# Assignment 4.1 Mood Detection with OpenCV

April 13, 2025

| Technological Institute of the Philippines | Quezon City - Computer Engineering |
|---|---|
| Course Code: | CPE 313 |
| Course Title: | Advanced Machine Learning and Deep Learning |
| 2nd Semester | AY 2024-2025 |
| **ACTIVITY NO.** | Assignment 4.1 Mood Detection with OpenCV |
| **Name** | Base, Angelo P. |
| **Section** | CPE32S1 |
| **Date Performed**: | 06/04/2025 |
| **Date Submitted**: | 13/04/2025 |
| **Instructor**: | Dr. Alonica Villanueva |

[2]:
```
import os
import cv2
import numpy as np

import pickle
from tqdm import tqdm

from tensorflow.keras.preprocessing.image import ImageDataGenerator,␣
↪img_to_array
from tensorflow.keras.applications import VGG16
from tensorflow.keras.models import Sequential, load_model
from tensorflow.keras.layers import Flatten, Dense, Dropout
from tensorflow.keras.optimizers import Adam

from sklearn.svm import SVC
from sklearn.model_selection import train_test_split
from sklearn.metrics import accuracy_score, confusion_matrix

import itertools

from deepface import DeepFace
import matplotlib.pyplot as plt
# from google.colab.patches import cv2_imshow
```

### 0.0.1 Face Recognition

[ ]:
```
dataset_path = "custom_facial_expression_dataset/train"
my_face_folders = ["angry", "happy", "sad", "confused"]
others_folder = "custom_facial_expression_dataset/others"

embeddings = []
labels = []

# Process my_face_folders images
for folder in my_face_folders:
  folder_path = os.path.join(dataset_path, folder)
  for img_name in os.listdir(folder_path):
    img_path = os.path.join(folder_path, img_name)
    try:
      embedding = DeepFace.represent(img_path,␣
 ↪model_name="Facenet")[0]["embedding"]
      embeddings.append(embedding)
      labels.append(1)
    except:
      print(f"Skipping image: {img_path}")

# Process others images
for img_name in os.listdir(others_folder):
  img_path = os.path.join(others_folder, img_name)
  try:
    embedding = DeepFace.represent(img_path,␣
 ↪model_name="Facenet")[0]["embedding"]
    embeddings.append(embedding)
    labels.append(0)
  except:
    print(f"Skipping image: {img_path}")

embeddings = np.array(embeddings)
labels = np.array(labels)

with open("pkl/face_embeddings.pkl", "wb") as f:
  pickle.dump((embeddings, labels), f)
print("Saved!")
```

```
Skipping image: custom_facial_expression_dataset/train\angry\Training_10.jpg
Skipping image: custom_facial_expression_dataset/train\angry\Training_110.jpg
Skipping image: custom_facial_expression_dataset/train\angry\Training_116.jpg
Skipping image: custom_facial_expression_dataset/train\angry\Training_117.jpg
Skipping image: custom_facial_expression_dataset/train\angry\Training_119.jpg
Skipping image: custom_facial_expression_dataset/train\angry\Training_12.jpg
Skipping image: custom_facial_expression_dataset/train\angry\Training_120.jpg
Skipping image: custom_facial_expression_dataset/train\angry\Training_121.jpg
```

```
Skipping image: custom_facial_expression_dataset/train\angry\Training_137.jpg
Skipping image: custom_facial_expression_dataset/train\angry\Training_14.jpg
Skipping image: custom_facial_expression_dataset/train\angry\Training_142.jpg
Skipping image: custom_facial_expression_dataset/train\angry\Training_157.jpg
Skipping image: custom_facial_expression_dataset/train\angry\Training_168.jpg
Skipping image: custom_facial_expression_dataset/train\angry\Training_171.jpg
Skipping image: custom_facial_expression_dataset/train\angry\Training_187.jpg
Skipping image: custom_facial_expression_dataset/train\angry\Training_189.jpg
Skipping image: custom_facial_expression_dataset/train\angry\Training_194.jpg
Skipping image: custom_facial_expression_dataset/train\angry\Training_195.jpg
Skipping image: custom_facial_expression_dataset/train\angry\Training_197.jpg
Skipping image: custom_facial_expression_dataset/train\angry\Training_198.jpg
Skipping image: custom_facial_expression_dataset/train\angry\Training_200.jpg
Skipping image: custom_facial_expression_dataset/train\angry\Training_28.jpg
Skipping image: custom_facial_expression_dataset/train\angry\Training_3.jpg
Skipping image: custom_facial_expression_dataset/train\angry\Training_38.jpg
Skipping image: custom_facial_expression_dataset/train\angry\Training_4.jpg
Skipping image: custom_facial_expression_dataset/train\angry\Training_5.jpg
Skipping image: custom_facial_expression_dataset/train\angry\Training_53.jpg
Skipping image: custom_facial_expression_dataset/train\angry\Training_6.jpg
Skipping image: custom_facial_expression_dataset/train\angry\Training_62.jpg
Skipping image: custom_facial_expression_dataset/train\angry\Training_65.jpg
Skipping image: custom_facial_expression_dataset/train\angry\Training_7.jpg
Skipping image: custom_facial_expression_dataset/train\angry\Training_73.jpg
Skipping image: custom_facial_expression_dataset/train\angry\Training_8.jpg
Skipping image: custom_facial_expression_dataset/train\angry\Training_82.jpg
Skipping image: custom_facial_expression_dataset/train\angry\Training_89.jpg
Skipping image: custom_facial_expression_dataset/train\happy\Training_100.jpg
Skipping image: custom_facial_expression_dataset/train\happy\Training_101.jpg
Skipping image: custom_facial_expression_dataset/train\happy\Training_102.jpg
Skipping image: custom_facial_expression_dataset/train\happy\Training_103.jpg
Skipping image: custom_facial_expression_dataset/train\happy\Training_114.jpg
Skipping image: custom_facial_expression_dataset/train\happy\Training_125.jpg
Skipping image: custom_facial_expression_dataset/train\happy\Training_129.jpg
Skipping image: custom_facial_expression_dataset/train\happy\Training_147.jpg
Skipping image: custom_facial_expression_dataset/train\happy\Training_149.jpg
Skipping image: custom_facial_expression_dataset/train\happy\Training_150.jpg
Skipping image: custom_facial_expression_dataset/train\happy\Training_151.jpg
Skipping image: custom_facial_expression_dataset/train\happy\Training_152.jpg
Skipping image: custom_facial_expression_dataset/train\happy\Training_153.jpg
Skipping image: custom_facial_expression_dataset/train\happy\Training_156.jpg
Skipping image: custom_facial_expression_dataset/train\happy\Training_157.jpg
Skipping image: custom_facial_expression_dataset/train\happy\Training_160.jpg
Skipping image: custom_facial_expression_dataset/train\happy\Training_162.jpg
Skipping image: custom_facial_expression_dataset/train\happy\Training_163.jpg
Skipping image: custom_facial_expression_dataset/train\happy\Training_164.jpg
Skipping image: custom_facial_expression_dataset/train\happy\Training_165.jpg
Skipping image: custom_facial_expression_dataset/train\happy\Training_169.jpg
```

```
Skipping image: custom_facial_expression_dataset/train\happy\Training_17.jpg
Skipping image: custom_facial_expression_dataset/train\happy\Training_170.jpg
Skipping image: custom_facial_expression_dataset/train\happy\Training_171.jpg
Skipping image: custom_facial_expression_dataset/train\happy\Training_179.jpg
Skipping image: custom_facial_expression_dataset/train\happy\Training_198.jpg
Skipping image: custom_facial_expression_dataset/train\happy\Training_2.jpg
Skipping image: custom_facial_expression_dataset/train\happy\Training_23.jpg
Skipping image: custom_facial_expression_dataset/train\happy\Training_44.jpg
Skipping image: custom_facial_expression_dataset/train\happy\Training_62.jpg
Skipping image: custom_facial_expression_dataset/train\happy\Training_68.jpg
Skipping image: custom_facial_expression_dataset/train\happy\Training_78.jpg
Skipping image: custom_facial_expression_dataset/train\happy\Training_79.jpg
Skipping image: custom_facial_expression_dataset/train\happy\Training_80.jpg
Skipping image: custom_facial_expression_dataset/train\happy\Training_81.jpg
Skipping image: custom_facial_expression_dataset/train\happy\Training_82.jpg
Skipping image: custom_facial_expression_dataset/train\happy\Training_83.jpg
Skipping image: custom_facial_expression_dataset/train\happy\Training_86.jpg
Skipping image: custom_facial_expression_dataset/train\happy\Training_87.jpg
Skipping image: custom_facial_expression_dataset/train\sad\Training_1.jpg
Skipping image: custom_facial_expression_dataset/train\sad\Training_101.jpg
Skipping image: custom_facial_expression_dataset/train\sad\Training_113.jpg
Skipping image: custom_facial_expression_dataset/train\sad\Training_116.jpg
Skipping image: custom_facial_expression_dataset/train\sad\Training_12.jpg
Skipping image: custom_facial_expression_dataset/train\sad\Training_138.jpg
Skipping image: custom_facial_expression_dataset/train\sad\Training_14.jpg
Skipping image: custom_facial_expression_dataset/train\sad\Training_150.jpg
Skipping image: custom_facial_expression_dataset/train\sad\Training_151.jpg
Skipping image: custom_facial_expression_dataset/train\sad\Training_152.jpg
Skipping image: custom_facial_expression_dataset/train\sad\Training_153.jpg
Skipping image: custom_facial_expression_dataset/train\sad\Training_155.jpg
Skipping image: custom_facial_expression_dataset/train\sad\Training_156.jpg
Skipping image: custom_facial_expression_dataset/train\sad\Training_158.jpg
Skipping image: custom_facial_expression_dataset/train\sad\Training_159.jpg
Skipping image: custom_facial_expression_dataset/train\sad\Training_16.jpg
Skipping image: custom_facial_expression_dataset/train\sad\Training_160.jpg
Skipping image: custom_facial_expression_dataset/train\sad\Training_165.jpg
Skipping image: custom_facial_expression_dataset/train\sad\Training_167.jpg
Skipping image: custom_facial_expression_dataset/train\sad\Training_169.jpg
Skipping image: custom_facial_expression_dataset/train\sad\Training_170.jpg
Skipping image: custom_facial_expression_dataset/train\sad\Training_171.jpg
Skipping image: custom_facial_expression_dataset/train\sad\Training_174.jpg
Skipping image: custom_facial_expression_dataset/train\sad\Training_175.jpg
Skipping image: custom_facial_expression_dataset/train\sad\Training_177.jpg
Skipping image: custom_facial_expression_dataset/train\sad\Training_178.jpg
Skipping image: custom_facial_expression_dataset/train\sad\Training_179.jpg
Skipping image: custom_facial_expression_dataset/train\sad\Training_180.jpg
Skipping image: custom_facial_expression_dataset/train\sad\Training_182.jpg
Skipping image: custom_facial_expression_dataset/train\sad\Training_184.jpg
```

```
Skipping image: custom_facial_expression_dataset/train\sad\Training_185.jpg
Skipping image: custom_facial_expression_dataset/train\sad\Training_186.jpg
Skipping image: custom_facial_expression_dataset/train\sad\Training_191.jpg
Skipping image: custom_facial_expression_dataset/train\sad\Training_193.jpg
Skipping image: custom_facial_expression_dataset/train\sad\Training_21.jpg
Skipping image: custom_facial_expression_dataset/train\sad\Training_22.jpg
Skipping image: custom_facial_expression_dataset/train\sad\Training_31.jpg
Skipping image: custom_facial_expression_dataset/train\sad\Training_33.jpg
Skipping image: custom_facial_expression_dataset/train\sad\Training_38.jpg
Skipping image: custom_facial_expression_dataset/train\sad\Training_45.jpg
Skipping image: custom_facial_expression_dataset/train\sad\Training_51.jpg
Skipping image: custom_facial_expression_dataset/train\sad\Training_58.jpg
Skipping image: custom_facial_expression_dataset/train\sad\Training_63.jpg
Skipping image: custom_facial_expression_dataset/train\sad\Training_7.jpg
Skipping image: custom_facial_expression_dataset/train\sad\Training_8.jpg
Skipping image: custom_facial_expression_dataset/train\sad\Training_82.jpg
Skipping image: custom_facial_expression_dataset/train\sad\Training_83.jpg
Skipping image: custom_facial_expression_dataset/train\sad\Training_85.jpg
Skipping image: custom_facial_expression_dataset/train\sad\Training_86.jpg
Skipping image: custom_facial_expression_dataset/train\sad\Training_87.jpg
Skipping image: custom_facial_expression_dataset/train\sad\Training_9.jpg
Skipping image: custom_facial_expression_dataset/train\confused\Training_10.jpg
Skipping image: custom_facial_expression_dataset/train\confused\Training_101.jpg
Skipping image: custom_facial_expression_dataset/train\confused\Training_102.jpg
Skipping image: custom_facial_expression_dataset/train\confused\Training_11.jpg
Skipping image: custom_facial_expression_dataset/train\confused\Training_113.jpg
Skipping image: custom_facial_expression_dataset/train\confused\Training_116.jpg
Skipping image: custom_facial_expression_dataset/train\confused\Training_118.jpg
Skipping image: custom_facial_expression_dataset/train\confused\Training_122.jpg
Skipping image: custom_facial_expression_dataset/train\confused\Training_123.jpg
Skipping image: custom_facial_expression_dataset/train\confused\Training_126.jpg
Skipping image: custom_facial_expression_dataset/train\confused\Training_128.jpg
Skipping image: custom_facial_expression_dataset/train\confused\Training_129.jpg
Skipping image: custom_facial_expression_dataset/train\confused\Training_130.jpg
Skipping image: custom_facial_expression_dataset/train\confused\Training_131.jpg
Skipping image: custom_facial_expression_dataset/train\confused\Training_132.jpg
Skipping image: custom_facial_expression_dataset/train\confused\Training_133.jpg
Skipping image: custom_facial_expression_dataset/train\confused\Training_136.jpg
Skipping image: custom_facial_expression_dataset/train\confused\Training_137.jpg
Skipping image: custom_facial_expression_dataset/train\confused\Training_138.jpg
Skipping image: custom_facial_expression_dataset/train\confused\Training_139.jpg
Skipping image: custom_facial_expression_dataset/train\confused\Training_140.jpg
Skipping image: custom_facial_expression_dataset/train\confused\Training_141.jpg
Skipping image: custom_facial_expression_dataset/train\confused\Training_142.jpg
Skipping image: custom_facial_expression_dataset/train\confused\Training_144.jpg
Skipping image: custom_facial_expression_dataset/train\confused\Training_145.jpg
Skipping image: custom_facial_expression_dataset/train\confused\Training_15.jpg
Skipping image: custom_facial_expression_dataset/train\confused\Training_154.jpg
```

```
Skipping image: custom_facial_expression_dataset/train\confused\Training_158.jpg
Skipping image: custom_facial_expression_dataset/train\confused\Training_159.jpg
Skipping image: custom_facial_expression_dataset/train\confused\Training_16.jpg
Skipping image: custom_facial_expression_dataset/train\confused\Training_161.jpg
Skipping image: custom_facial_expression_dataset/train\confused\Training_162.jpg
Skipping image: custom_facial_expression_dataset/train\confused\Training_163.jpg
Skipping image: custom_facial_expression_dataset/train\confused\Training_164.jpg
Skipping image: custom_facial_expression_dataset/train\confused\Training_165.jpg
Skipping image: custom_facial_expression_dataset/train\confused\Training_166.jpg
Skipping image: custom_facial_expression_dataset/train\confused\Training_167.jpg
Skipping image: custom_facial_expression_dataset/train\confused\Training_168.jpg
Skipping image: custom_facial_expression_dataset/train\confused\Training_169.jpg
Skipping image: custom_facial_expression_dataset/train\confused\Training_170.jpg
Skipping image: custom_facial_expression_dataset/train\confused\Training_172.jpg
Skipping image: custom_facial_expression_dataset/train\confused\Training_175.jpg
Skipping image: custom_facial_expression_dataset/train\confused\Training_176.jpg
Skipping image: custom_facial_expression_dataset/train\confused\Training_177.jpg
Skipping image: custom_facial_expression_dataset/train\confused\Training_178.jpg
Skipping image: custom_facial_expression_dataset/train\confused\Training_182.jpg
Skipping image: custom_facial_expression_dataset/train\confused\Training_183.jpg
Skipping image: custom_facial_expression_dataset/train\confused\Training_184.jpg
Skipping image: custom_facial_expression_dataset/train\confused\Training_188.jpg
Skipping image: custom_facial_expression_dataset/train\confused\Training_189.jpg
Skipping image: custom_facial_expression_dataset/train\confused\Training_191.jpg
Skipping image: custom_facial_expression_dataset/train\confused\Training_192.jpg
Skipping image: custom_facial_expression_dataset/train\confused\Training_193.jpg
Skipping image: custom_facial_expression_dataset/train\confused\Training_194.jpg
Skipping image: custom_facial_expression_dataset/train\confused\Training_195.jpg
Skipping image: custom_facial_expression_dataset/train\confused\Training_196.jpg
Skipping image: custom_facial_expression_dataset/train\confused\Training_198.jpg
Skipping image: custom_facial_expression_dataset/train\confused\Training_199.jpg
Skipping image: custom_facial_expression_dataset/train\confused\Training_20.jpg
Skipping image: custom_facial_expression_dataset/train\confused\Training_200.jpg
Skipping image: custom_facial_expression_dataset/train\confused\Training_22.jpg
Skipping image: custom_facial_expression_dataset/train\confused\Training_23.jpg
Skipping image: custom_facial_expression_dataset/train\confused\Training_25.jpg
Skipping image: custom_facial_expression_dataset/train\confused\Training_27.jpg
Skipping image: custom_facial_expression_dataset/train\confused\Training_28.jpg
Skipping image: custom_facial_expression_dataset/train\confused\Training_29.jpg
Skipping image: custom_facial_expression_dataset/train\confused\Training_3.jpg
Skipping image: custom_facial_expression_dataset/train\confused\Training_30.jpg
Skipping image: custom_facial_expression_dataset/train\confused\Training_34.jpg
Skipping image: custom_facial_expression_dataset/train\confused\Training_36.jpg
Skipping image: custom_facial_expression_dataset/train\confused\Training_37.jpg
Skipping image: custom_facial_expression_dataset/train\confused\Training_38.jpg
Skipping image: custom_facial_expression_dataset/train\confused\Training_41.jpg
Skipping image: custom_facial_expression_dataset/train\confused\Training_42.jpg
Skipping image: custom_facial_expression_dataset/train\confused\Training_43.jpg
```

```
Skipping image: custom_facial_expression_dataset/train\confused\Training_46.jpg
Skipping image: custom_facial_expression_dataset/train\confused\Training_47.jpg
Skipping image: custom_facial_expression_dataset/train\confused\Training_49.jpg
Skipping image: custom_facial_expression_dataset/train\confused\Training_5.jpg
Skipping image: custom_facial_expression_dataset/train\confused\Training_50.jpg
Skipping image: custom_facial_expression_dataset/train\confused\Training_53.jpg
Skipping image: custom_facial_expression_dataset/train\confused\Training_54.jpg
Skipping image: custom_facial_expression_dataset/train\confused\Training_55.jpg
Skipping image: custom_facial_expression_dataset/train\confused\Training_56.jpg
Skipping image: custom_facial_expression_dataset/train\confused\Training_58.jpg
Skipping image: custom_facial_expression_dataset/train\confused\Training_59.jpg
Skipping image: custom_facial_expression_dataset/train\confused\Training_6.jpg
Skipping image: custom_facial_expression_dataset/train\confused\Training_64.jpg
Skipping image: custom_facial_expression_dataset/train\confused\Training_67.jpg
Skipping image: custom_facial_expression_dataset/train\confused\Training_68.jpg
Skipping image: custom_facial_expression_dataset/train\confused\Training_69.jpg
Skipping image: custom_facial_expression_dataset/train\confused\Training_8.jpg
Skipping image: custom_facial_expression_dataset/train\confused\Training_81.jpg
Skipping image: custom_facial_expression_dataset/train\confused\Training_82.jpg
Skipping image: custom_facial_expression_dataset/train\confused\Training_90.jpg
Skipping image: custom_facial_expression_dataset/train\confused\Training_91.jpg
Skipping image: custom_facial_expression_dataset/train\confused\Training_92.jpg
Skipping image: custom_facial_expression_dataset/train\confused\Training_93.jpg
Skipping image: custom_facial_expression_dataset/others\20230604_010504.jpg
Saved!
```

[ ]:
```python
X_train, X_test, y_train, y_test = train_test_split(embeddings, labels,␣
  ↪test_size=0.2, random_state=42)

svm_model = SVC(kernel='linear', probability=True)
svm_model.fit(X_train, y_train)

y_pred = svm_model.predict(X_test)
accuracy = accuracy_score(y_test, y_pred)
print(f"Face Recognition Accuracy: {accuracy * 100:.2f}%")

with open("pkl/face_recognition_model.pkl", "wb") as f:
  pickle.dump(svm_model, f)
print("Saved!")
```

```
Face Recognition Accuracy: 98.14%
Saved!
```

## 0.1 Mood Detection

### 0.1.1 Load Dataset

[8]:
```
TRAIN_DIR = ('custom_facial_expression_dataset/train/')
TEST_DIR = ('custom_facial_expression_dataset/test/')
VAL_DIR = ('custom_facial_expression_dataset/validation/')
```

[9]:
```
def load_data(dir_path, IMG_SIZE):

    X = []
    y = []
    i = 0
    labels = dict()
    for path in tqdm(sorted(os.listdir(dir_path))):
        if not path.startswith('.'):
            labels[i] = path
            for file in os.listdir(dir_path + path):
                if not file.startswith('.'):
                    img = cv2.imread(dir_path + path + '/' + file)
                    img = img.astype('float32') / 255
                    resized = cv2.resize(img, IMG_SIZE, interpolation = cv2.
 ↪INTER_AREA)
                    X.append(resized)
                    y.append(i)
            i += 1
    X = np.array(X)
    y = np.array(y)
    print(f'{len(X)} images loaded from {dir_path} directory.')
    return X, y, labels

IMG_SIZE = 48
```

[10]:
```
X_train, y_train, train_labels = load_data(TRAIN_DIR, (IMG_SIZE, IMG_SIZE))
X_test, y_test, test_labels = load_data(TEST_DIR, (IMG_SIZE, IMG_SIZE))
X_val, y_val, val_labels = load_data(VAL_DIR, (IMG_SIZE, IMG_SIZE))
```

```
  0%|          | 0/4 [00:00<?, ?it/s]100%|      | 4/4 [00:26<00:00,
6.72s/it]

800 images loaded from custom_facial_expression_dataset/train/ directory.

100%|      | 4/4 [00:03<00:00,  1.01it/s]

120 images loaded from custom_facial_expression_dataset/test/ directory.

100%|      | 4/4 [00:04<00:00,  1.08s/it]

120 images loaded from custom_facial_expression_dataset/validation/ directory.
```

[11]:
```
train_labels
```

[11]:
```
{0: 'angry', 1: 'confused', 2: 'happy', 3: 'sad'}
```

[12]:
```
test_labels
```

[12]:
```
{0: 'angry', 1: 'confused', 2: 'happy', 3: 'sad'}
```

[13]:
```
val_labels
```

[13]:
```
{0: 'angry', 1: 'confused', 2: 'happy', 3: 'sad'}
```

[14]:
```
from keras.utils.np_utils import to_categorical

y_train = to_categorical(y_train, num_classes=4)
y_train.shape
```

[14]:
```
(800, 4)
```

[15]:
```
y_test = to_categorical(y_test, num_classes=4)
y_test.shape
```

[15]:
```
(120, 4)
```

[16]:
```
y_val = to_categorical(y_val, num_classes=4)
y_val.shape
```

[16]:
```
(120, 4)
```

### 0.1.2 Creating Model

[122]:
```
conv_base = VGG16(weights='imagenet',
                  include_top=False,
                  input_shape=(IMG_SIZE, IMG_SIZE, 3)
)

for layer in conv_base.layers[:15]:
    layer.trainable = False

for layer in conv_base.layers[15:]:
    layer.trainable = True

conv_base.summary()
```

```
Model: "vgg16"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 input_13 (InputLayer)       [(None, 48, 48, 3)]       0
```

```
 block1_conv1 (Conv2D)       (None, 48, 48, 64)        1792

 block1_conv2 (Conv2D)       (None, 48, 48, 64)        36928

 block1_pool (MaxPooling2D)  (None, 24, 24, 64)        0

 block2_conv1 (Conv2D)       (None, 24, 24, 128)       73856

 block2_conv2 (Conv2D)       (None, 24, 24, 128)       147584

 block2_pool (MaxPooling2D)  (None, 12, 12, 128)       0

 block3_conv1 (Conv2D)       (None, 12, 12, 256)       295168

 block3_conv2 (Conv2D)       (None, 12, 12, 256)       590080

 block3_conv3 (Conv2D)       (None, 12, 12, 256)       590080

 block3_pool (MaxPooling2D)  (None, 6, 6, 256)         0

 block4_conv1 (Conv2D)       (None, 6, 6, 512)         1180160

 block4_conv2 (Conv2D)       (None, 6, 6, 512)         2359808

 block4_conv3 (Conv2D)       (None, 6, 6, 512)         2359808

 block4_pool (MaxPooling2D)  (None, 3, 3, 512)         0

 block5_conv1 (Conv2D)       (None, 3, 3, 512)         2359808

 block5_conv2 (Conv2D)       (None, 3, 3, 512)         2359808

 block5_conv3 (Conv2D)       (None, 3, 3, 512)         2359808

 block5_pool (MaxPooling2D)  (None, 1, 1, 512)         0

=================================================================
Total params: 14,714,688
Trainable params: 7,079,424
Non-trainable params: 7,635,264
_________________________________________________________________
```

[123]:
```
model = Sequential([
    conv_base,
    Flatten(),
    Dense(512, activation='relu'),
```

```
    Dropout(0.4),
    Dense(4, activation='softmax')
])

model.compile(
    optimizer=Adam(learning_rate=1e-5),
    loss='categorical_crossentropy',
    metrics=['accuracy']
)

model.summary()
```

```
Model: "sequential_13"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 vgg16 (Functional)          (None, 1, 1, 512)         14714688

 flatten_13 (Flatten)        (None, 512)               0

 dense_26 (Dense)            (None, 512)               262656

 dropout_13 (Dropout)        (None, 512)               0

 dense_27 (Dense)            (None, 4)                 2052

=================================================================
Total params: 14,979,396
Trainable params: 7,344,132
Non-trainable params: 7,635,264
_________________________________________________________________
```

### 0.1.3 Data Augmentation

[124]:
```
datagen = ImageDataGenerator(
    rotation_range=20,
    width_shift_range=0.2,
    height_shift_range=0.2,
    shear_range=0.2,
    zoom_range=0.2,
    horizontal_flip=True,
    fill_mode='nearest'
)

datagen.fit(X_train)
```

### 0.1.4 Train Model

[125]:
```
EPOCHS = 50
BATCH_SIZE = 64

history = model.fit(
    datagen.flow(X_train, y_train, batch_size=BATCH_SIZE),
    validation_data = (X_val, y_val),
    epochs=EPOCHS,
    batch_size=BATCH_SIZE,
    verbose=1
)
```

```
Epoch 1/50
13/13 [==============================] - 2s 126ms/step - loss: 1.4976 -
accuracy: 0.2587 - val_loss: 1.3841 - val_accuracy: 0.2917
Epoch 2/50
13/13 [==============================] - 1s 51ms/step - loss: 1.4240 - accuracy:
0.2912 - val_loss: 1.3688 - val_accuracy: 0.3250
Epoch 3/50
13/13 [==============================] - 1s 52ms/step - loss: 1.3925 - accuracy:
0.3075 - val_loss: 1.3546 - val_accuracy: 0.3750
Epoch 4/50
13/13 [==============================] - 1s 52ms/step - loss: 1.3382 - accuracy:
0.3600 - val_loss: 1.3328 - val_accuracy: 0.4500
Epoch 5/50
13/13 [==============================] - 1s 50ms/step - loss: 1.3354 - accuracy:
0.3562 - val_loss: 1.3178 - val_accuracy: 0.4417
Epoch 6/50
13/13 [==============================] - 1s 52ms/step - loss: 1.2568 - accuracy:
0.4175 - val_loss: 1.2867 - val_accuracy: 0.4333
Epoch 7/50
13/13 [==============================] - 1s 54ms/step - loss: 1.2503 - accuracy:
0.4375 - val_loss: 1.2783 - val_accuracy: 0.4250
Epoch 8/50
13/13 [==============================] - 1s 57ms/step - loss: 1.2155 - accuracy:
0.4650 - val_loss: 1.2501 - val_accuracy: 0.5167
Epoch 9/50
13/13 [==============================] - 1s 54ms/step - loss: 1.1864 - accuracy:
0.4762 - val_loss: 1.2138 - val_accuracy: 0.4917
Epoch 10/50
13/13 [==============================] - 1s 62ms/step - loss: 1.1407 - accuracy:
0.5125 - val_loss: 1.2030 - val_accuracy: 0.4583
Epoch 11/50
13/13 [==============================] - 1s 54ms/step - loss: 1.1337 - accuracy:
0.5125 - val_loss: 1.1679 - val_accuracy: 0.5417
Epoch 12/50
13/13 [==============================] - 1s 52ms/step - loss: 1.0737 - accuracy:
```

```
0.5387 - val_loss: 1.1639 - val_accuracy: 0.4667
Epoch 13/50
13/13 [==============================] - 1s 64ms/step - loss: 1.0606 - accuracy:
0.5425 - val_loss: 1.1435 - val_accuracy: 0.4917
Epoch 14/50
13/13 [==============================] - 1s 57ms/step - loss: 1.0351 - accuracy:
0.5500 - val_loss: 1.1662 - val_accuracy: 0.4583
Epoch 15/50
13/13 [==============================] - 1s 56ms/step - loss: 0.9881 - accuracy:
0.5888 - val_loss: 1.1383 - val_accuracy: 0.4417
Epoch 16/50
13/13 [==============================] - 1s 52ms/step - loss: 0.9582 - accuracy:
0.6175 - val_loss: 1.0923 - val_accuracy: 0.5083
Epoch 17/50
13/13 [==============================] - 1s 53ms/step - loss: 0.9173 - accuracy:
0.6313 - val_loss: 1.0813 - val_accuracy: 0.5000
Epoch 18/50
13/13 [==============================] - 1s 53ms/step - loss: 0.9329 - accuracy:
0.6212 - val_loss: 1.0313 - val_accuracy: 0.6083
Epoch 19/50
13/13 [==============================] - 1s 54ms/step - loss: 0.9042 - accuracy:
0.6250 - val_loss: 1.0471 - val_accuracy: 0.5750
Epoch 20/50
13/13 [==============================] - 1s 52ms/step - loss: 0.8628 - accuracy:
0.6513 - val_loss: 1.0306 - val_accuracy: 0.6000
Epoch 21/50
13/13 [==============================] - 1s 53ms/step - loss: 0.8555 - accuracy:
0.6612 - val_loss: 1.0903 - val_accuracy: 0.5000
Epoch 22/50
13/13 [==============================] - 1s 54ms/step - loss: 0.8373 - accuracy:
0.6525 - val_loss: 1.0456 - val_accuracy: 0.5667
Epoch 23/50
13/13 [==============================] - 1s 53ms/step - loss: 0.8597 - accuracy:
0.6587 - val_loss: 1.0393 - val_accuracy: 0.5750
Epoch 24/50
13/13 [==============================] - 1s 53ms/step - loss: 0.8386 - accuracy:
0.6575 - val_loss: 0.9881 - val_accuracy: 0.5750
Epoch 25/50
13/13 [==============================] - 1s 53ms/step - loss: 0.7686 - accuracy:
0.7013 - val_loss: 1.0058 - val_accuracy: 0.5750
Epoch 26/50
13/13 [==============================] - 1s 54ms/step - loss: 0.7505 - accuracy:
0.6938 - val_loss: 1.0124 - val_accuracy: 0.6000
Epoch 27/50
13/13 [==============================] - 1s 57ms/step - loss: 0.7819 - accuracy:
0.6825 - val_loss: 1.0670 - val_accuracy: 0.5333
Epoch 28/50
13/13 [==============================] - 1s 53ms/step - loss: 0.7665 - accuracy:
```

```
0.6837 - val_loss: 1.0938 - val_accuracy: 0.4917
Epoch 29/50
13/13 [==============================] - 1s 54ms/step - loss: 0.7408 - accuracy:
0.6975 - val_loss: 0.9218 - val_accuracy: 0.6500
Epoch 30/50
13/13 [==============================] - 1s 52ms/step - loss: 0.7352 - accuracy:
0.7175 - val_loss: 0.9751 - val_accuracy: 0.6083
Epoch 31/50
13/13 [==============================] - 1s 51ms/step - loss: 0.7218 - accuracy:
0.7100 - val_loss: 0.9992 - val_accuracy: 0.6083
Epoch 32/50
13/13 [==============================] - 1s 53ms/step - loss: 0.6734 - accuracy:
0.7262 - val_loss: 0.9584 - val_accuracy: 0.6167
Epoch 33/50
13/13 [==============================] - 1s 54ms/step - loss: 0.6830 - accuracy:
0.7287 - val_loss: 1.0450 - val_accuracy: 0.5667
Epoch 34/50
13/13 [==============================] - 1s 54ms/step - loss: 0.6708 - accuracy:
0.7287 - val_loss: 0.9440 - val_accuracy: 0.5917
Epoch 35/50
13/13 [==============================] - 1s 51ms/step - loss: 0.6259 - accuracy:
0.7738 - val_loss: 0.9291 - val_accuracy: 0.6167
Epoch 36/50
13/13 [==============================] - 1s 54ms/step - loss: 0.6225 - accuracy:
0.7412 - val_loss: 1.0581 - val_accuracy: 0.6333
Epoch 37/50
13/13 [==============================] - 1s 54ms/step - loss: 0.6341 - accuracy:
0.7437 - val_loss: 0.9643 - val_accuracy: 0.6500
Epoch 38/50
13/13 [==============================] - 1s 52ms/step - loss: 0.6018 - accuracy:
0.7725 - val_loss: 0.9027 - val_accuracy: 0.6750
Epoch 39/50
13/13 [==============================] - 1s 52ms/step - loss: 0.5724 - accuracy:
0.7663 - val_loss: 0.9509 - val_accuracy: 0.6583
Epoch 40/50
13/13 [==============================] - 1s 52ms/step - loss: 0.5979 - accuracy:
0.7763 - val_loss: 1.0357 - val_accuracy: 0.6167
Epoch 41/50
13/13 [==============================] - 1s 54ms/step - loss: 0.5895 - accuracy:
0.7775 - val_loss: 0.9343 - val_accuracy: 0.6667
Epoch 42/50
13/13 [==============================] - 1s 54ms/step - loss: 0.5793 - accuracy:
0.7788 - val_loss: 0.9821 - val_accuracy: 0.5833
Epoch 43/50
13/13 [==============================] - 1s 53ms/step - loss: 0.5775 - accuracy:
0.7763 - val_loss: 0.9667 - val_accuracy: 0.6250
Epoch 44/50
13/13 [==============================] - 1s 51ms/step - loss: 0.5998 - accuracy:
```

```
0.7675 - val_loss: 0.9110 - val_accuracy: 0.7083
Epoch 45/50
13/13 [==============================] - 1s 52ms/step - loss: 0.5500 - accuracy:
0.7812 - val_loss: 0.9139 - val_accuracy: 0.6750
Epoch 46/50
13/13 [==============================] - 1s 52ms/step - loss: 0.5622 - accuracy:
0.7775 - val_loss: 0.8670 - val_accuracy: 0.6667
Epoch 47/50
13/13 [==============================] - 1s 53ms/step - loss: 0.5550 - accuracy:
0.7700 - val_loss: 0.9434 - val_accuracy: 0.6667
Epoch 48/50
13/13 [==============================] - 1s 55ms/step - loss: 0.5509 - accuracy:
0.7887 - val_loss: 0.9269 - val_accuracy: 0.7083
Epoch 49/50
13/13 [==============================] - 1s 53ms/step - loss: 0.5649 - accuracy:
0.7775 - val_loss: 0.9939 - val_accuracy: 0.5750
Epoch 50/50
13/13 [==============================] - 1s 52ms/step - loss: 0.5454 - accuracy:
0.7788 - val_loss: 0.8716 - val_accuracy: 0.7000
```

### 0.2 Evaluation and Testing

[128]:
```
model.save('model/new_dataset/new_dataset_mood_classification_vgg16_5.h5')
```

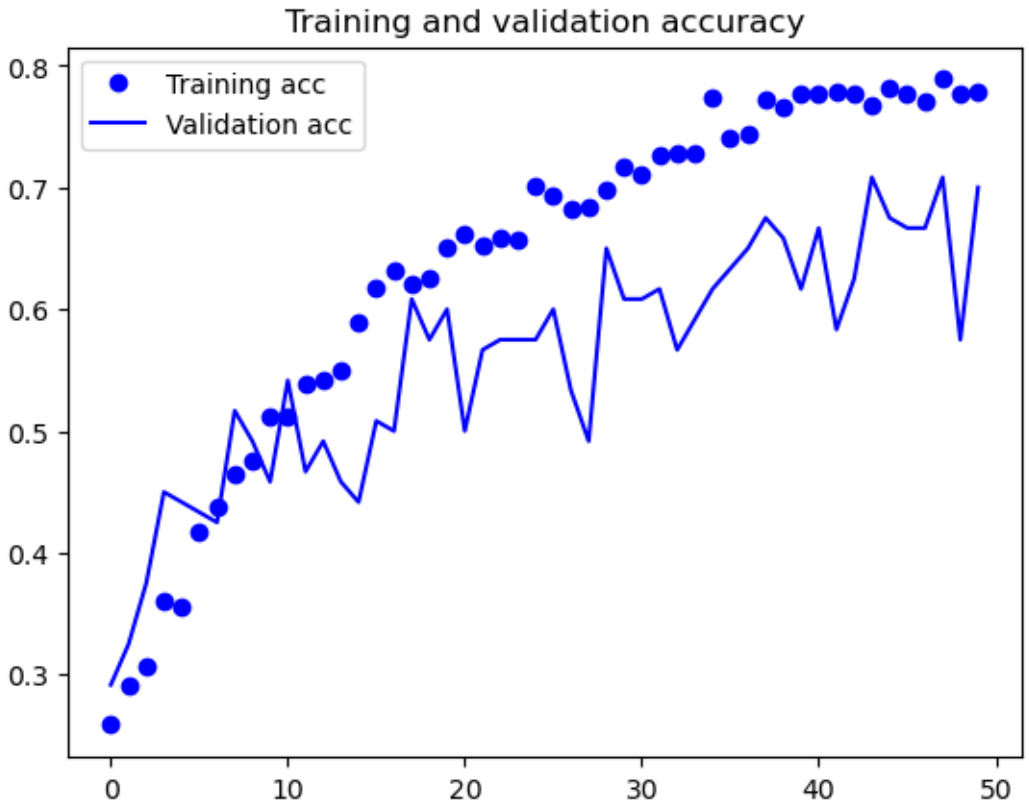
[ ]:
```
# Save the history to compare with other model
# with open("pkl/new_dataset/new_dataset_mood_classification_vgg16_5.pkl",␣
 ↪"wb") as f:
#     pickle.dump(history.history, f)
```

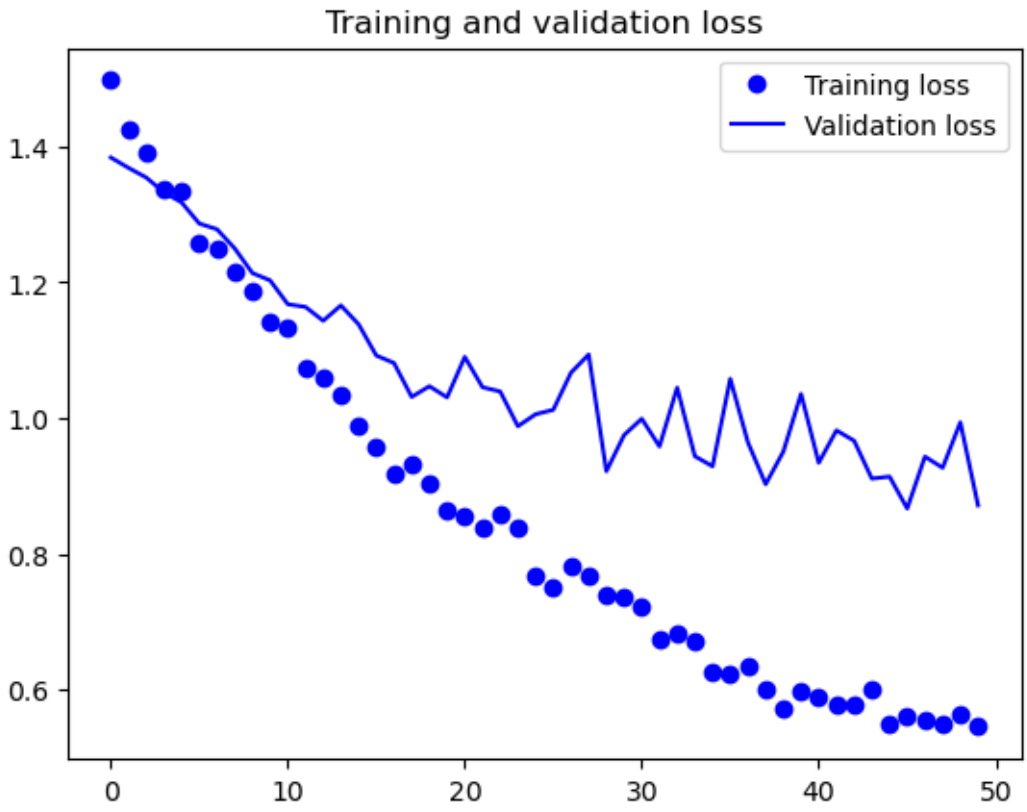
[3]:
```
model = load_model('model/new_dataset/new_dataset_mood_classification_vgg16_5.
 ↪h5')
```

[4]:
```
# Load the history
with open("pkl/new_dataset/new_dataset_mood_classification_vgg16_5.pkl", "rb")␣
 ↪as f:
    history = pickle.load(f)
```

[5]:
```
acc = history['accuracy']
val_acc = history['val_accuracy']
loss = history['loss']
val_loss = history['val_loss']

epochs = range(len(acc))

plt.plot(epochs, acc, 'bo', label='Training acc')
plt.plot(epochs, val_acc, 'b', label='Validation acc')
plt.title('Training and validation accuracy')
```

```
plt.legend()

plt.figure()
plt.plot(epochs, loss, 'bo', label='Training loss')
plt.plot(epochs, val_loss, 'b', label='Validation loss')
plt.title('Training and validation loss')
plt.legend()

plt.show()
```



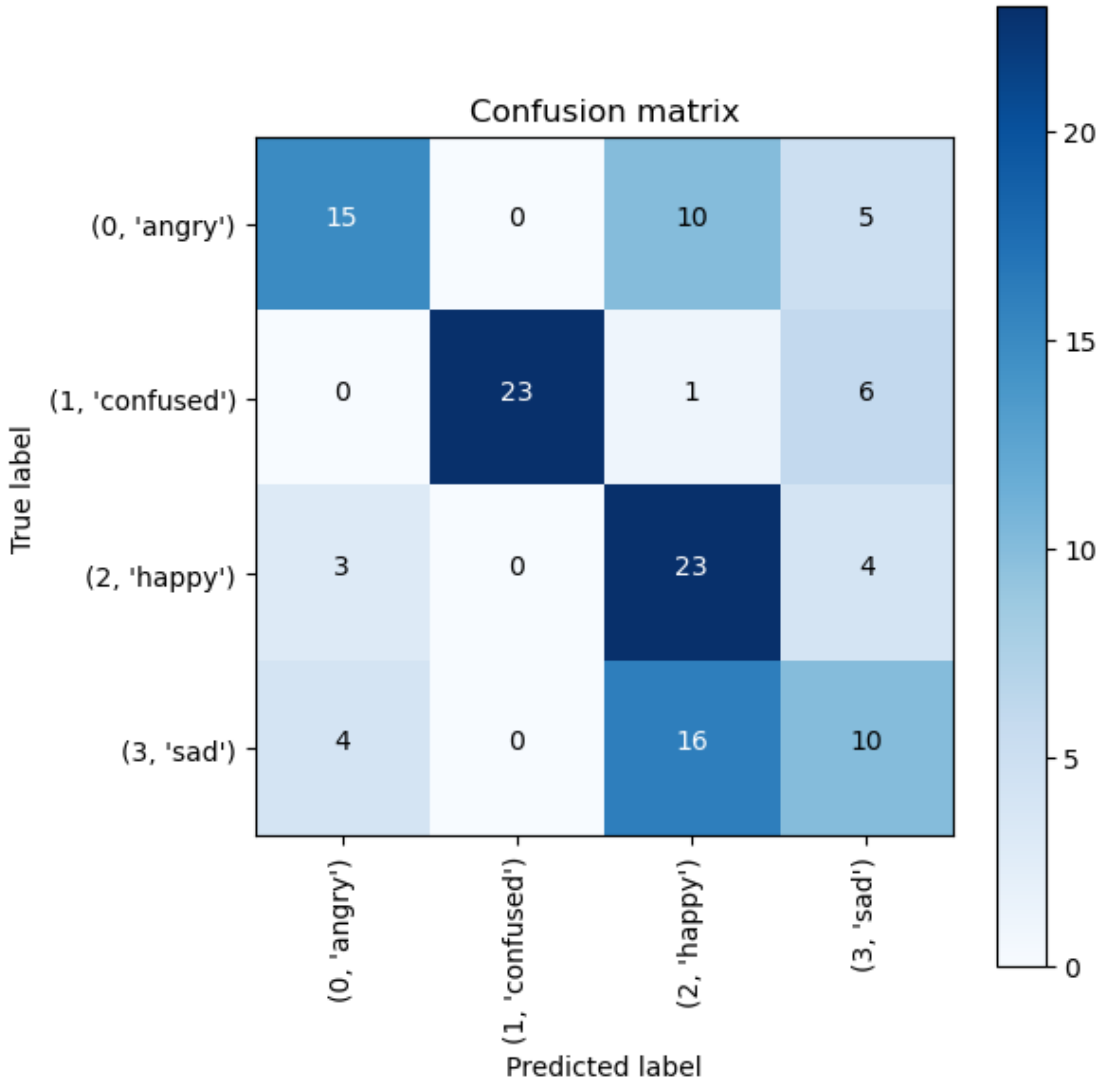



[6]:
```
def plot_confusion_matrix(cm, classes,
                          normalize=False,
                          title='Confusion matrix',
                          cmap=plt.cm.Blues):

    plt.figure(figsize = (6,6))
    plt.imshow(cm, interpolation='nearest', cmap=cmap)
    plt.title(title)
    plt.colorbar()
    tick_marks = np.arange(len(classes))
    plt.xticks(tick_marks, classes, rotation=90)
    plt.yticks(tick_marks, classes)
    if normalize:
        cm = cm.astype('float') / cm.sum(axis=1)[:, np.newaxis]

    thresh = cm.max() / 2.
    cm = np.round(cm,2)
    for i, j in itertools.product(range(cm.shape[0]), range(cm.shape[1])):
        plt.text(j, i, cm[i, j],
                 horizontalalignment="center",
                 color="white" if cm[i, j] > thresh else "black")
```

```
    plt.tight_layout()
    plt.ylabel('True label')
    plt.xlabel('Predicted label')
    plt.show()
```

[17]:
```
predictions = model.predict(X_test)
y_pred = [np.argmax(probas) for probas in predictions]

y_test_class_indices = np.argmax(y_test, axis=1)
accuracy = accuracy_score(y_test_class_indices, y_pred)
print('Test Accuracy = %.2f' % accuracy)

confusion_mtx = confusion_matrix(y_test_class_indices, y_pred)
cm = plot_confusion_matrix(confusion_mtx, classes = list(test_labels.items()),␣
 ↪normalize=False)
```

```
4/4 [==============================] - 40s 2s/step
Test Accuracy = 0.59
```




[ ]:
```
with open("pkl/face_recognition_model.pkl", "rb") as f:
    face_recognizer = pickle.load(f)
with open("pkl/face_embeddings.pkl", "rb") as f:
    embeddings_data = pickle.load(f)
embeddings, labels = embeddings_data

mood_model = load_model("model/mood_classification_vgg16.h5")

face_cascade = cv2.CascadeClassifier(cv2.data.haarcascades +␣
 ↪"haarcascade_frontalface_default.xml")

emotion_labels = ["angry", "happy", "sad", "confused"]
```

```
webcam_capture = cv2.VideoCapture(0)

while True:
    ret, frame = webcam_capture.read()
    if not ret:
        break
    gray = cv2.cvtColor(frame, cv2.COLOR_BGR2GRAY)
    faces = face_cascade.detectMultiScale(gray, 1.3, 5)

    for (x, y, w, h) in faces:
        face = frame[y: y+h, x: x+w]

        label_text = "Unknown"

        try:
            embedding = DeepFace.represent(face, model_name="Facenet",␣
 ↪enforce_detection=False)[0]["embedding"]
            probs = face_recognizer.predict_proba([embedding])[0]
            pred_label = face_recognizer.predict([embedding])[0]
            face_conf = probs[1] if pred_label == 1 else probs[0]

            if pred_label == 1:
                try:
                    face_resized = cv2.resize(face, (48, 48))
                    face_array = img_to_array(face_resized) / 255.0
                    face_array = np.expand_dims(face_array, axis=0)

                    prediction = mood_model.predict(face_array, verbose=0)[0]
                    emotion = emotion_labels[np.argmax(prediction)]
                    emotion_conf = np.max(prediction)

                    label_text = f"{emotion} ({emotion_conf*100:.1f}%)"
                except:
                    label_text = "(Error)"

            else:
                label_text = f"Unknown ({face_conf*100:.1f}%)"

        except:
            label_text = "Recognition Error"

        cv2.rectangle(frame, (x, y), (x+w, y+h), (0, 255, 0), 2)
        cv2.putText(frame, label_text, (x, y-10),
                    cv2.FONT_HERSHEY_SIMPLEX, 0.8, (0, 255, 0), 2)

    cv2.imshow("Face Recognition & Mood Detection", frame)
```

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

webcam_capture.release()
cv2.destroyAllWindows()
```