CSY3025 Artificial Intelligence Techniques

Assignment 1 – Image Classification using Deep Learning

Dataset Link:

https://drive.google.com/file/d/1kmzULCVD4f3marsNvsocyLBVrzb_YvNo/view

1-INTRODUCTION:

Facial expression is one of the most powerful, natural, and universal signals for human beings to convey their emotional states and intentions regardless of national borders, race, and gender and there were multitudinous related applications such as the health management .
Facial expressions are the facial changes in response to a person's internal emotional states, intentions, or social communications. After that, much progress has been made to build computer application systems to help us understand and use this natural form of human communication.

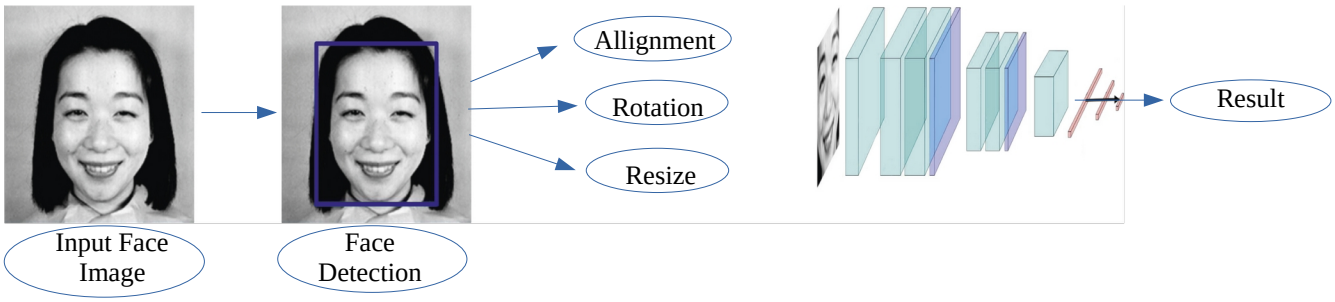

Figure-1 The Process

computer systems that attempt to automatically analyze and recognize facial motions and facial feature changes
from visual information. Sometimes the facial expression analysis has been confused with emotion analysis in the computer vision domain. For emotion analysis, higher level knowledge is required. For example, although facial expressions can convey emotion, they can also express intention, cognitive processes, physical effort,
or other intra- or interpersonal meanings. Interpretation is aided by context, body gesture, voice, individual differences, and cultural factors as well as by facial configuration and timing. Computer facial expression analysis systems need to analyze the facial actions regardless of context, culture, gender, and so on.

2. Problem Analysis and Background Research:

An emotion is a strong automatic inkling that occurs in response to a particular situation or condition. It has three define components: a subjective experience of something, a physiological reaction and a behavioral response. In simple terms, emotion is driven by two opposing factors, reward and punishment. We can define into another way as autonomic response appropriate to a given situation. Human face always concertation to the direct communication, because as a human we directly identify whether someone happy,sad,upset or another state of emotion. The scientific study of emotion expressionism usually traced to Darwin's seminal work. "On the expression of the emotions in man and animal" . He concluded that emotion expression as the visible part of an underlying emotional state, which are evolved and adaptive. Yet, Darwin's view has been disputed and rejected by those who considered facial expressions as exclusively or predominantly social or cultural signals. Also, a number of studies in the early years of the twentieth century came to conclusion that emotion can only be "recognized at chance levels, whereas other studies found good recognition rates. This disparity in findings led Bruner and Toguri in their "Handbook of Social Psychology article to state that" … the evidence for the recognizability of emotional expression is unclear. They concluded that, if anything, emotional expression is unclear, they concluded that, if anything, emotional facial

expression are culturally learned. This view remained basically unchanged until the early 1970's when research by Ekman and colleagues as well as Izar Darwin's idea that at least some basic emotional expression are universal and directly associated with an underlying emotional state.

The word **"learning"** come within its meaning, but it is not piece of cake when we are talking about machines learning. Early nineteen centuries, a remarkable AI pioneer **Alan Turing** published his landmark 1950 paper, that was basically objections of **"Lady Lovelace."** [Computing Machinery and Intelligence][2]. **Turing** was quoting **Ada Lovelace** while thinking whether general-purpose computers able to capable of learning and originality, and he evaluated that they could**.** A nascent **Machine learning** come from that question: could a computer go ahead "what we know how to order it to perform" and learn on its own how to perform a specified task? Could a computer surprise us? Rather than programmers crafting data-processing. could a computer instinctive learn these rules by looking at data?
In normally, programming languages, we have data and rules. Our algorithm has certain conditions, on that conditions we are expecting certain output if output has same as we wanted, our logic is to be truth. Meanwhile we must change our constraints. While in Supervised learning, we have data and expected output (answer) we will get out the rules. These rules can be applied to unfamiliar datasets to produce original answers.

"machine learning as the field of study that gives computers the ability to learn without being explicitly learned".

3-Building The Deep Learning Network:

3.1-Dataset:

First we load the Different reaction images dataset from kaggle to jupyter then we divided the data into seven output classes
then we set pixels to 48 by 48 because computer understand gray scale images in the form of 0's and 1's
then we divided 32 training example utilized in one iteration and 25 total epochs
we used training and validation training data for training of our model and validation data to see how validate our model is
then we Generate batches of tensor image data with real-time data augmentation and normalized the data
The next part was to rescale the images and filtered them and cropped them so that we can prove the model accuracy

3.1.1-Data loading:

Keras models accept three types of inputs:

- **NumPy arrays:** just like Scikit-Learn and many other Python-based libraries. This is a good option if your data fits in memory.
- **TensorFlow Dataset Objects:** This is a high-performance option that is more suitable for datasets that do not fit in memory and that are streamed from disk or from a distributed filesystem.

•**Python generators:** that yield batches of data (such as custom subclasses of the keras.utils.Sequence class)

3.1.2-Data preprocessing with Keras:

Once your data is in the form of string/int/float NumpPy arrays, or a Dataset object (or Python generator) that yields batches of string/int/float tensors, it is time to **preprocess** the data. This can mean:

•**Tokenization** of string data, followed by token indexing.

•Feature **normalization.**

•**Rescaling** the data to small values (in general, input values to a neural network should be close to zero -- typically we expect either data with zero-mean and unit-variance, or data in the[0, 1] range.

•**TextVectorization** holds an index mapping words or tokens to integer indices

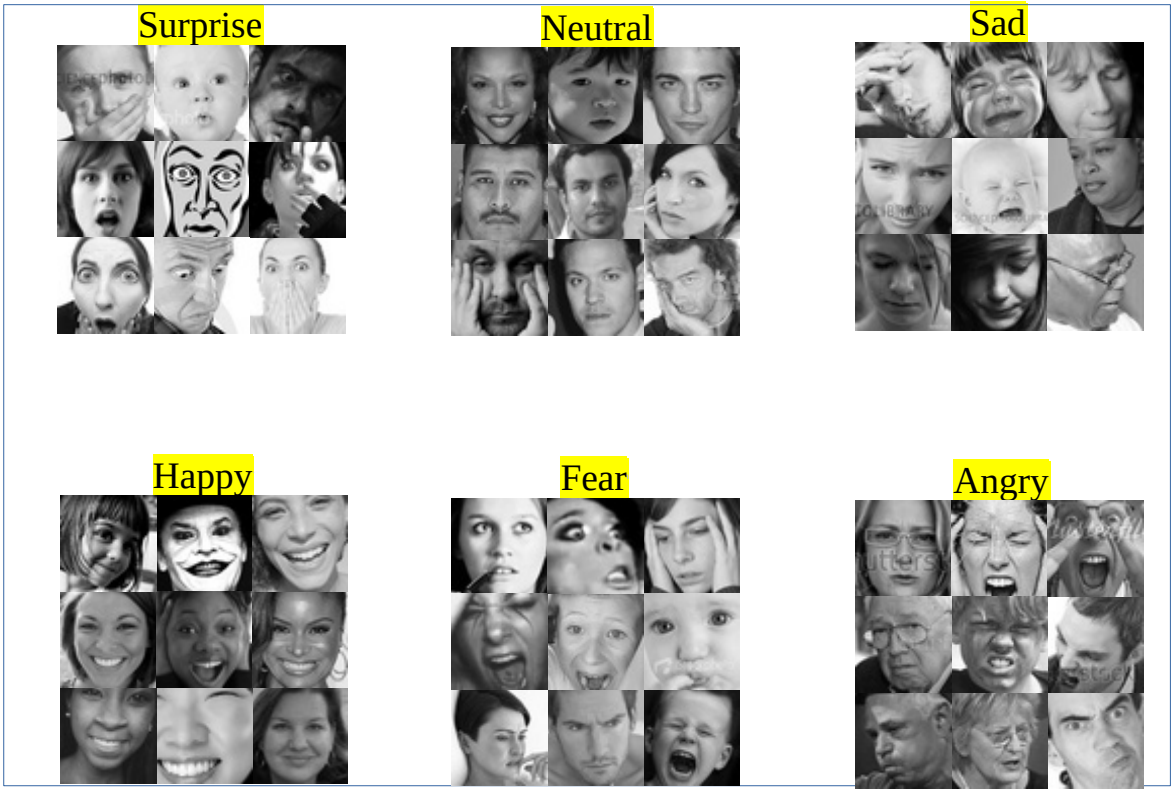

Figure-2 Data Images

Data Generator and Augment

In [7]:
```
print("[INFO] building data generators...\n")

    # Generate batches of tensor image data with real-time data augmentation.
train_datagen = ImageDataGenerator(rescale=1./255,
                rotation_range=30,
                shear_range=0.3,
                zoom_range=0.3,
                width_shift_range=0.4,
                height_shift_range=0.4,
                horizontal_flip=True,
                fill_mode='nearest')


# Normalized images
validation_datagen = ImageDataGenerator(rescale=1./255)

# transforming train images and masks together.
train_generator = train_datagen.flow_from_directory(
                    train_data_dir,
                    color_mode='grayscale',
                    target_size=(img_rows,img_cols),
                    batch_size=batch_size,
                    class_mode='categorical',
                    shuffle=True)
```

Figure-3 Data Normalization

3.2-Network:

Neural networks don't process raw data, like text files, encoded JPEG image files, or CSV files. They process **vectorized** & **standardized** representations.

- •Text files need to be read into string tensors, then split into words. Finally, the words need to be indexed & turned into integer tensors.
- •Images need to be read and decoded into integer tensors, then converted to floating point and normalized to small values (usually between 0 and 1).
- •CSV data needs to be parsed, with numerical features converted to floating point tensors and categorical features indexed and converted to integer tensors. Then each feature typically needs to be normalized to zero-mean and unit-variance.

3.2.1-Convolutional neural networks:

“It is the technique to isolate fetures in image it will train the edges of the image”
After passing 3 by 3 kernal it will learn 3 by 3 matrix .max pool will half the size of convolution layer. This model performs well during testing and training

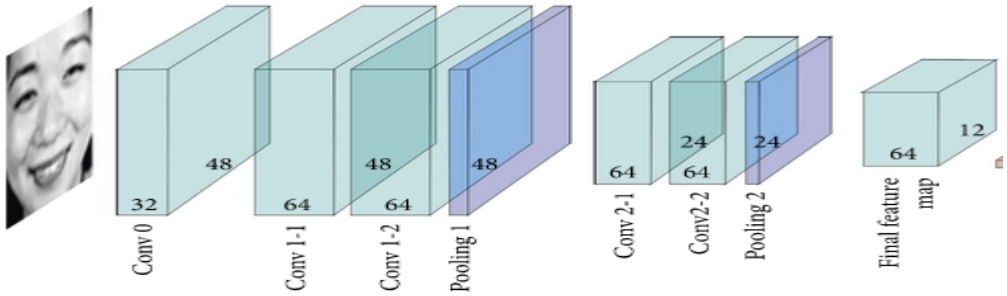

Figure-4 Convolutional Neural Network

3.2.2-Sequential model:

“A Sequential model is appropriate for a plain stack of layers where each layer has exactly one input tensor and one output tensor”.
A Sequential model is not appropriate when your model has multiple inputs or multiple outputs

3.2.3-Activation function:

“ELU. Exponential Linear Unit or its widely known name ELU is a function that tend to converge cost to zero faster and produce more accurate results”. Different to other activation functions, ELU has a extra alpha constant which should be positive number. ELU is very similiar to RELU except negative inputs.

3.2.4-Padding:

Padding is a term relevant to convolutional neural networks as it refers to the amount of pixels added to an image when it is being processed by the kernel of a CNN

3.2.5-loss function:

CategoricalCrossentropy :

“Computes the crossentropy loss between the labels and predictions.Use this crossentropy loss function when there are two or more label classes.”

3.2.6-Optimizer:

Adam:
“Adam optimization is a stochastic gradient descent method that is based on adaptive estimation of first-order and second-order moments has *little memory requirement, invariant to diagonal rescaling of gradients, and is well suited for problems that are large in terms of data/parameters*".

3.3-Training And Evolution:

We used the total of 24176(80%) images for training Each has an equal number of images we used the maximum number of images for training to avoid overfitting . We compile the model with

'categorical_crossentropy' loss function, ‘Adam’ Optimizer 'accuracy' matrics with learning rate of 0.001.
We start the training with realtime with the use of datetime.datetime.now() function After total of 25 epochs loss reduced to 1.0762.We visualized the result with the help matplotlib.

```
Epoch 00022: val_loss did not improve from 1.11445
Epoch 23/25
755/755 [==============================] - 1219s 2s/step - loss: 1.3036 - accuracy: 0.5078 - val_loss: 1.1114 - v
al_accuracy: 0.5756

Epoch 00023: val_loss improved from 1.11445 to 1.11143, saving model to C:\\Users\\Haier\\OneDrive\\Freelancing W
ork\\FaceDetection\Emotion_little_vgg.h5
Epoch 24/25
755/755 [==============================] - 1114s 1s/step - loss: 1.3076 - accuracy: 0.5071 - val_loss: 1.0643 - v
al_accuracy: 0.5917

Epoch 00024: val_loss improved from 1.11143 to 1.06430, saving model to C:\\Users\\Haier\\OneDrive\\Freelancing W
ork\\FaceDetection\Emotion_little_vgg.h5
Epoch 25/25
755/755 [==============================] - 1225s 2s/step - loss: 1.2876 - accuracy: 0.5155 - val_loss: 1.0762 - v
al_accuracy: 0.5951
```

Figure-5 Total Epochs

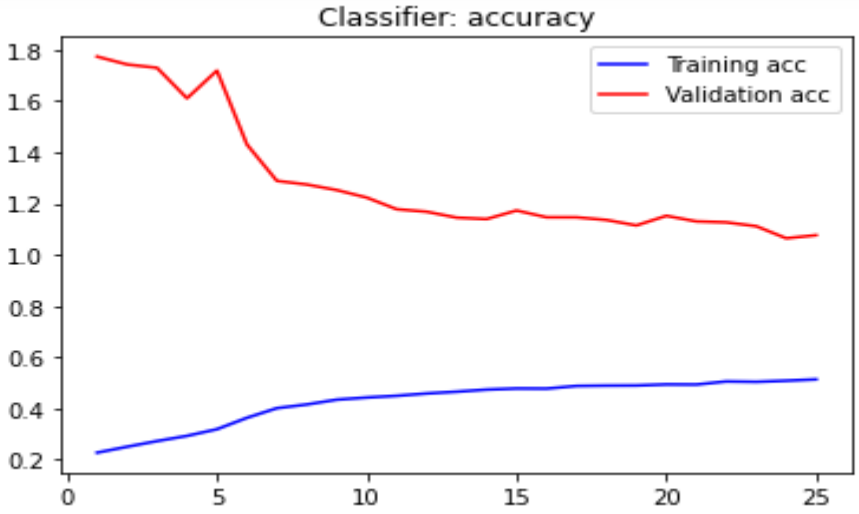

Figure-6 Training And Validation Accuracy

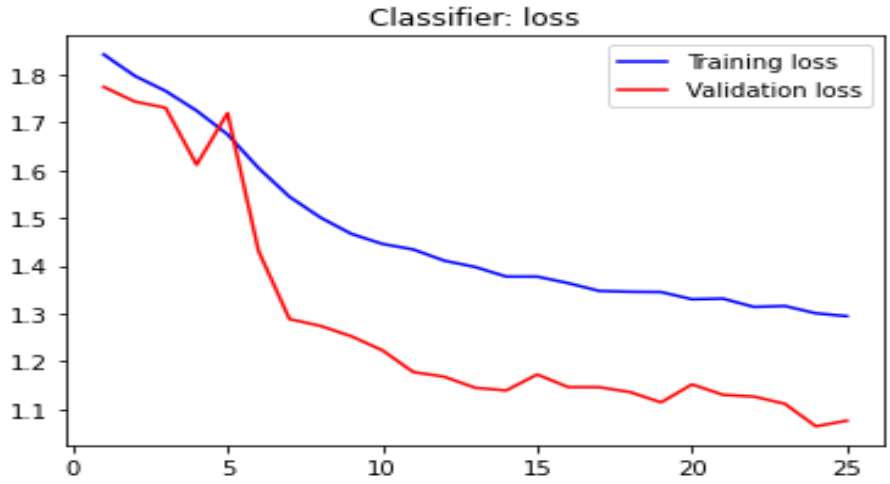

Figure-7 Training And Validation loss

3.4-Testing

(Run the implementation by using camera and take 5,6 snapshots of live camera and paste them there)

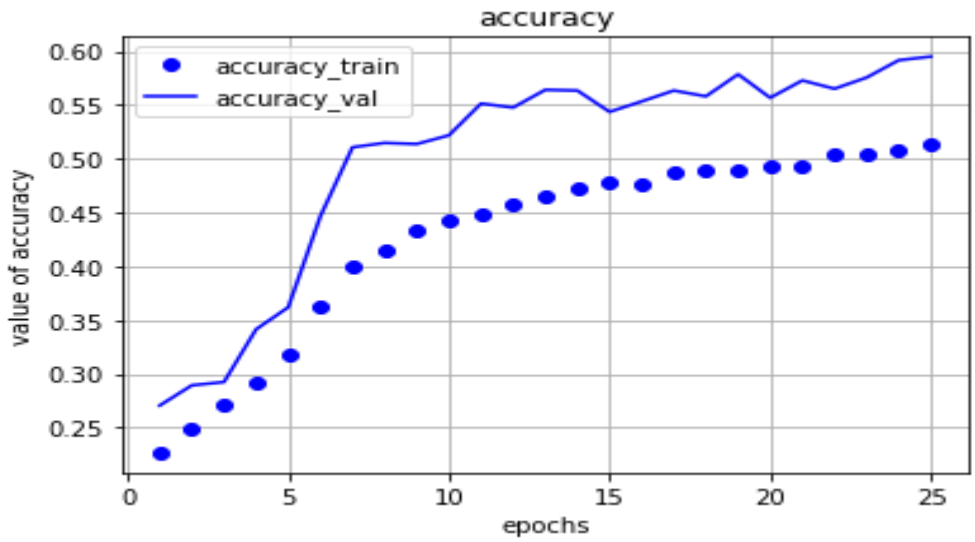

Figure-8 AccuracyTrain And Value

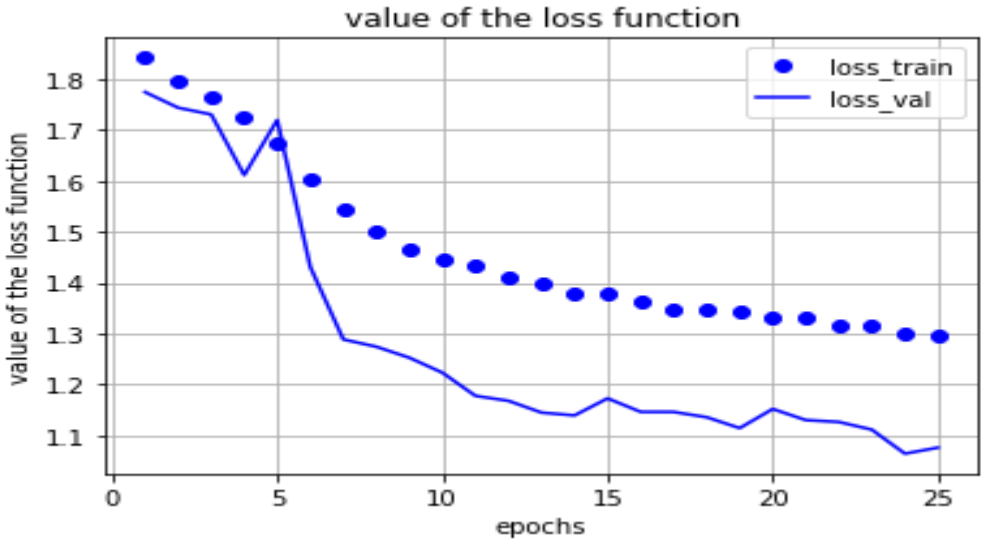

Figure-9 Loss Train And Value

4-Additional Features:

The additional features we use in this model is that we implement our model with the use of real time camera with the help of **CV2(Opencv)** library
"OpenCV-Python is a library of Python bindings designed to solve computer vision" problems.OpenCV-Python makes use of **Numpy**, which is a highly optimized library for numerical operations with a MATLAB-style syntax. All the OpenCV array structures are converted to and from Numpy arrays. This also makes it easier to integrate with other libraries that use Numpy such as SciPy and Matplotlib.

```
face_classifier = cv2.CascadeClassifier(r'C:\home\ebad\Downloads\haarcascade_frontalface_default.xml')
classifier =load_model(r'\home\ebad\Downloads\Emotion_little_vgg.h5')

class_labels = ['Angry','Happy','Neutral','Sad','Surprise','fear','upset']

cap = cv2.VideoCapture(0)



while True:
    # Grab a single frame of video
    ret, frame = cap.read()
    labels = []
    gray = cv2.cvtColor(frame,cv2.COLOR_BGR2GRAY)
    faces = face_classifier.detectMultiScale(gray,1.3,5)

    for (x,y,w,h) in faces:
        cv2.rectangle(frame,(x,y),(x+w,y+h),(255,0,0),2)
        roi_gray = gray[y:y+h,x:x+w]
        roi_gray = cv2.resize(roi_gray,(48,48),interpolation=cv2.INTER_AREA)
    # rect,face,image = face_detector(frame)


        if np.sum([roi_gray])!=0:
            roi = roi_gray.astype('float')/255.0
            roi = img_to_array(roi)
            roi = np.expand_dims(roi,axis=0)

        # make a prediction on the ROI, then lookup the class

            preds = classifier.predict(roi)[0]
            label=class_labels[preds.argmax()]
            label_position = (x,y)
            cv2.putText(frame,label,label_position,cv2.FONT_HERSHEY_SIMPLEX,2,(0,255,0),3)
        else:
            cv2.putText(frame,'No Face Found',(20,60),cv2.FONT_HERSHEY_SIMPLEX,2,(0,255,0),3)
    cv2.imshow('Emotion Detector',frame)
    if cv2.waitKey(1) & 0xFF == ord('q'):
        break

cap.release()
cv2.destroyAllWindows()
```

Figure-10 Implemention By Using Opencv

5-Discussion And Conclusion:

5.1-Lesson Learned:

In this study, by combining the online courses platforms and a compact deep learning model based on the architecture of CNN, we construct a framework to analyze people emotions according to their facial expressions from the perspective of computer simulation
We Obtained the emotional tags for each face.It has been proved that the model is performing well in practical activities.
It plays a good role in solvinf the problems of people by looking at their faces
It also plays a dpositive role in different departments such as teacher can recognized that the student is happy,sad,angry etc with the use of real time implemention by camera

5.2-Limitation:

Athough Our applications is cross platform which we can run at a time Linux/Unix or window operating system, its not supported with mobile or nano architectural frame like raspberry pi (any version).
computational resources are one of the key factor in Deep-learning, if we're going to train a large data set that contain images, normal pc can't handle these huge computational when it comes to processing, we need some additional decencies mostly machine/deep learning practitioner goes with online services that are available in market. we can set up own GPU process for long term utilization of our model due to online resources utilization are limited

6-Refrences:

- Alan Turning (AI-1956) Research Paper
- Python for deep learning
- Facial expressions IEEE
- Facial Emotions IEEE
- Human Phys
- James Anderson Russell, James A. Russell, José Miguel Fernández-Dols · 1997
- Keras documentation
- opencv documentation