FACE EMOTION DETECTION

ABSTRACT

These Human facial expressions convey a lot of information visually rather than articulately. Facial expression recognition plays a crucial role in the area of human-machine interaction. Automatic facial expression recognition system has many applications including, but not limited to, human behavior understanding, detection of mental disorders, and synthetic humanexpressions. Recognition of facial expression by computer with high recognition rate is still a challenging task. Two popular methods utilized mostly in the literature for the automatic FER systems are based on geometry and appearance. Facial Expression Recognition usually performed in four-stages consisting of pre-processing, face detection, feature extraction, and expression classification. In this project we applied various deep learning methods (convolutional neural networks) to identify the key seven human emotions: anger, disgust, fear, happiness, sadness, surprise and neutrality.

HARDWARE AND SOFTWARE REQUIREMENTS

As the project is developed in python and used Anaconda for Python 3.6.5 and Spyder.

Anaconda

It is a free and open source distribution of the Python and R programming languagesfor data science and machine learning related applications (large-scale data processing, predictive analytics, scientific computing), that aims to simplify package management and deployment. Package versions are managed by the package management system *conda*. The Anaconda distribution is used by over 6 million users, and it includes more than 250 popular data science packages suitable for Windows, Linux, and Mac OS

JUPYTER NOTEBOOK

It is a open source web application that allows data scientist to create and share documents that include live code ,equations and other multi media resources.

Hardware Interfaces

Processor : Intel CORE i5 processor with minimum 2.9 GHz speed.

RAM : Minimum 4 GB.

Hard Disk : Minimum 500 GB

Software Interfaces

Microsoft Word 2003

Database Storage : Microsoft Excel

ALGORITHM

Step 1 : Collection of a data set of images. (In this case we are using FER2013 database of**35887 pre-cropped, 48-by-48-pixel grayscale images** of faces each labeled with one of the 7 emotion classes: anger, disgust, fear, happiness, sadness, surprise, and neutral.

Step 2 : Pre-processing of images.

Step 3 : Detection of a face from each image.

Step 4 : The cropped face is converted into grayscale images.

Step 5 : The pipeline ensures every image can be fed into the input layer as a (1, 48, 48) numpy array.

Step 5 : The numpy array gets passed into the Convolution2D layer.

Step 6 : Convolution generates feature maps.

Step 7 : Pooling method called MaxPooling2D that uses (2, 2) windows across the featuremap only keeping the maximum pixel value.

Step 8 : During training, Neural network Forward propagation and Backward propagation performed on the pixel values.

Step 9 : The Softmax function presents itself as a probability for each emotion class. The model is able to show the detail probability composition of the emotions in the face.

IMPLEMENTATION DETAILS

The dataset, used for training the model is from a Kaggle Facial Expression Recognition Challenge a few years back (FER2013). The data consists of 48x48 pixel grayscale images of faces. The faces have been automatically registered so that the face is more or less centered and occupies about the same amount of space in each image. The task is to categorize each face based on the emotion shown in the facial expression in to one of seven categories (0=Angry, 1=Disgust, 2=Fear, 3=Happy, 4=Sad, 5=Surprise, 6=Neutral).The training set consists of 28,709 examples. The public test set used for the leaderboard consists of 3,589 examples. The final test set, which was used to determine the winner of the competition, consists of another 3,589

SOURCE CODE

DATA SET

```
# import required packages
import cv2
from keras.models import Sequential
from keras.layers import Conv2D, MaxPooling2D, Dense, Dropout, Flatten
from keras.optimizers import Adam
from keras.preprocessing.image import ImageDataGenerator
# Initialize image data generator with rescaling
train_data_gen = ImageDataGenerator(rescale=1./255)
validation_data_gen = ImageDataGenerator(rescale=1./255)
# Preprocess all test images
train_generator = train_data_gen.flow_from_directory(
    r"C:\Users\mohan\Downloads\archive (1)\train",
    target_size=(48, 48),
    batch_size=64,
    color_mode="grayscale",
    class_mode='categorical')
validation_generator = validation_data_gen.flow_from_directory(
    r"C:\Users\mohan\Downloads\archive (1)\test",
    target_size=(48, 48),
    batch_size=64,
    color_mode="grayscale",
    class_mode='categorical')

import cv2
from tensorflow.keras.models import Sequential
```

```
from tensorflow.keras.layers import Conv2D, MaxPooling2D, Dropout, Flatten, Dense
from tensorflow.keras.optimizers import Adam
import tensorflow as tf
# create model structure
emotion_model = Sequential()
emotion_model.add(Conv2D(32, kernel_size=(3, 3), activation='relu', input_shape=(48, 48, 1)))
emotion_model.add(Conv2D(64, kernel_size=(3, 3), activation='relu'))
emotion_model.add(MaxPooling2D(pool_size=(2, 2)))
emotion_model.add(Dropout(0.25))
emotion_model.add(Conv2D(128, kernel_size=(3, 3), activation='relu'))
emotion_model.add(MaxPooling2D(pool_size=(2, 2)))
emotion_model.add(Conv2D(128, kernel_size=(3, 3), activation='relu'))
emotion_model.add(MaxPooling2D(pool_size=(2, 2)))
emotion_model.add(Dropout(0.25))
emotion_model.add(Flatten())

emotion_model.add(Dense(1024, activation='relu'))
emotion_model.add(Dropout(0.5))
emotion_model.add(Dense(7, activation='softmax'))
cv2.ocl.setUseOpenCL(False)
emotion_model.compile(loss='categorical_crossentropy',
optimizer=tf.keras.optimizers.legacy.Adam(lr=0.0001, decay=1e-6), metrics=['accuracy'])
emotion_model_info = emotion_model.fit_generator(
        train_generator,
        steps_per_epoch=28709 // 64,
        epoch
    s=20,
        validation_data=validation_generator,
        validation_steps=7178 // 64)
# save model structure in jason file
model_json =  emotion_model.to_json()
with open("emotion.json", "w") as json_file:
    json_file.write(model_json)
```

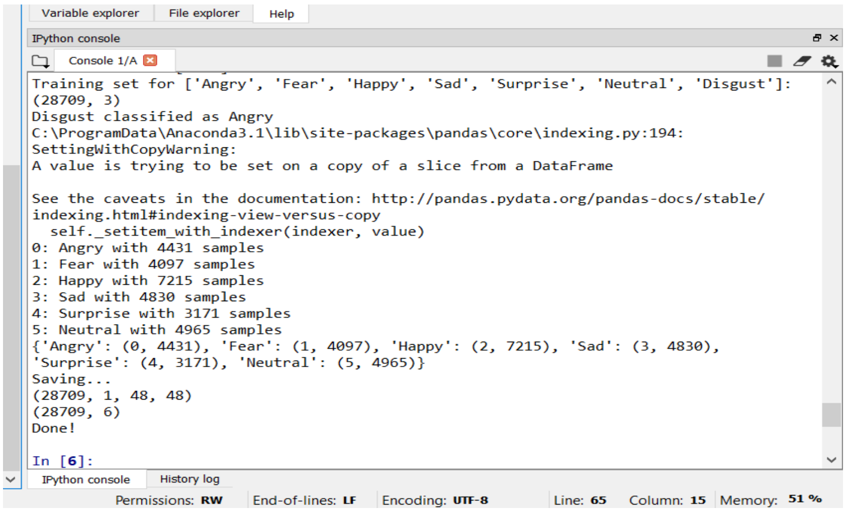
save trained model weight in h5 file emotion model.save_weights(‘emotion.h5’)

DETECTION

```
import cv2
import numpy as np
from keras.models import model_from_json
from IPython.display import display, Image
from PIL import Image as PILImage
from io import BytesIO
# Load the model architecture from JSON
with open("emotion.json", "r") as json_file:
    loaded_model_json = json_file.read()
    model = model_from_json(loaded_model_json)
# Load the weights into the model
model.load_weights(r"C:\Users\mohan\Downloads\emotion.h5")
# Define the emotions
emotions = ["Angry", "Disgust", "Fear", "Happy", "Sad", "Surprise", "Neutral"]
# Load the face cascade for detecting faces
face_cascade = cv2.CascadeClassifier(cv2.data.haarcascades + 'haarcascade_frontalface_default.xml')
# Function to detect and classify emotions
def detect_emotion(frame):
    gray = cv2.cvtColor(frame, cv2.COLOR_BGR2GRAY)
    faces = face_cascade.detectMultiScale(gray, scaleFactor=1.3, minNeighbors=5)
    for (x, y, w, h) in faces:
        face_roi = gray[y:y + h, x:x + w]
        face_roi = cv2.resize(face_roi, (48, 48))
        face_roi = face_roi.astype("float") / 255.0
        face_roi = np.expand_dims(face_roi, axis=0)
        face_roi = np.expand_dims(face_roi, axis=-1)
        emotion_pred = model.predict(face_roi)[0]
        emotion_label = emotions[np.argmax(emotion_pred)]
        # Display the detected emotion in the notebook
        display(f"Detected Emotion: {emotion_label}")
        cv2.rectangle(frame, (x, y), (x + w, y + h), (255, 0, 0), 2)
        cv2.putText(frame, emotion_label, (x, y - 10), cv2.FONT_HERSHEY_SIMPLEX, 0.9, (255, 0, 0), 2,
```

```
cv2.LINE_AA)
    return  frame
# Load an image
image_path = r"C:\Users\mohan\Downloads\WhatsApp Image 2023-09-17 at 9.06.47 PM.jpeg"
frame = cv2.imread(image_path)
if frame is not None:
    frame = detect_emotion(frame)
    # Convert the frame to a format that can be displayed in the notebook
    frame_pil = PILImage.fromarray(cv2.cvtColor(frame, cv2.COLOR_BGR2RGB))
    buffered = BytesIO()
    frame_pil.save(buffered, format="JPEG")
    display(Image(data=buffered.getvalue()))
else:
    print("Failed to load the image.")
```

1 RESULT



Variable explorer File explorer Help

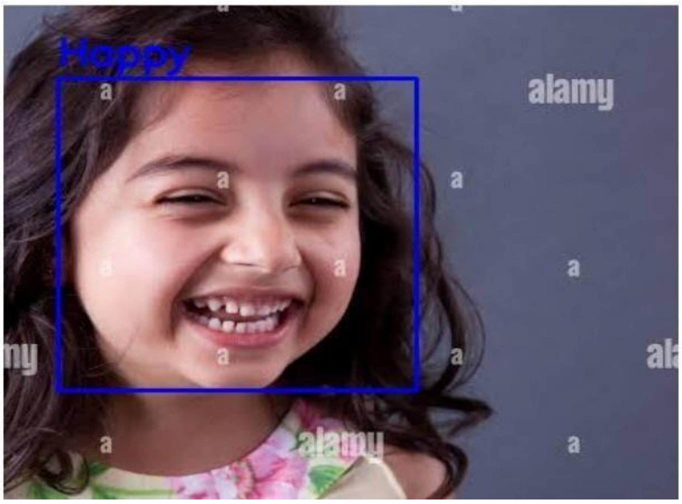
IPython console

Console 1/A

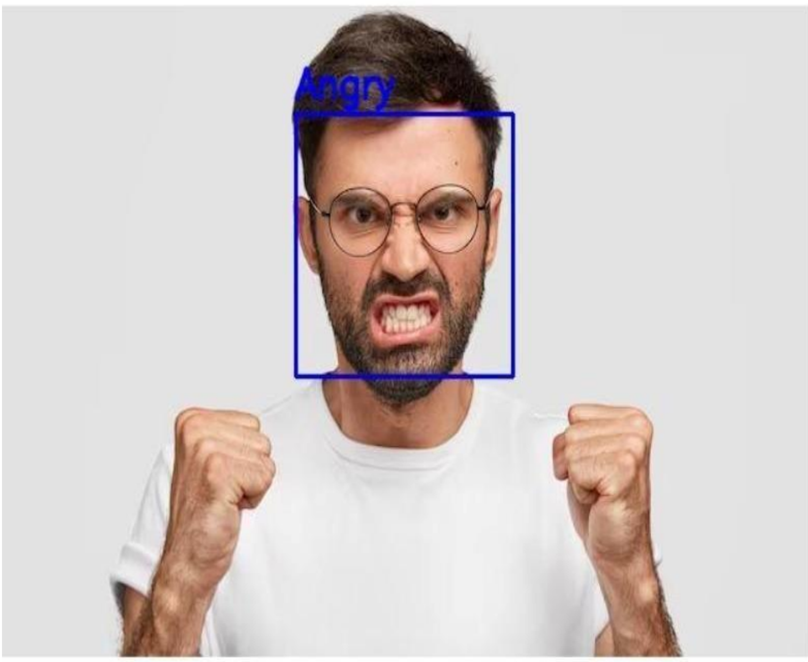
```
E:\facial rec\myVGG.py:31: UserWarning: Update your `Conv2D` call to the Keras 2 API:
`Conv2D(64, (3, 3), activation="relu")`
  model.add(Convolution2D(64, 3, 3, activation='relu'))
E:\facial rec\myVGG.py:33: UserWarning: Update your `Conv2D` call to the Keras 2 API:
`Conv2D(64, (3, 3), activation="relu")`
  model.add(Convolution2D(64, 3, 3, activation='relu'))
E:\facial rec\myVGG.py:37: UserWarning: Update your `Conv2D` call to the Keras 2 API:
`Conv2D(128, (3, 3), activation="relu")`
  model.add(Convolution2D(128, 3, 3, activation='relu'))
E:\facial rec\myVGG.py:39: UserWarning: Update your `Conv2D` call to the Keras 2 API:
`Conv2D(128, (3, 3), activation="relu")`
  model.add(Convolution2D(128, 3, 3, activation='relu'))
E:\facial rec\myVGG.py:41: UserWarning: Update your `Conv2D` call to the Keras 2 API:
`Conv2D(128, (3, 3), activation="relu")`
  model.add(Convolution2D(128, 3, 3, activation='relu'))
Create model successfully
Create model successfully
(28709, 1, 48, 48)
(28709, 6)
Training started
C:\ProgramData\Anaconda3.1\lib\site-packages\keras\models.py:942: UserWarning: The
`nb_epoch` argument in `fit` has been renamed `epochs`.
  warnings.warn('The `nb_epoch` argument in `fit` '
batch  0
```

IPython console History log

```
1/1 [==============================] - 1s 533ms/step
'Detected Emotion: Happy'
```




```
'Detected Emotion: Angry'
```



CONCLUSION

In this case, when the model predicts incorrectly, the correct label is often the second most likely emotion.The facial expression recognition system presented in this research workcontributes a resilient face recognition model based on the mapping of behavioral characteristics with the physiological biometric characteristics. The physiological characteristics of the human face with relevance to various expressions such as happiness, sadness, fear, anger, surprise and disgust are associated with geometrical structures which restored as base matching template for the recognition system.The behavioral aspect of this system relates the attitude behind different expressions as property base. The property bases are alienated as exposed and hidden category in genetic algorithmic genes. The gene training set evaluates the expressional uniqueness of individual faces and provide a resilient expressional recognition model in the field of biometric security.The design of a novel asymmetric cryptosystem based on biometrics having features like hierarchical group security eliminates the use of passwords and smart cards as opposed to earlier cryptosystems. It requires a special hardware support like all other biometrics system. This research work promises a new direction of research in the field of asymmetric biometric cryptosystems which is highly desirable in order to get rid of passwords and smart cards completely. Experimental analysis and study show that the hierarchical security structures are effective in geometric shape identification for physiological traits**.**

FUTURE SCOPE

It is important to note that there is no specific formula to build a neural network that wouldguarantee to work well. Different problems would require different network architecture and a lot of trail and errors to produce desirable validation accuracy. This is the reason whyneural nets are often perceived as "black box algorithms.