoid activation function to squash the output into the range [0,1] .
- The output layer essentially makes the final decision regarding the classification of the input data.

4. TRAINING PROCESS

❑ GENERAL STEPS

- DATA LOADING AND RESIZING
- ASSIGN LABELS [0 – REAL AND 1- FAKE]
- MODEL COMPILATION – ADAM OPTIMIZER IS USED
- TRAINING PROCESS
- SAVE THE BEST MODEL

❑ TRAINING PROCESS

- Training data and labels are provided as input.
- The number of training epochs is set to 60.
- During training, the model adjusts its internal parameters (weights and biases) to minimize the loss function and improve its ability to correctly classify images.
- At the end of each epoch, the model's performance on the training data is evaluated.

❑ MODEL CHECKPOINTING

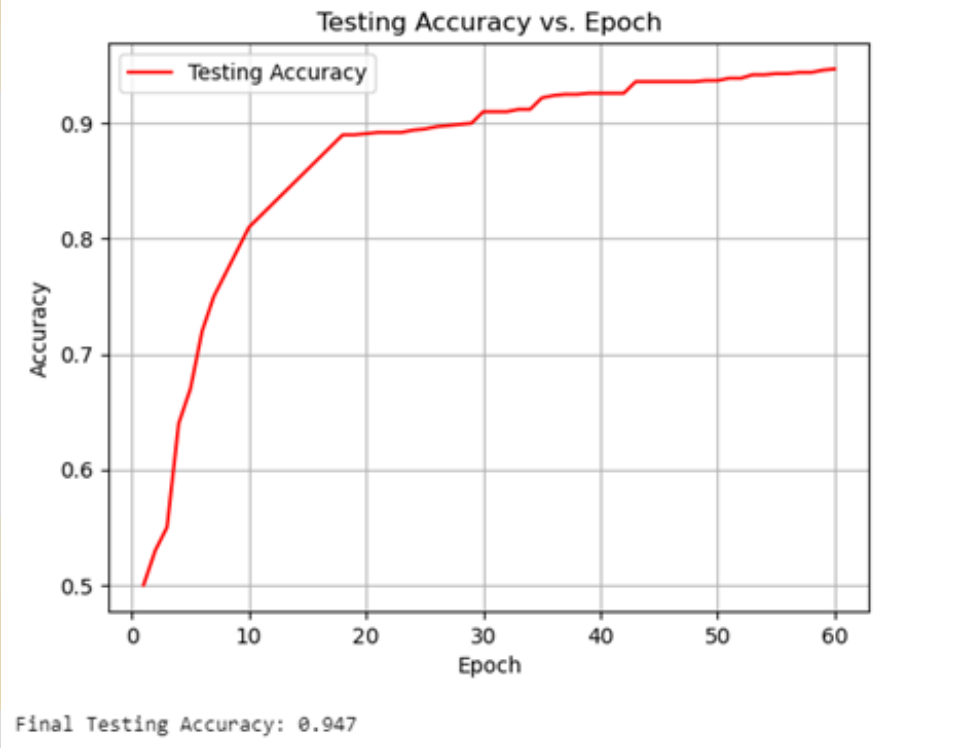
- A ModelCheckpoint callback is defined to save the model with the best accuracy seen during training.
- The callback monitors the 'accuracy' metric and saves the model weights to a file whenever there's an improvement in accuracy compared to the previous best model.

❑ SAVING THE MODELS

- After training completes, the trained model is saved.

TRAINING AND TESTING ACCURACY

- FINAL TRAINING ACCURACY SCORE : 0.978
- FINAL TESTING ACCURACY SCORE : 0.947



TRAINING PROCESS

5. PREDICTION PROCESS

- Input video or image
- Load Trained model
- Read frames from the input video(Only for video)
- Perform face detection and Bounding box cropping
- Perform Face and Context Segmentation
- Feature Extraction
- Feature vectors are concatenated into a single vector
- Classify as REAL or FAKE

6. FRONTEND GUI DESIGN AND MODEL INTEGRATION

❑ PYTHON TKINTER IS USED FOR GUI DESIGN

- The GUI allows users to either select an image or a video file.
- Once a file is selected, the GUI utilizes a combination of deep learning models to predict whether the content is real or fake.

❑ USER INTERFACE COMPONENTS

- The GUI consists of various graphical components such as buttons, frames, labels, and file dialogs.
- These components are arranged to create an intuitive and user-friendly interface for interacting with the application.

FRONTEND GUI DESIGN AND MODEL INTEGRATION

❑ MODEL INTEGRATION

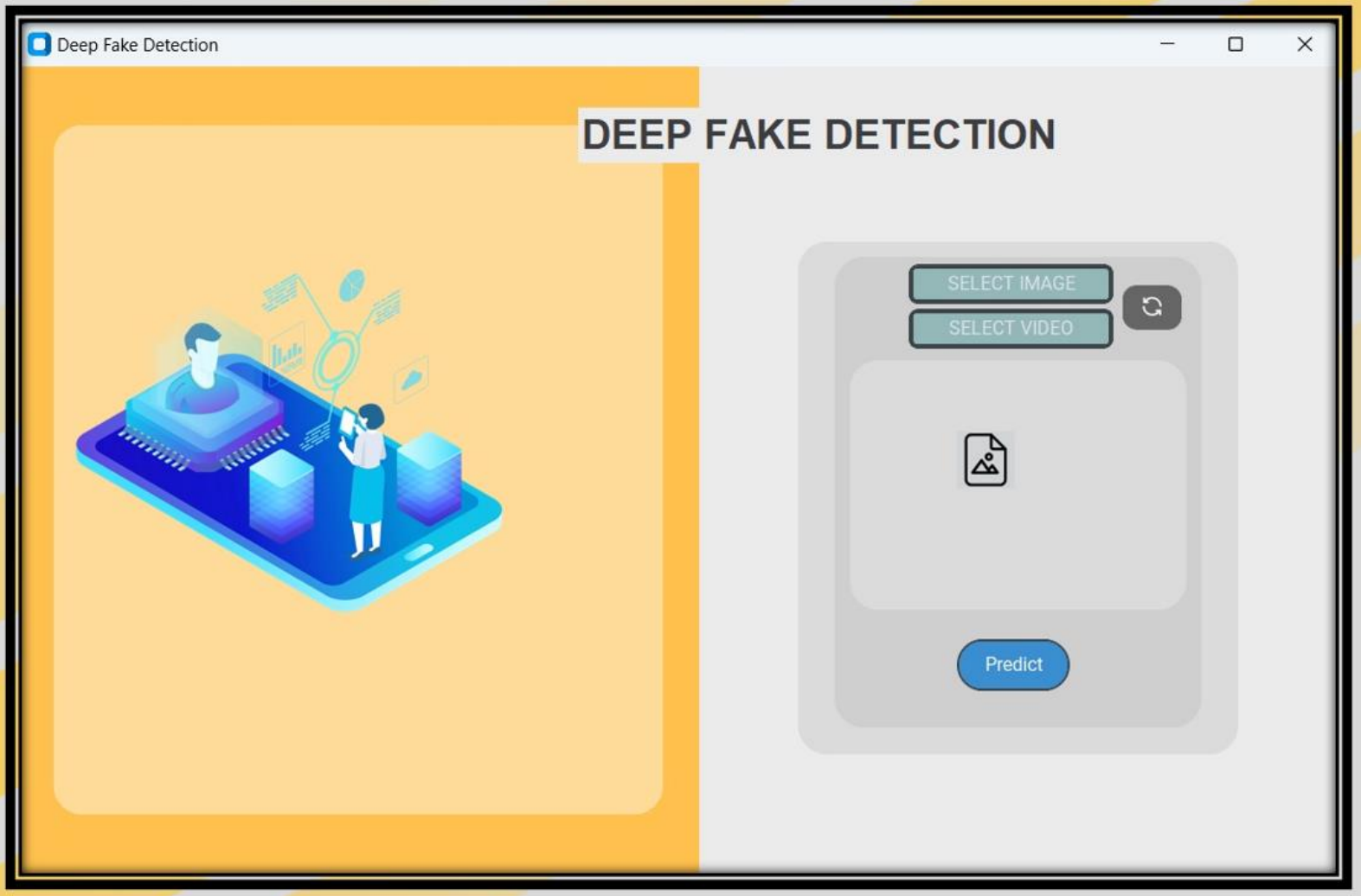
- The GUI integrates multiple deep learning models for different stages of the detection process, including bounding box detection, facial analysis, and contextual analysis.
- The deepfake detection models (model1, model2, model3, model4) are loaded using Keras (load_model()).

- **FEEDBACK MECHANISM**
- The GUI provides feedback to the user in the form of prediction results displayed in message boxes.
- After selecting an image or video, the user can click the "Predict" button to initiate the detection process. The predicted result (real or fake) is then presented to the user through a message box.

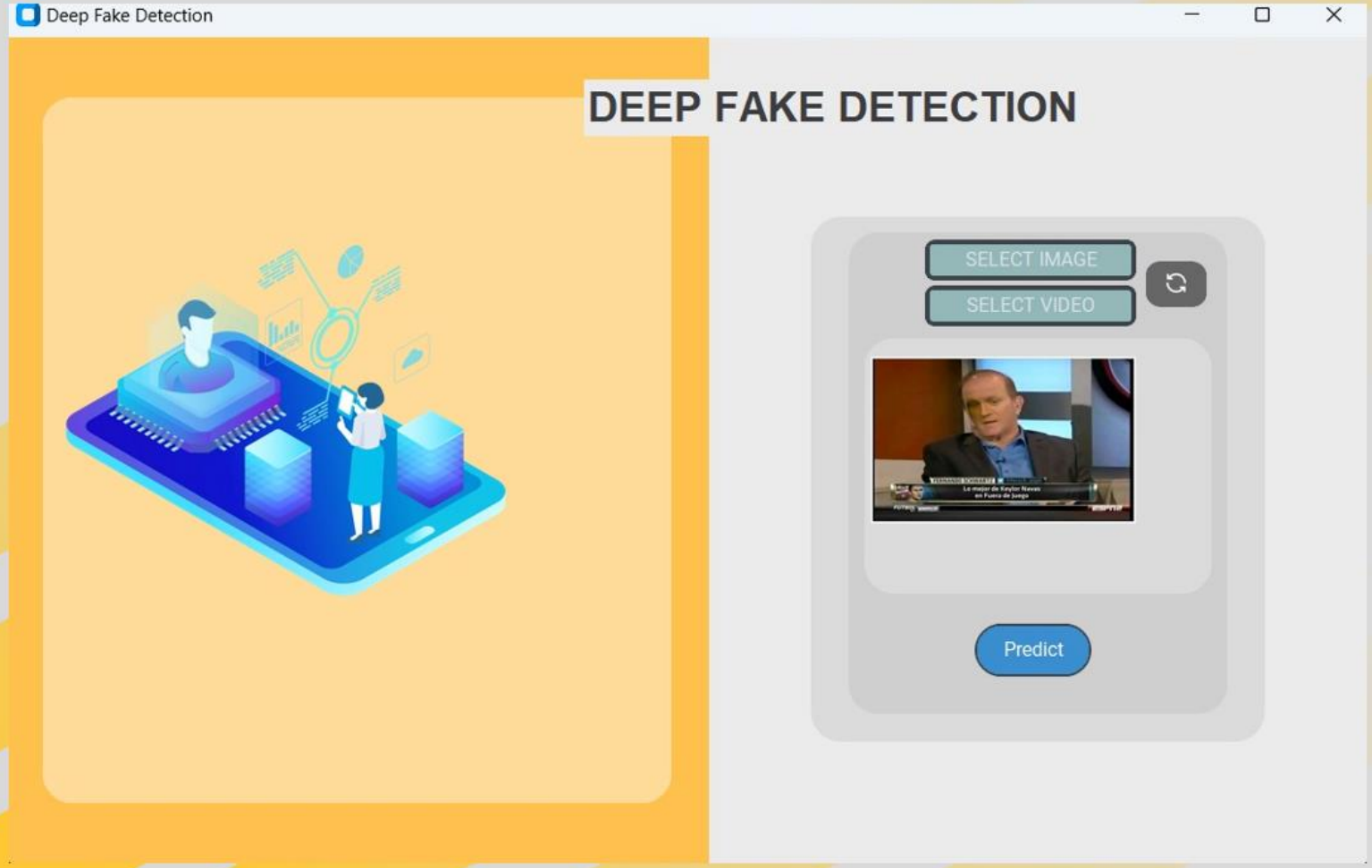
RESULTS

RESULTS

MAIN WINDOW



RESULTS



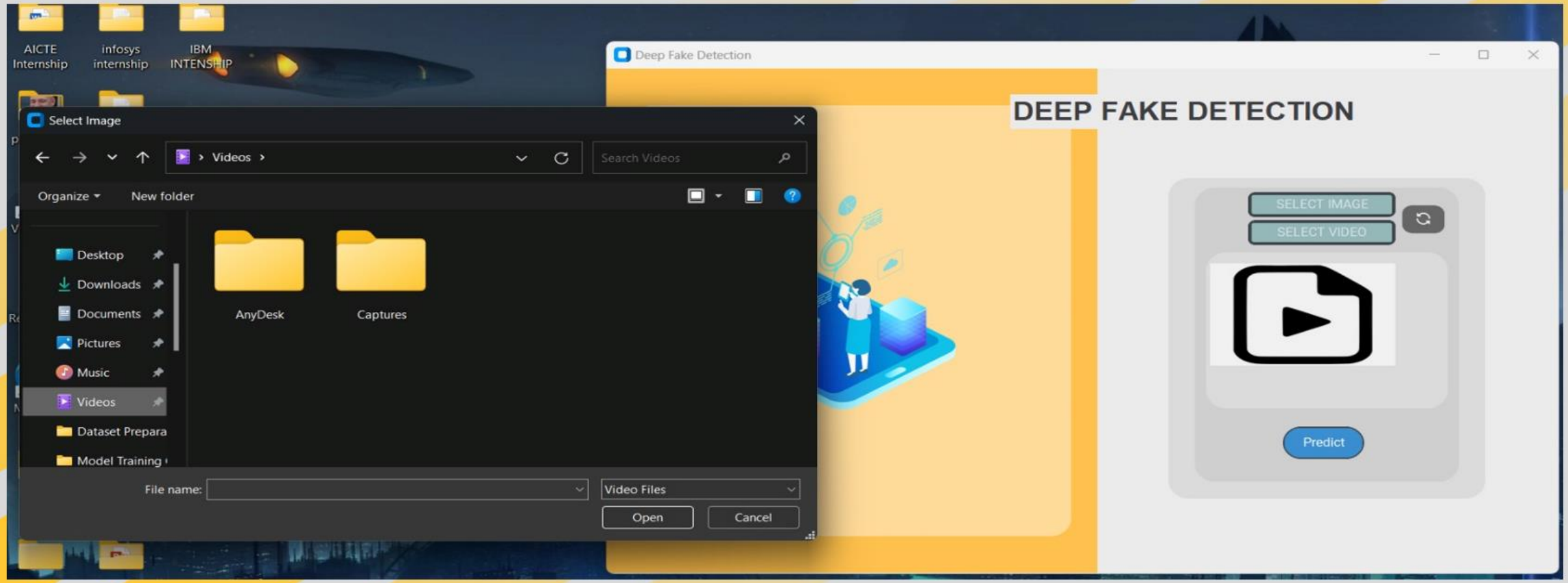
WHEN AN IMAGE IS PROVIDED FOR PREDICTION

RESULTS

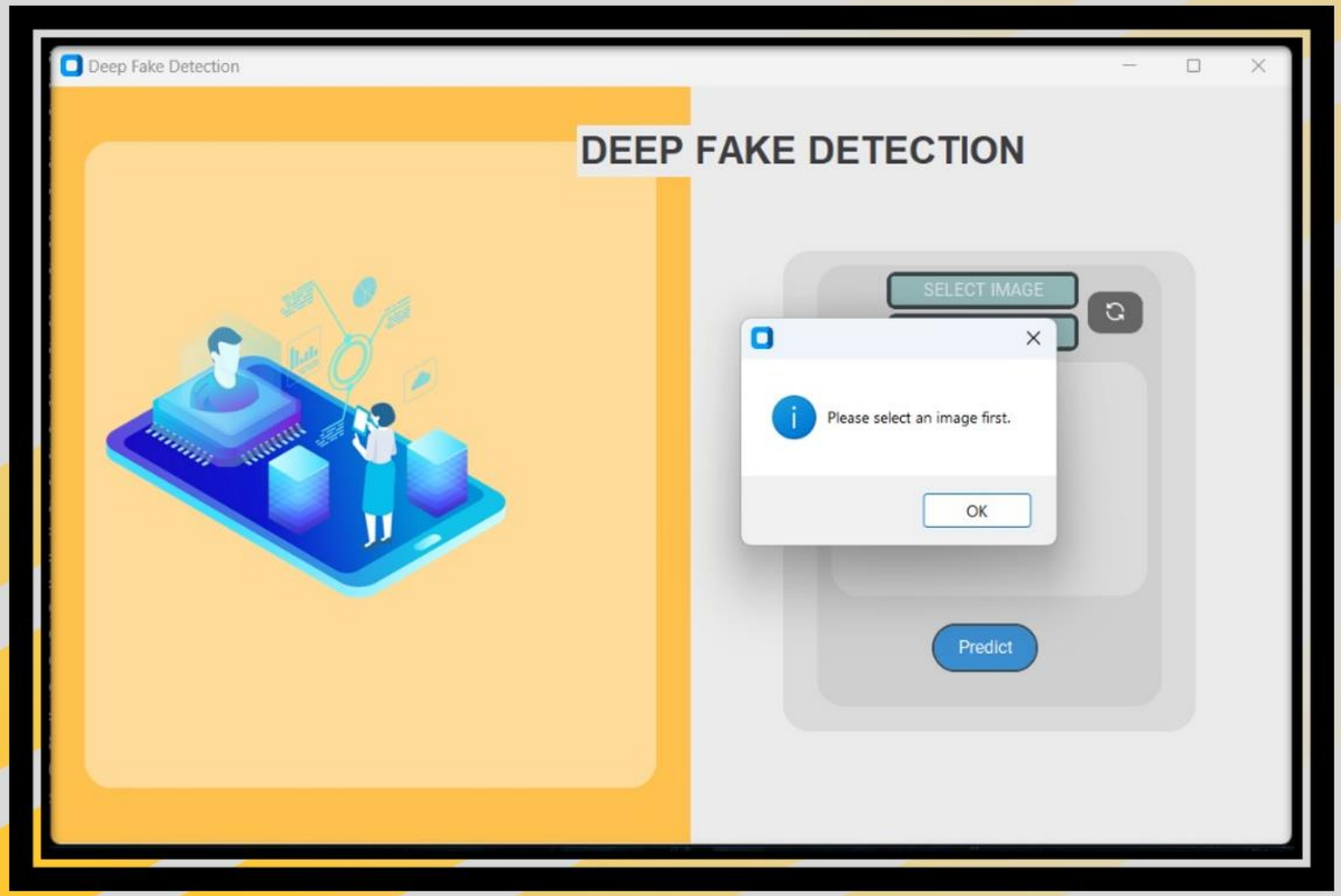
- POP UP WINDOW WHEN SELECT IMAGE BUTTON IS SELECTED

RESULTS

- POP UP WINDOW WHEN SELECT VIDEO BUTTON IS SELECTED

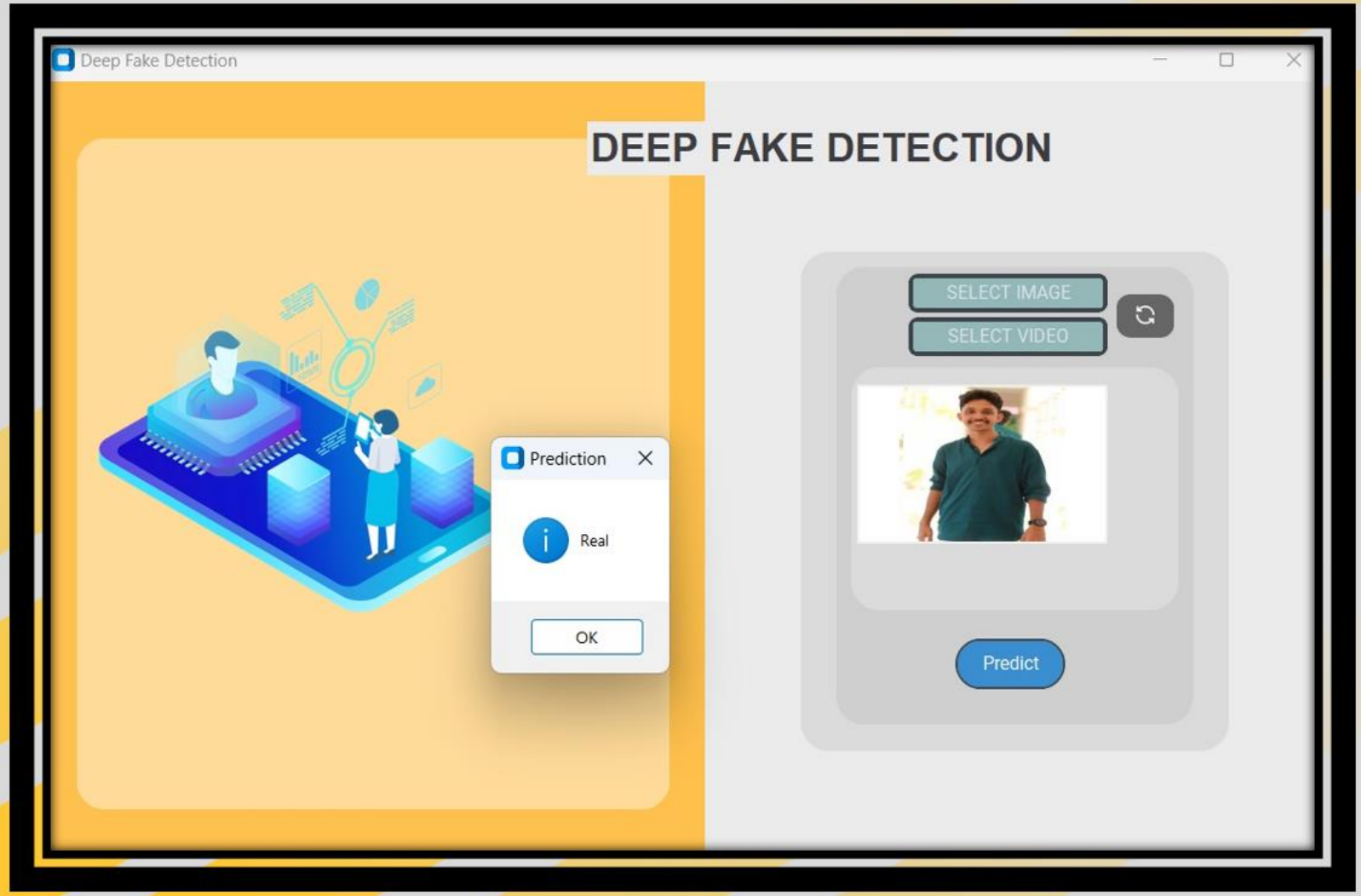


RESULTS



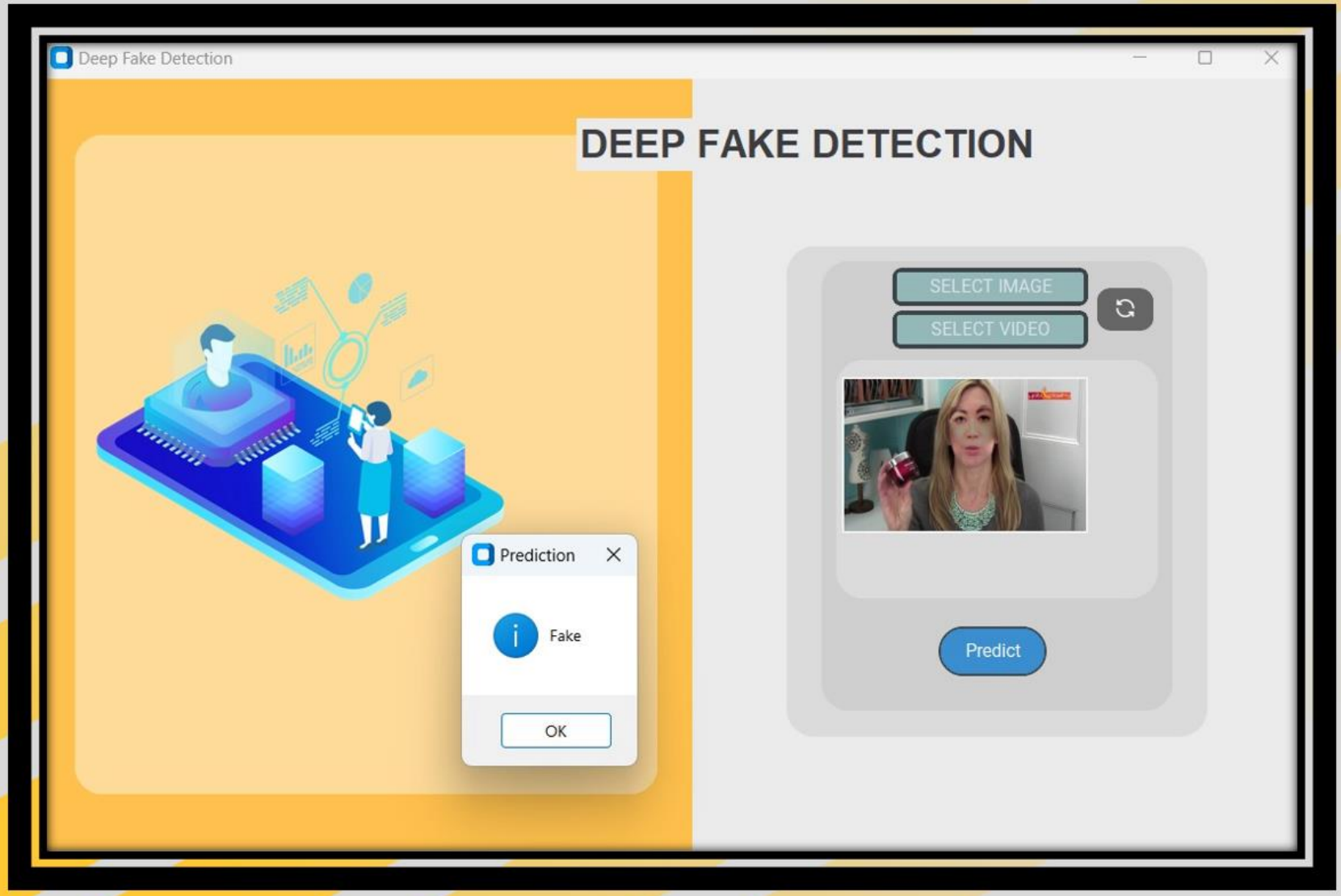
IF Predict BUTTON IS PRESSED WITHOUT INPUT

RESULTS



POP UP WINDOW FOR REAL PREDICTION

RESULTS



POP UP WINDOW FOR FAKE PREDICTION

CONCLUSION

- The development of deepfake detection technologies represents a crucial step in addressing the growing threat of manipulated multimedia content.

- It is essential to acknowledge that the cat-and-mouse game between deepfake creators and detection algorithms is ongoing, and continual research and development in this field are necessary.

- Deepfake detection is a valuable tool in the fight against misinformation, it should be part of a broader strategy that includes media literacy, responsible content creation, and legal frameworks to address this evolving threat effectively.

FUTURE SCOPE

- **Multimodal Detection**:
- Combining analysis of both visual and audio elements will become more prevalent.
- Detecting inconsistencies between lip movements and speech, for example, can enhance deepfake detection.
- **Real-Time Detection**:
- The development of real-time deepfake detection tools will be crucial, especially for live video conferencing, social media platforms, and other applications where rapid detection is essential.

REFERENCES

1. Detecting GAN generated Fake Images using Co-occurrence Matrices Lakshmanan Nataraj; Mayachitra Inc., Santa Barbara, California, USAhttps://doi.org/10.2352/ISSN.2470-1173.2019.5.MWSF-532 © 2019, Society for Imaging Science and Technology
2. Güera D.Delp, E.J. Deepfake video detection using recurrent neural networks. In Proceedings of the 2020 15th IEEE International Conference on Advanced Video and Signal Based Surveillance (AVSS), Auckland, New Zealand, 27–30 November 2020
3. Deepfake Detection through Deep Learning Deng Pan, Lixian Sun, Rui Wang, Xingjian Zhang, Richard O. Sinnott 2020 IEEE/ACM International Conference on Big Data Computing, Applications and Technologies (BDCAT)
4. Li, Y.; Lyu, S. Exposing deepfake videos by detecting face warping artifacts. arXiv 2020, arXiv:1811.00656.
5. Ismail, A.; Elpeltagy, M.; S. Zaki, M.; Eldahshan, K. A New Deep Learning-Based Methodology for Video Deepfake Detection Using XGBoost. Sensors 2021, 21, 5413. https://doi.org/10.3390/s21165413
6. Li, Y.; Chang, M.C.; Lyu, S. In ictu oculi: Exposing ai created fake videos by detecting eye blinking. In Proceedings of the 2019 IEEE International Workshop on Information Forensics and Security (WIFS), Hong Kong, China, 11–13 December 2019
7. FaceForensics++: Learning to Detect Manipulated Facial Images Andreas Rossler Davide Cozzolino Luisa Verdoliva Christian Riess Justus Thies Matthias Nießner 2019 IEEE/CVF International Conference on Computer Vision (ICCV)
8. Methods of Deepfake Detection Based on Machine Learning Artem A. Maksutov1, Viacheslav O. Morozov, Aleksander A. Lavrenov, Alexander S. Smirnov 2020 IEEE Conference of Russian Young Researchers in Electrical and Electronic Engineering (EIConRus) 10.1109/EIConRus49466.2020.9039057

THANK YOU