Facial Emotion Recognition using CNN ~ Real Time

Spurthi Shetty
Northeastern University
shetty.sp@husky.neu.edu

Sayali Borse
Northeastern University
borse.s@husky.neu.edu

Abstract-The identification of facial expressions is a fundamental topic in the area of human computer interaction and pattern recognition. The research has gained significant attention in recent years. However, many challenges still exist. This is because an individual might display different expressions at different times for the same mood. Expressions can also be influenced by health**.** Our proposed framework aims to capture unique information related to expressions by implementing a Convolution Neural Network has been proposed. Deep learning is one of the most important features that can be used to recognize the facial emotions of human.

The main task here is to recognize constantly changing facial emotions of a person and segregation them into 1 of the 7 categories namely: happy, sad, neutral, surprised, anger, fear and disgust.

Also ever emotion is denoted a value accordingly:
Emotion [0]: Happy
Emotion [1]: Sad
Emotion [2]: Neutral
Emotion [3]: Surprised
Emotion [4]: Angry
Emotion [5]: Fear
Emotion [6]: Disgust
A six convolution-layered CNN is implemented to reach at an optimum conclusion.

I. INTRODUCTION

Facial expression is a natural way of conveying social information without words between humans. Facial Expression Recognition (FER) is an active research area in psychological study as well as computer-human interaction. A facial expression can be considered as a visible attestation of inner state of mind and hence gives idea about intention, interest, and psychology of that person. As a result of the huge information carried by facial expressions, they play vital role in human interaction with humans as well as machines.

It is easy to classify a person's facial expressions by looking at them through the eyes. It is however difficult to implement the model used by the computer to identify the facial expression with the same accuracy and precision as humans. Facial expression differs across the age groups [2], [3], [4] and [5]. Thus, aging has impact on FER accuracy, which cannot be neglected. One of the major non-verbal ways of communication between humans. Hence, recognizing facial emotions is an integral aspect in human computer interaction also having endless applications.
Technological advances make it possible for machines to identify expressions in a variety of applications, e.g., sociable robotics, interactive games, and assisted rehabilitation. Three main steps are needed to label expressions:
(1) face detection,
(2) feature extraction and
(3) emotion classification.

Automatic facial expression recognition can be used as a vital component in natural interface between human and machine which can be called as conversational interfaces. This enables the provision of services automatically and gives good appreciation from the service user. Smart Devices like computers or robots can recognize the expressions and hence emotional state of human. Use of machines will increase the efficiency of the system significantly.

We also compare our computational approach with human perception of expressions in real time.

II. DATASET

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 facial emotions have been ubiquitous and constant throughout the years and one such dataset which captures the humongous variety and quantity of emotions is the FER2013 dataset.

Figure 1. Example images from the FER2013 dataset, illustrating variations in lighting, age, pose, intensity of expressions and occlusions that occur under realistic conditions. These images depict expressions, namely anger, disgust, fear, happiness, sadness, surprise, neutral.

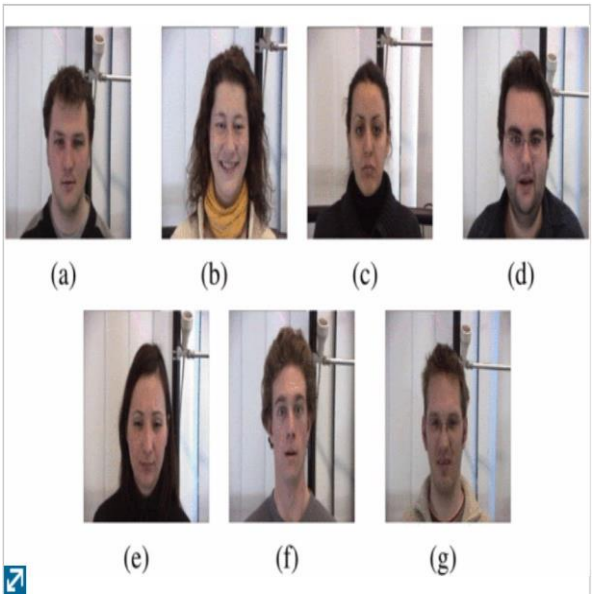


Figure 1.

The data consists of 48x48 pixel grayscale images of faces. The faces have been automatically registered so that the face is 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 emotions in to one of seven categories (0=Angry, 1=Disgust, 2=Fear, 3=Happy, 4=Sad, 5=Surprise, 6=Neutral).

The Training csv contains two columns, "emotion" and "pixels". The "emotion" column contains a numeric code ranging from 0 to 6, inclusive, for the emotion that is present in the image. Test csv contains only the "pixels" column and your task is to predict the emotion column. The training set consists of 28,709 examples.

III. EVALUATION METRIC

Model Evaluation is an integral part of the model development process. It helps to find the best model that represents our data and how well the chosen model will work in the future. Evaluating model performance with the data used for training is not acceptable in data science because it can easily generate overoptimistic and overfitted models. To avoid overfitting, both methods use a test set (not seen by the model) to evaluate model performance.

Evaluation metric that can be done on facial emotion recognition allows users to evaluate the emotion classified from happy to sad to dear to neutral based on Convolution Neural Network that is by deploying Deep Learning model on to it That is the perfect way to find the accuracy for the system and also they are used to measure the performance and working of the model.
Recognition can be given from 0-6 that is meant to be the score for any human emotion specified. With help of these emotions/ prediction values are arrived from test and train data set for further evaluation of the system.

IV. PROPSED FRAME WORK

The model that we used for data classification is Deep Learning→ Convolution Neural Networks that is proposed CNN approach and the CNN methods lies in the imagery input of deep models. Specifically, not only the whole image regions of faces, but also the facial components in the form of local

Our proposed framework consists of the following steps:

Step 1: Given the training facial images and their expression labels, we first localize the face foreground area from the image background. Then, we generate a number of face part patches which contain both the local and global personal identity information.

Step 2: We feed the generated image patches into CNN for deep model training. The same network is shared by all patches for computational convenience.

For testing, we first adopt a face detector to a new patches are employed in our proposed CNN model. The work flow of our proposal in comparison with the conventional approach is illustrated in Fig. 2.

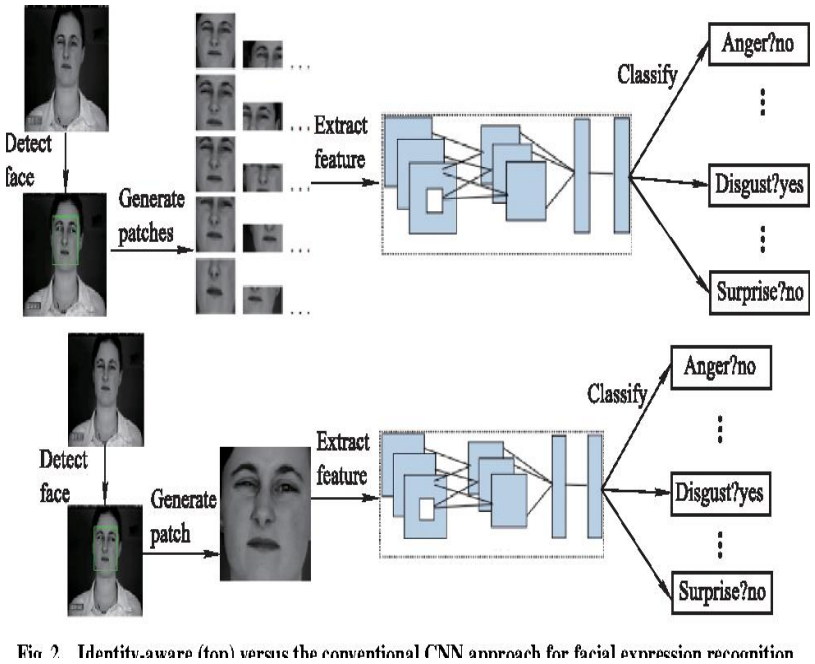


Fig. 2. Identity-aware (top) versus the conventional CNN approach for facial expression recognition

Figure 2.

V. DEEP LEARNING MODEL

CONVOLUTION NEURAL NETWORK

CNNs, like neural networks, are made up of neurons with learnable weights and biases. Each neuron receives several inputs, takes a weighted sum over them, pass it through an activation function and responds with an output. The whole network has a loss function and all the tips and tricks that we developed for neural networks still apply on CNNs.

Each neuron receives some input, performs a dot product and optionally follows it with a non-linearity. The whole network still expresses a single differentiable score function: from the raw image pixels on one end to class scores at the other.

Layers in CNN:

- **INPUT** layer will hold the raw pixel values of the image, of width 32, height 32, and with three color channels Red, Green, Blue.
- **CONV** layer will compute the output of neurons that are connected to local regions in the input, each computing a dot product between their weights and a small region they are connected to in the input volume.
- **RELU** layer will apply an elementwise activation function, such as the max (0, x) thresholding at zero.
- **POOL** layer will perform a down sampling operation along the spatial dimensions (width, height)
- **FC** (i.e. fully-connected) layer will compute the class scores.

For this particular model we implemented a 6-convolution layered neural network for visualizing the internal working from layer-to-layer. Each filter is independently convolved with the image and we end up with 6 feature maps of shape 28*28*1.

CNNs have a couple of concepts called parameter sharing and local connectivity

Parameter sharing is sharing of weights by all neurons in a particular feature map.

Local connectivity is the concept of each neural connected only to a subset of the input image (unlike a neural network where all the neurons are fully connected)

This helps to reduce the number of parameters in the whole system and makes the computation more efficient.

Layer (type)	Output Shape	Param #
conv2d_1 (Conv2D)	(None, 44, 44, 64)	1664
conv2d_2 (Conv2D)	(None, 40, 40, 64)	102464
batch_normalization_1 (Batch	(None, 40, 40, 64)	256
max_pooling2d_1 (MaxPooling2	(None, 20, 20, 64)	0
conv2d_3 (Conv2D)	(None, 18, 18, 128)	73856
conv2d_4 (Conv2D)	(None, 16, 16, 128)	147584
batch_normalization_2 (Batch	(None, 16, 16, 128)	512
max_pooling2d_2 (MaxPooling2	(None, 8, 8, 128)	0
conv2d_5 (Conv2D)	(None, 6, 6, 256)	295168
conv2d_6 (Conv2D)	(None, 4, 4, 256)	590080
batch_normalization_3 (Batch	(None, 4, 4, 256)	1024
max_pooling2d_3 (MaxPooling2	(None, 2, 2, 256)	0
flatten_1 (Flatten)	(None, 1024)	0
dense_1 (Dense)	(None, 1024)	1049600
batch_normalization_4 (Batch	(None, 1024)	4096
activation_1 (Activation)	(None, 1024)	0
dropout_1 (Dropout)	(None, 1024)	0
dense_2 (Dense)	(None, 1024)	1049600
batch_normalization_5 (Batch	(None, 1024)	4096
activation_2 (Activation)	(None, 1024)	0
dropout_2 (Dropout)	(None, 1024)	0
dense_3 (Dense)	(None, 7)	7175

Total params: 3,327,175
Trainable params: 3,322,183
Non-trainable params: 4,992

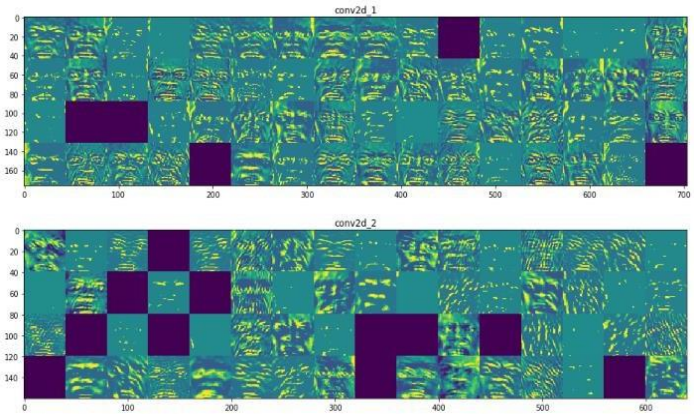
Compiling and running epochs for batch size of 256 to improve the accuracy of the model

```
#Fit the model

history = model.fit_generator(train_generator,
                              validation_data = (x_val, y_val),
                              steps_per_epoch=batch_size,
                              epochs=epochs)
```

Visualizing intermediate activations consists in displaying the feature maps that are output by various convolution and pooling layers in a network, given a certain input. This gives a view into how an input is decomposed unto the different filters learned by the network. Each channel encodes relatively independent features, so the proper way to visualize these feature maps is by independently plotting the contents of every channel, as a 2D image. The 1st layer acts as collection of various edge detectors. At that stage, the activations are still retaining almost all of the information present in the initial picture.

First two layers of our convolution model ~ FER data

.



As we go higher-up, the activations become increasingly abstract and less visually interpretable Higher-up presentations carry increasingly less info about the visual contents of the image, and increasingly more info related to the class of the image

As we go higher-up, the activations become increasingly abstract and less visually interpretable. Higher-up presentations carry increasingly less info about the visual contents of the image, and increasingly more info related to the class of the image

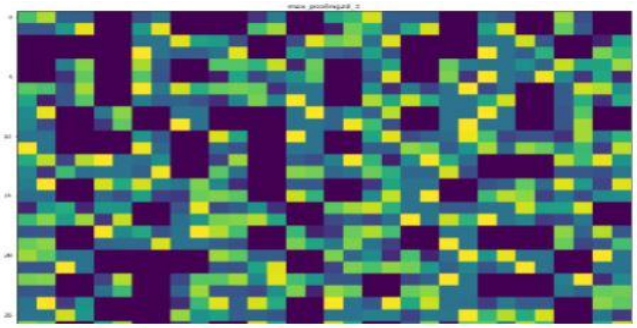


Figure 4 Showing the 12th layer in our CNN model

The activations of layers higher-up carry less and less information about the specific input being seen, and more and more information about the target.

In the initial layers emotions or face structure is easily detectable as we try or move high in layers it starts to distort and it is not visibly identifiable.

A deep neural network is a neural network with a certain level of complexity, a neural network with more than two

layers. Deep neural networks use sophisticated mathematical modeling to process data in complex ways

In this paper we propose an implement a general convolutional neural network (CNN) building framework for designing real-time CNNs. We validate our models by creating a real-time vision system which accomplishes the tasks of face detection, gender classification and emotion classification

The conclusions drawn in this paper are thus relevant for sequence based facial emotion recognition as well.

VI. DATA ANALYSIS

Using the CNN model that we have deployed for FER data set

We trained the model on 200 epochs and we were getting training accuracy of 86%. But the model was overfitting as the test accuracy was around 40%.

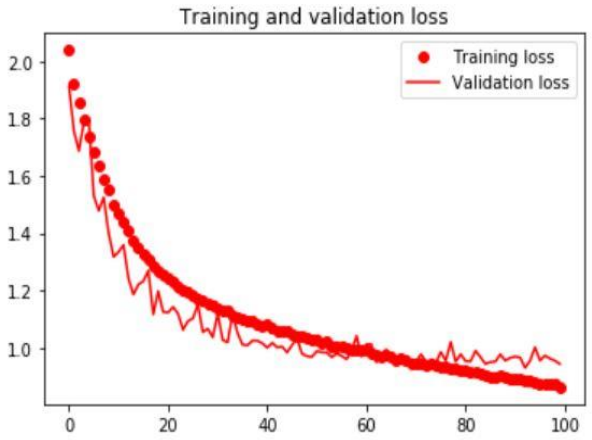
So, in order to decrease the overfitting, we decrease the number of epochs to 100. This solved the overfitting problem. To increase the testing accuracy, we added batch normalization in every convolution layer. Batch normalization reduces the amount by what the hidden unit values shift around. It normalizes the output of a previous activation layer by subtracting the batch mean and dividing by the batch standard deviation. Adding batch normalization further improved the test accuracy to up to 65%. We fine-tuned hyperparameters of the model and finally got training and validation accuracy of 67%.

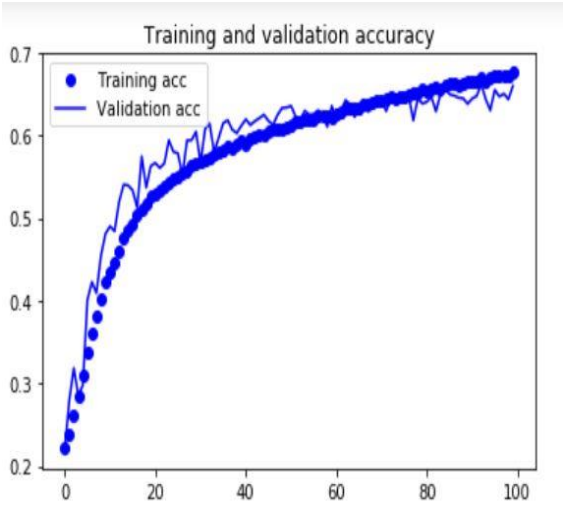
Also we trained the model and found the max accuracy that we could have achieved for our model and displayed some regression graphs for training and validation data

Below are the snapshots of our training and validation score

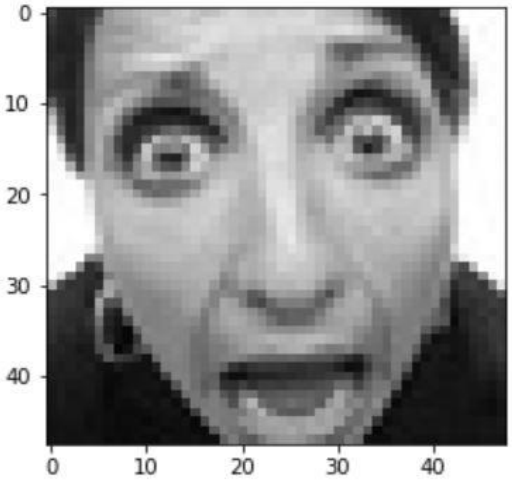
```
#Graph for training and validation accuracy

acc = history.history['acc']
val_acc = history.history['val_acc']
loss = history.history['loss']
val_loss = history.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()

#Graph for training and validation loss

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

plt.show()
```

Below are graphs for the above code





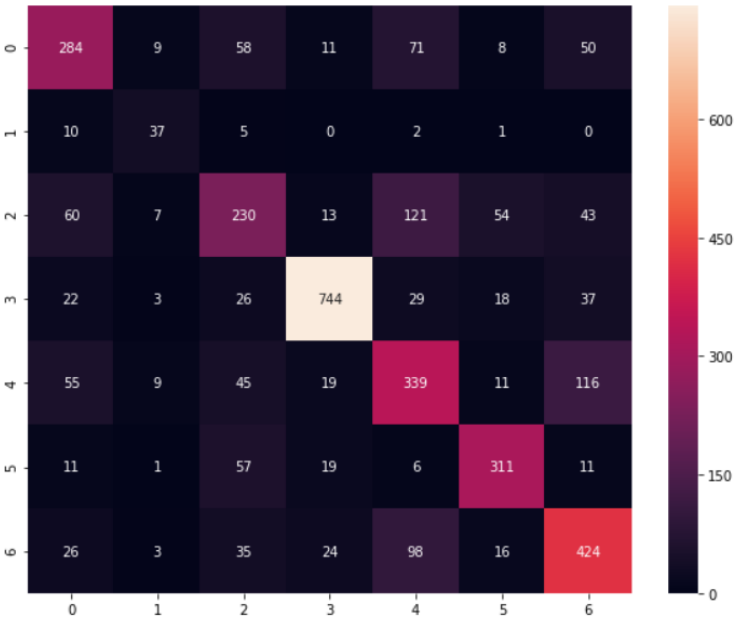
Also we did a few analysis on predicting the emotion



Out of the many test we conducted it seems that model is able to predict well for expressions such as happy, neutral, angry, surprise. The model however struggles to predict disgust. It is because the training set didn't have enough image for the model to learn.

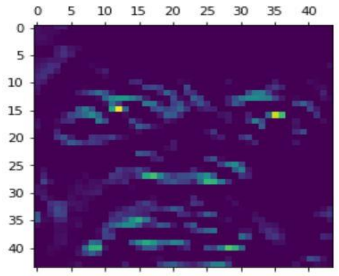
We can see from the confusion matrix in the model performance.

Figure Confusion Matrix for prediction of various expressions [Index: 0 – Angry 1 – Disgust 2 – Fear 3 – Happy 4 – Sad 5 – Surprise 6 - Neutral]

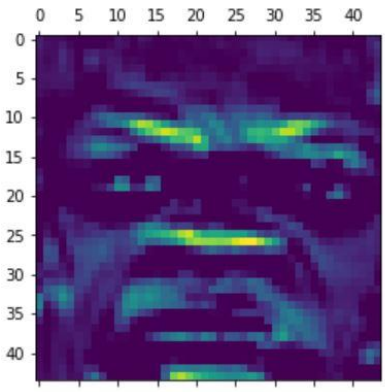


First and 30 th activation layer images of emotion

```
import matplotlib.pyplot as plt

plt.matshow(first_layer_activation[0, :, :, 1], cmap='viridis')
plt.show()
```



Displaying the 30th channel:

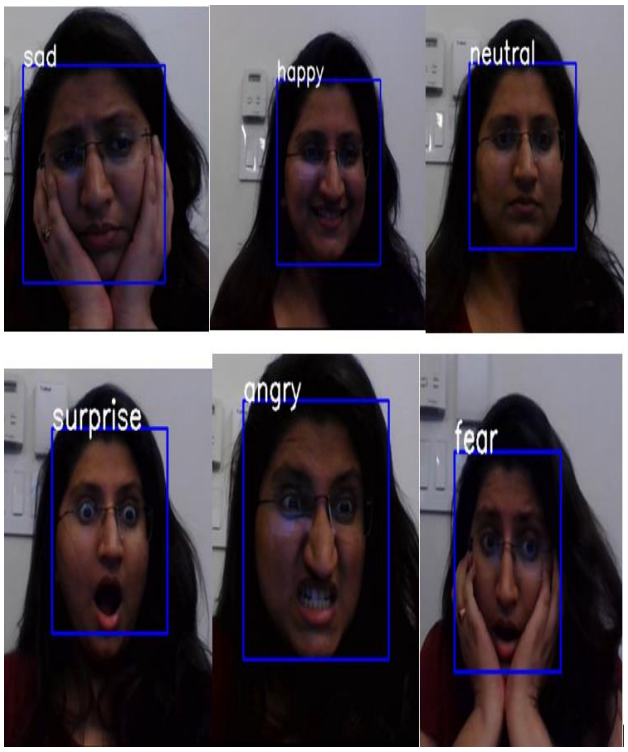


VII. REAL TIME IMPLEMENTATION

Results of the real-time emotion classification task in unseen faces can be observed in Figure 5. Our complete realtime pipeline including: face detection, emotion classification have been fully integrated.

A comparison of the learned features between several emotions and both of our proposed models can be observed in Figure 8. The white areas in figure 8b correspond to the pixel values that activate a selected neuron in our last convolution layer. The selected neuron was always selected in accordance to the highest activation. We can observe that the CNN learned to get activated by considering features such as the frown, the teeth, the eyebrows and the widening of one's eyes, and that each feature remains constant within the same class. These results reassure that the CNN learned to interpret understandable human-like features, that provide generalizable elements

These interpretable results have helped us understand several common misclassifications such as persons with glasses being classified as "angry". This happens since the label "angry" is highly activated when it believes a person is frowning and frowning features get confused with darker glass frames.



VIII. CONCLUSION

We have proposed and tested a general building designs for creating real-time CNNs. Our proposed architectures have been systematically built in order to reduce the number of parameters.

Specifically, we have developed a vision system that performs face detection, and emotion classification in a single integrated module. We have achieved human-level performance in our classification's tasks using a single CNN that leverages modern architecture constructs.

Finally, we presented a visualization of the learned features in the CNN using the visualization. This visualization technique is able to show us the high-level features learned by our models and discuss their interpretability.

Also, we successfully implemented a real time facial expression recognition system.

IX. FUTURE WORK

In the future, we would augment the current analysis to include moving motion in real time and user evaluations as features in the prediction model. We would also explore further hybrid approaches and evaluate their performances. Also other approaches would be to detect subtle or minor emotion change within seconds.

X. REFERENCES

[1]https://medium.com/@RaghavPrabhu/understanding-of-convolutional-neural-network-cnn-deep-learning-99760835f148

[2]https://sefiks.com/2018/01/10/real-time-facial-expression-recognition-on-streaming-data/

[3]https://towardsdatascience.com/tagged/opencv

[4] https://sefiks.com/2018/01/10/real-time-facial-expression-recognition-on-streaming-data/

[5] https://github.com/topics/facial-expression-recognition