FACE MASK DETECTION

P SAIKARTHIK

VU21CSEN0100398

1 Abstract

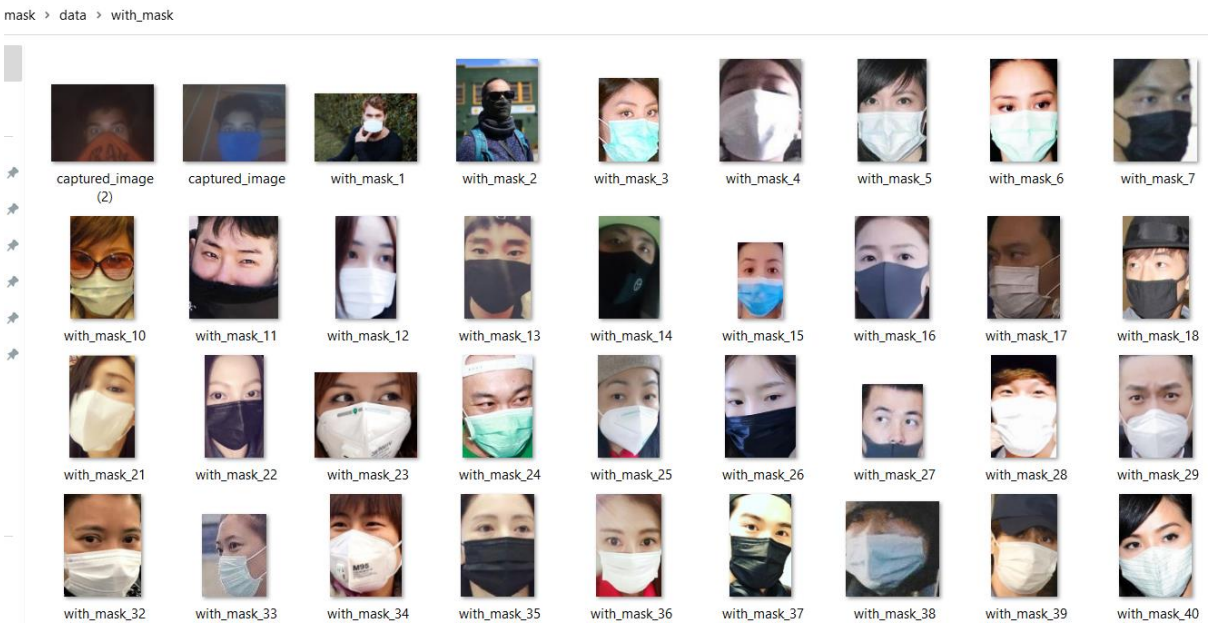
This project involves developing a face mask detection system using a machine learning model. The system is designed to classify whether a person is wearing a mask or not based on images. The primary components of this system include data preprocessing, model training, and real-time face mask detection using a webcam.

2. Dataset Preparation

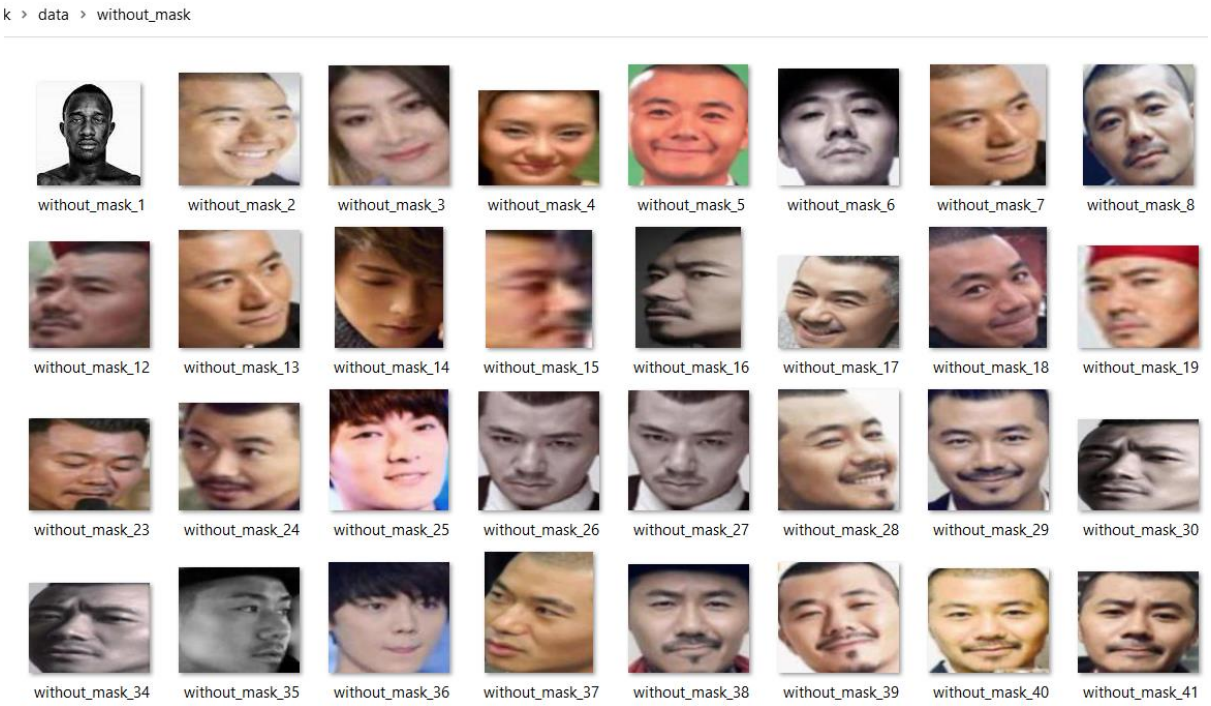
Data Collection: The dataset consists of images stored in two folders: with_mask and without_mask. These folders contain images labeled as either wearing a mask (with_mask) or not wearing a mask (without_mask).

Feature Extraction: Images are read in grayscale, resized to 60x60 pixels, flattened, and normalized. Each image is labeled accordingly (1 for mask, 0 for no mask).

WITH MASK



WITHOUT MASK



3. Data Processing

Data Splitting: The dataset is split into training and testing sets using an 80-20 split ratio.

Feature Scaling: StandardScaler is used to standardize the features, ensuring that each feature contributes equally to the model training process.

4. Model Training

Model Selection: A Histogram-Based Gradient Boosting Classifier is chosen for training due to its efficiency in handling structured data.

Training: The model is trained on the preprocessed training data and subsequently evaluated on the test data.

5. Real-Time Face Mask Detection

Webcam Integration: A webcam is used to capture real-time images. The captured frame is displayed using OpenCV, and the image is saved for further processing.

Face Mask Detection: The captured image can be processed through the trained model to predict whether the person is wearing a mask or not.

IMPLEMENTATION

[1]:
```
import pandas as pd
import numpy as np
import cv2
import numpy as np
import os
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import accuracy_score
import joblib
```

[2]:
```
folder_path = 'with_mask'
jpg_files = []
for file_name in os.listdir(folder_path):
    if file_name.endswith('.jpg'):
        jpg_files.append(file_name)
```

[ ]:
```
```

[3]:
```
l=[]
def feature_extractor(file,clas):
    img = cv2.imread(file,cv2.IMREAD_GRAYSCALE)
    img = cv2.resize(img,(60,60))
    img = img.flatten()
    img = img/255
    l.append([img,clas])
```

[4]:
```
for i in jpg_files:
    feature_extractor('with_mask/'+i,1)
```

[5]:
```
folder_path = 'without_mask'
jpg = []
for file_name in os.listdir(folder_path):
    if file_name.endswith('.jpg'):
        jpg.append(file_name)
```

[6]:
```
for i in jpg:
    feature_extractor('without_mask/'+i,0)
```

[7]:
```
df = pd.DataFrame(l,columns=['feature','label'])
```

[8]:
```
x = np.array(df['feature'].tolist())
y = df['label']
```

[9]:
```
x_train,x_test,y_train,y_test = train_test_split(x,y,test_size=0.2,random_state=42)
```

[10]:
```
scaler = StandardScaler()
```

[11]:
```
x_train = scaler.fit_transform(x_train)
x_test = scaler.transform(x_test)
```

[12]:
```
from sklearn.ensemble import HistGradientBoostingClassifier
```

[13]:
```
clr1 = HistGradientBoostingClassifier()
clr1.fit(x_train,y_train)
```

[13]:
```
HistGradientBoostingClassifier()
```

[14]:
```
y_pred1 = clr1.predict(x_test)
print(accuracy_score(y_test,y_pred1))
```

```
0.8821972203838517
```

[ ]:

[32]:
```
import cv2

cap = cv2.VideoCapture(0)
if not cap.isOpened():
    print("Error: Could not open camera.")
else:
    ret, frame = cap.read()
    if ret:
        cv2.imshow('Captured Image', frame)
        cv2.imwrite('captured_image.jpg', frame)
        cv2.waitKey(0)
        cv2.destroyAllWindows()
    else:
        print("Error: Could not read frame.")
    cap.release()
```

[33]:
```
def test(file):
    img = cv2.imread(file,cv2.IMREAD_GRAYSCALE)
    img = cv2.resize(img,(60,60))
    img = img.flatten()
    img = img/255
    return img
```

[34]:
```
l=test('with_mask\captured_image.jpg')
```

[35]:
```
l= scaler.transform([l])
y_pred1 = clr1.predict(l)
```

[37]:
```
if(y_pred1[0]==0):
    print("NO MASK DETECTED")
else:
    print("MASK DETECTED")
```

```
NO MASK DETECTED
```

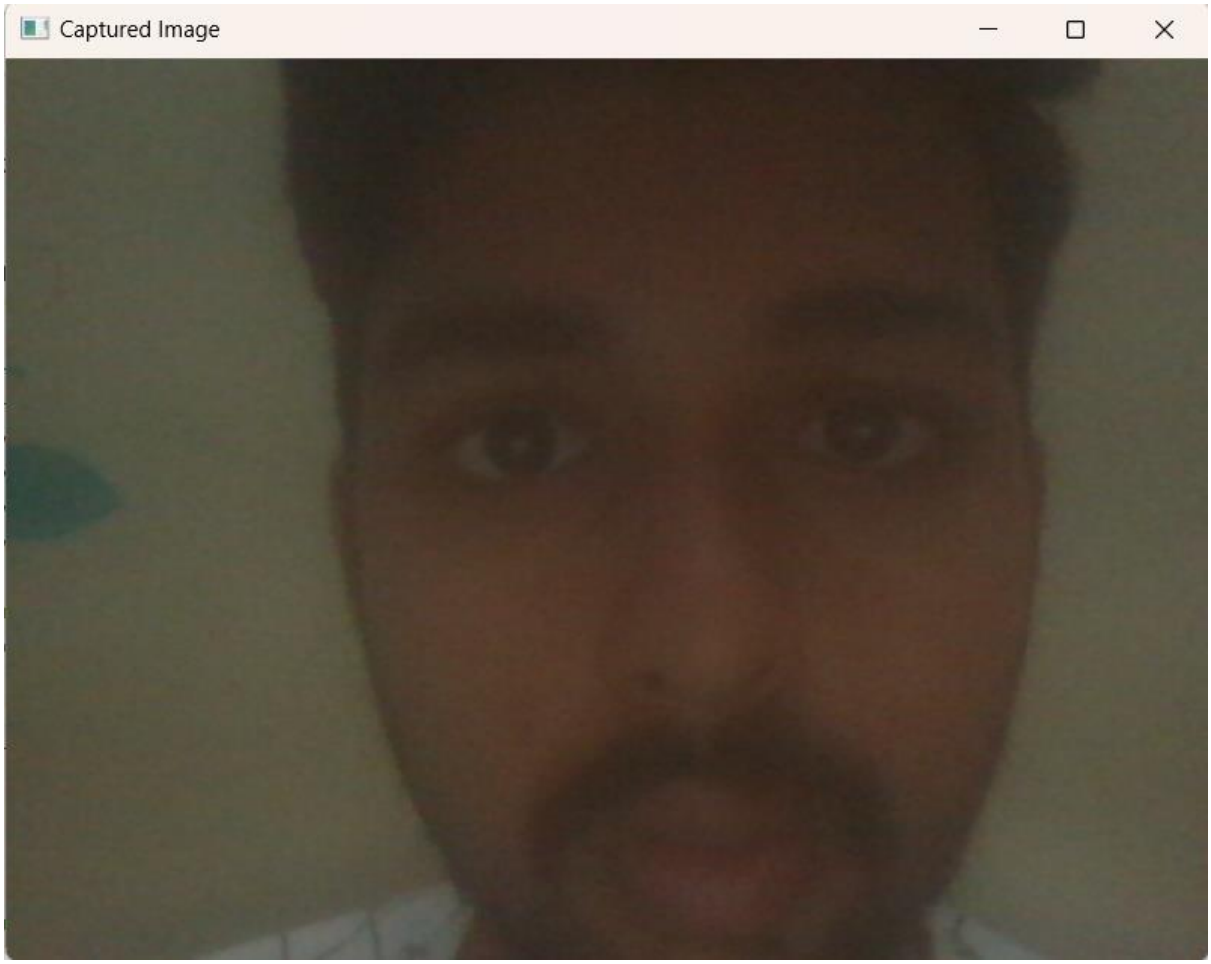
[26]:
```
joblib.dump(clr1, 'random_forest_model.pkl')
```

[26]:
```
['random_forest_model.pkl']
```

6. Results

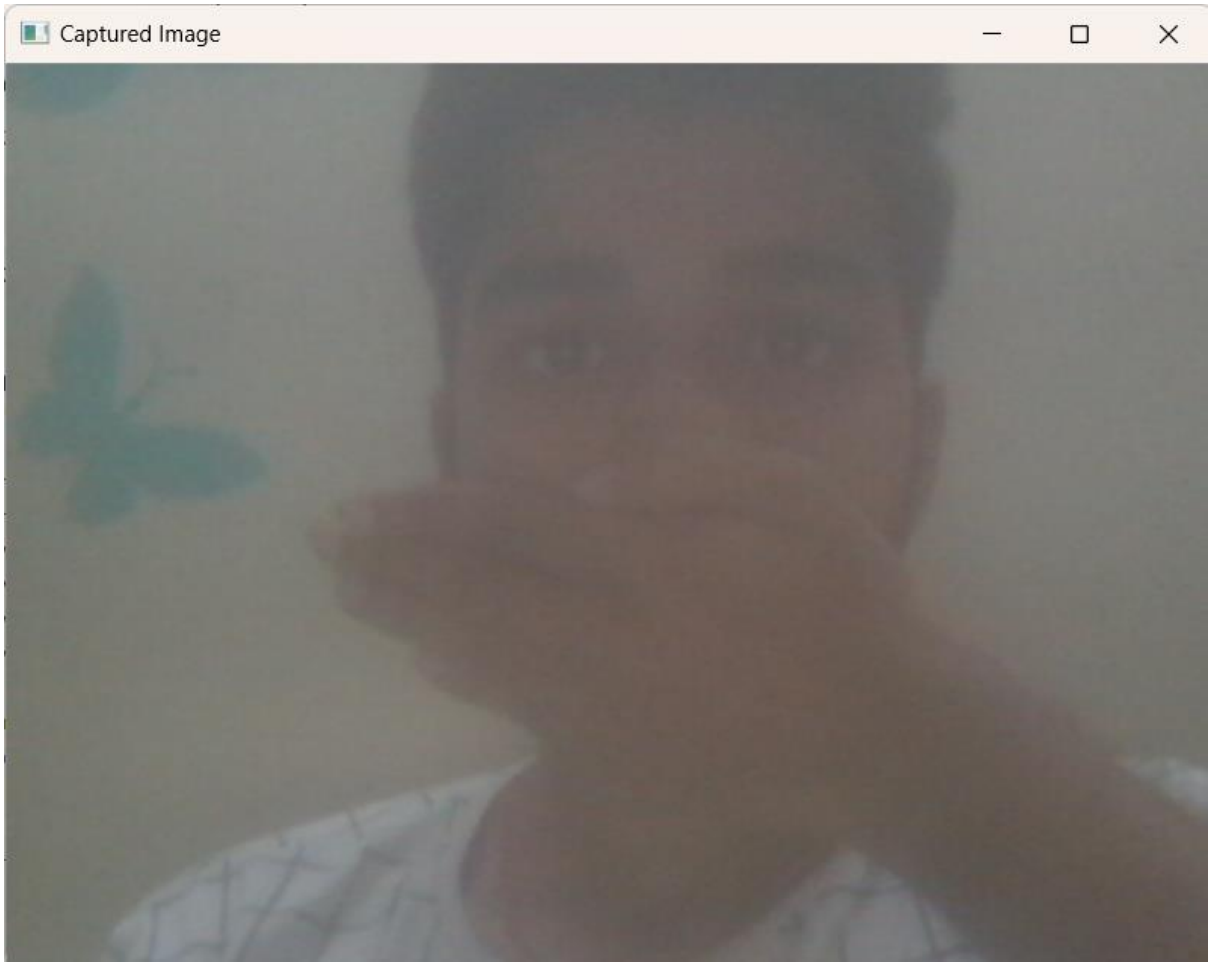
Model Evaluation: The accuracy of the model is evaluated on the test set, and further fine-tuning can be done to improve performance.

Real-Time Performance: The system successfully captures and processes real-time images, though integration with the prediction model is implied but not shown in the code.



]:
```
if(y_pred1[0]==0):
    print("NO MASK DETECTED")
else:
    print("MASK DETECTED")
```

```
NO MASK DETECTED
```



[42]:
```
if(y_pred1[0]==0):
    print("NO MASK DETECTED")
else:
    print("MASK DETECTED")
```

```
MASK DETECTED
```

7. Conclusion

This face mask detection system leverages machine learning and computer vision to detect the presence of face masks in real-time. The project demonstrates a practical application of machine learning for public health and safety, particularly in the context of the COVID-19 pandemic.

🌐 Sources

geeksforgeeks.org - FaceMask Detection using TensorFlow in Python

pyimagesearch.com - COVID-19: Face Mask Detector with OpenCV, Keras ...

data-flair.training - Real-Time Face Mask Detector with Python, OpenCV, Keras

github.com - bighneshpati/Face-Mask-Detection-using-CNN

youtube.com - Face Mask Detection using Python and Machine Learning ...

github.com - chandrikadeb7/Face-Mask-Detection