Changes

def forgeDetection():

text.delete('1.0', END)

global dnn\_model, labels

images = []

filename = askopenfilename(initialdir="testVideos")

pathlabel.config(text=filename)

actual\_labels = []

predicted\_labels = []

cap = cv2.VideoCapture(filename)

while True:

ret, frame = cap.read()

if ret:

img = cv2.resize(frame, (32, 32))

im2arr = np.array(img)

im2arr = im2arr.reshape(1, 32, 32, 3)

temp = [im2arr]

img = np.asarray(temp)

img = img.astype('float32') / 255 # Normalize

preds = dnn\_model.predict(img)

predict = np.argmax(preds)

recognize = labels[predict]

# Assume first class (index 0) is "Forgery" and others are "Real"

actual\_labels.append(0 if "forge" in filename.lower() else 1)

predicted\_labels.append(predict)

frame = cv2.resize(frame, (500, 500))

if predict == 0:

cv2.putText(frame, 'Forge Frame', (10, 50), cv2.FONT\_HERSHEY\_SIMPLEX, 0.9, (255, 0, 0), 3)

else:

cv2.putText(frame, 'Real Frame', (10, 50), cv2.FONT\_HERSHEY\_SIMPLEX, 0.9, (255, 0, 0), 3)

images.append(frame)

cv2.imshow('Forge & Real Frame Detection Output', frame)

if cv2.waitKey(1) & 0xFF == ord('q'):

break

else:

break

cap.release()

cv2.destroyAllWindows()

# Calculate metrics for the selected video

if len(actual\_labels) > 0:

precision = precision\_score(actual\_labels, predicted\_labels, average='macro') \* 100

recall = recall\_score(actual\_labels, predicted\_labels, average='macro') \* 100

f1 = f1\_score(actual\_labels, predicted\_labels, average='macro') \* 100

accuracy = accuracy\_score(actual\_labels, predicted\_labels) \* 100

text.insert(END, f"Metrics for {filename}:\n")

text.insert(END, f"Accuracy : {accuracy:.2f}%\n")

text.insert(END, f"Precision : {precision:.2f}%\n")

text.insert(END, f"Recall : {recall:.2f}%\n")

text.insert(END, f"F1 Score : {f1:.2f}%\n\n")

# Save the real frames video

fourcc = cv2.VideoWriter\_fourcc(\*"mp4v")

video = cv2.VideoWriter('video\_after\_removing\_forge.mp4', fourcc, 25, (500, 500))

for img in images:

video.write(img)

video.release()

cv2.destroyAllWindows()