# Automatic Human Emotion Recognition Model Using Convolution Neural Network

P.Subashini[1], Medharametla Alekya[2], K.Yasaswi[3]
[1,2,3] *SRM Institute of science and Technology*

***Abstract*- Facial expression recognition systems have attracted abundant analysis interest among the sphere of AI. Many established facial expression recognition (FER) systems apply standard machine learning to pull out the image features. This article is designed for developing an automated framework for emotion detection. This project builds upon recent research to arrange images of human faces into discrete emotion categories using convolutional neural networks (CNN). These methods were trained on the posed-emotion dataset (JAFFE). In our framework we have taken a frame from live streaming video and processed it using cascade image detector and convolution neural network.**

**Index terms- Face expression, CNN, Emotion detection, Cascade image detector.**

## INTRODUCTION

The main aim of this article is to detect the emotion of a person through his or her facial expressions. Now a day it has become more interesting in the researchers mind that to explore the interaction between human and the computer. Based on the researchers' the facial expression recognition is classified into six basic emotions that are sadness, disgust, happiness, anger, fear, and surprise. It is very challenging task to establish a effective automatic emotion recognition framework.

This recognition of emotion is useful to make a better interaction between the system and the human. The human emotion recognition have many applications in heterogeneous field. The applications are mainly based on the man and machine interaction, patient surveilling, inspecting for antisocial motives etc.

In Section-II some of the related work to the emotion recognition has been mentioned. The section-III describes the data set. Section-IV mostly talked about approach, where at first frame extraction from a live streaming video and a well-known face detection through Convolution neural network and applied K-mean with a little modification for clustering, along with a pictorial representation of flow chart and output of each individual step. Section-V shows the result, further work and the conclusion.

## II. RELATED WORK

Generally, the emotion recognition is two steps processes which are feature extraction and Classification. Feature Extraction majorly helps us to find out the emotion. Classification finally classifies the emotion among the basic six emotions present.


*Zhihong Zeng, Member, IEEE Computer Society, Maja Pantic, Senior Member, IEEE, Glenn I. Roisman, and Thomas S. Huang, Fellow, IEEE (2009)* A Survey of Affect Recognition Methods: Audio, Visual, and Spontaneous Expressions: Automated analysis of human affective behavior has attracted increasing attention from researchers in psychology, computer science, linguistics, neuroscience, and related disciplines. However, the existing methods typically handle only deliberately displayed and exaggerated expressions of prototypical emotions, despite the fact that deliberate behavior differs in visual appearance, audio profile,

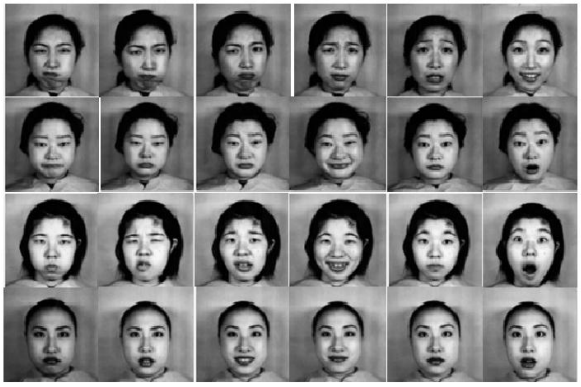
and timing from spontaneously occurring behavior. To address this problem, efforts to develop algorithms that can process naturally occurring human affective behavior have recently emerged. Moreover, an increasing number of efforts are reported toward multimodal fusion for human affect analysis, including audio visual fusion, linguistic and paralinguistic fusion, and multicue visual fusion based on facial expressions, head movements, and body gestures. This paper introduces and surveys these recent advances. We first discuss human emotion perception from a psychological perspective. Next, we examine available approaches for solving the problem of machine understanding of human affective behavior and discuss important issues like the collection and availability of training and test data. We finally outline some of the scientific and engineering challenges to advancing human affect sensing technology. This article is greatly influenced by all of this contribution in the field of emotion recognition.

EXISTING SYSTEM

The human faces play a vital role for automatic recognition of emotion in the region of identification of human emotion. Most recorded facial emotion recognition systems, are not fully considered. So they are not powerful enough for real world recognition of a face. It has been designed an automated framework for emotion detection using the facial expression. It's has been said that a frame from live streaming and processed it using Gabor feature extraction and neural network. To detect the facial gestures it had used a method called artificial neural network.

III.DATASET

In this article we use JAFFE (Japanese Female Facial Expressions) database to train our network. . This database contains 180 gray images of 6 facial expressions posed by 10 Japanese female models. All these 180 images are static with 256 X 256 pixles. After doing lot of experiments on Neural Networks one can understand, the data which is used to train the system plays an important role. For best classification accuracy the neural network architecture need to be trained by large amount of data.



PROPOSED SYSTEM

IV.IMPLEMENTATION

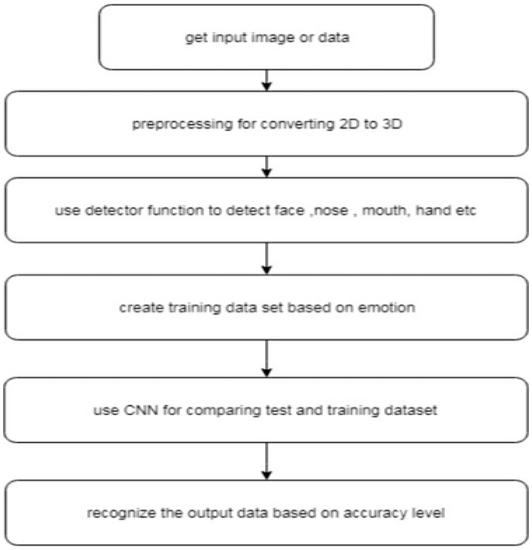
A) Extraction of frames:

From a live streaming video, the frames are been extracted. Frames are been detected that are having faces.

B) Face detection:

Before detecting the 3D image is converted into 2D image by using grayscale converter. For detection of face from the frames, a well-known technique Cascade image detector and convolution neural network are applied. These frames are sent for further processing where the system is already trained for emotion recognition.

These two techniques plays an important role in recognition of emotion from the frames. The flow chart describes the complete emotion detection process.



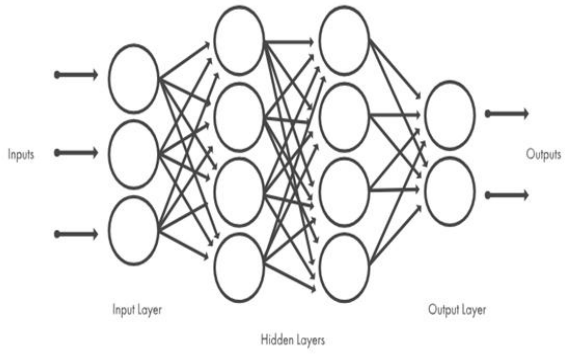

Flow chart for emotion detection

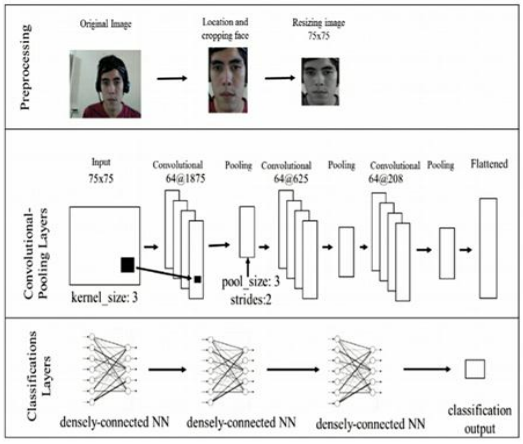
i) Cascade image detector:

For detecting the emotion the facial expressions plays vital role. The frame may contain other parts of the body along with the face. For detecting the face alone from the given frame we are using this Detecor.

ii) Convolution neural network:

CNNs are regularized versions of multilayer perceptrons. A convolutional neural network consists of an input and an output layer, as well as multiple hidden layers. The hidden layers of a CNN typically consist of a series of convolutional layers. The activation function is commonly a RELU layer, and is subsequently followed by additional convolutions such as pooling layers, fully connected layers and normalization layers, referred to as hidden layers because their inputs and outputs are masked by the activation function and final convolution. The final convolution, in turn, often involves back propagation in order to more accurately weight the end product.



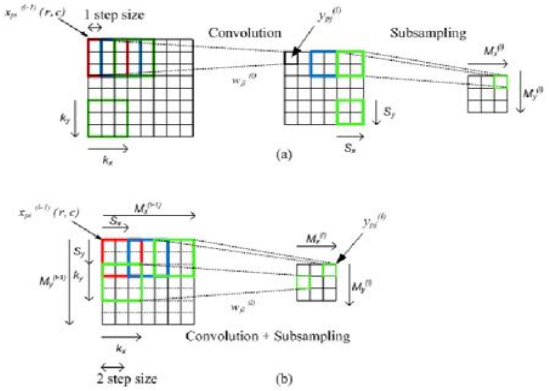

1. Convolution

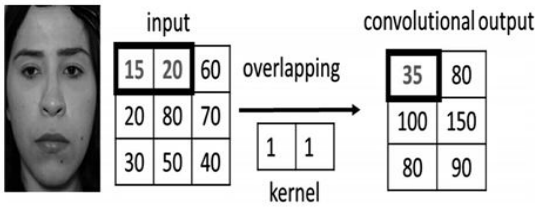
A convolution is a combined integration of two functions that shows you how one function modifies the other.

$$(f * g)(t) \stackrel{\text{def}}{=} \int_{-\infty}^{\infty} f(\tau) g(t-\tau)\, d\tau = \int_{-\infty}^{\infty} f(t-\tau) g(\tau)\, d\tau.$$

There are three important items to mention in this process: the input image, the feature detector, and the feature map. The input image is the image being detected. The feature detector is a matrix, usually 3x3 (it could also be 7x7). A feature detector is also referred to as a kernel or a filter.

Intuitively, the matrix representation of the input image is multiplied element-wise with the feature detector to produce a feature map, also known as a convolved feature or an activation map. The aim of this step is to reduce the size of the image and make processing faster and easier. Some of the features of the image are lost in this step.




However, the main features of the image that are important in image detection are retained. These features are the ones that are unique to identifying that specific object. For example each animal has unique features that enable us to identify it. The way we prevent loss of image information is by having many feature lighting in the pictures and different angles of the images. maps. Each feature map detects the location of certain features in the image.

2. Apply the ReLu (Rectified Linear Unit)

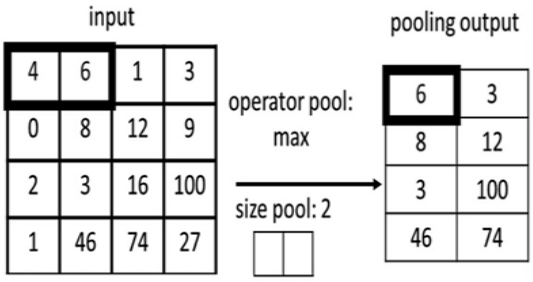
In this step we apply the rectifier function to increase non-linearity in the CNN. Images are made of

different objects that are not linear to each other. Without applying this function the image classification will be treated as a linear problem while it is actually a non-linear one.

3. Pooling

Spatial invariance is a concept where the location of an object in an image doesn't affect the ability of the neural network to detect its specific features. Pooling enables the CNN to detect features in various images irrespective of the difference in lighting in the pictures and different angles of the images.

There are different types of pooling, for example, max pooling and min pooling. Max pooling works by placing a matrix of 2x2 on the feature map and picking the largest value in that box. The 2x2 matrix is moved from left to right through the entire feature map picking the largest value in each pass.




These values then form a new matrix called a pooled feature map. Max pooling works to preserve the main features while also reducing the size of the image. This helps reduce over fitting, which would occur if the CNN is given too much information, especially if that information is not relevant in classifying the image.

4. Flattening

Once the pooled featured map is obtained, the next step is to flatten it. Flattening involves transforming the entire pooled feature map matrix into a single column which is then fed to the neural network for processing.

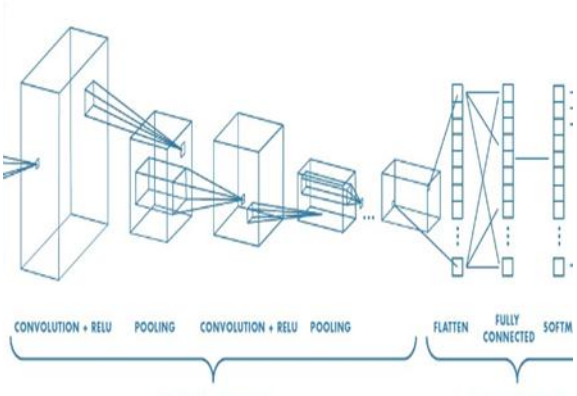
5. Full connection

After flattening, the flattened feature map is passed through a neural network. This step is made up of the input layer, the fully connected layer, and the output layer. The fully connected layer is similar to the hidden layer in ANNs but in this case it's fully connected. The output layer is where we get the predicted classes. The information is passed through the network and the error of prediction is calculated. The error is then back propagated through the system to improve the prediction.

The final figures produced by the neural network don't usually add up to one. However, it is important that these figures are brought down to numbers between zero and one, which represent the probability of each class. This is the role of the Softmax function.

6. softmax

In mathematics, the softmax function, also known as softargmax or normalized exponential function,198 is a function that takes as input a vector of K real numbers, and normalizes it into a probability distribution consisting of K probabilities proportional to the exponentials of the input numbers. That is, prior to applying softmax, some vector components could be negative, or greater than one; and might not sum to 1; but after applying softmax, each component will be in the interval and the components will add up to 1, so that they can be interpreted as probabilities. Furthermore, the larger input components will correspond to larger probabilities. Softmax is often used in neural networks, to map the non-normalized output of a network to a probability distribution over predicted output classes.




In CNN face detection is done completely when the given image is passed through each layer. These layers are divided into two parts as Feature learning and Classification. The algorithm followed by Feature learning is Forward Propagation.

1. Forward propagation:

Output of neuron of row k, column y in the convolution layer and

$$O_{x,y}^{(l,k)} = \tanh(\sum_{t=0}^{f-1}\sum_{r=0}^{k_h}\sum_{c=0}^{k_w} W_{(r,c)}^{(k,t)} O_{(x+r,x+c)}^{(l-1,t)} + Bias^{(l,k)})$$

among them, f is the number of convolution cores in a feature pattern. output of neuron of row x , column y in the l th sub sample layer and k th feature pattern

$$O_{x,y}^{(l,k)} = \tanh(W^{(k)} \sum_{r=0}^{S_h}\sum_{c=0}^{S_w} O_{(x\times S_h+r,y\times S_w+c)}^{(l-1,k)} + Bias^{(l,k)})$$

the output of the j th neuron in l th hide layer H :

$$O_{(l,j)} = \tanh(\sum_{k=0}^{s-1}\sum_{x=0}^{S_h}\sum_{y=0}^{S_w} W_{(x,y)}^{(j,k)} O_{(x,y)}^{(l-1,k)} + Bias^{(l,j)})$$

among them, sis the number of feature patterns in sample layer. output of the i th neuron l th output layer F

$$O_{(l,i)} = \tanh(\sum_{j=0}^{H} O_{(l-1,j)} W_{(i,j)}^{l} + Bias^{(l,i)})$$

2. Back Propagation:

output deviation of the k th neuron in output layer O :

$$d(O_k^O) = y_k - t_k$$

input deviation of the k th neuron in output layer:

$$d(I_k^O) = (y_k - t_k)\varphi'(v_k) = \varphi'(v_k) d(O_k^o)$$

weight and bias variation of k th neuron in output O :

$$\Delta W_{k,x}^{O} = d(I_k^O) y_{k,x}$$

$$\Delta Bias_k^{O} = d(I_k^O)$$

output bias of k th neuron in hide layer

$$d(O_k^H) = \sum_{i=0}^{i<17} d(I_i^O) W_{i,k}$$

input bias of k th neuron in hide layer H :

$$d(I_k^H) = \varphi'(v_k) d(O_k^H)$$

weight and bias variation in row x , column y in the m th feature pattern ,a former layer in front of k neurons in h

ide layer H

$$\Delta W_{m,x,y}^{H,k} = d(I_k^H) y_{x,y}^{m}$$

$$\Delta Bias_k^{H} = d(I_k^H)$$

output bias of row x , column y in m th feature pattern ,sub sample layer S

$$d(O_{x,y}^{S,m}) = \sum_{k}^{170} d(I_{m,x,y}^{H}) W_{m,x,y}^{H,k}$$

input bias of row x , column y in m th feature pattern ,sub sample layer S

$$d(I_{x,y}^{S,m}) = \varphi'(v_k) d(O_{x,y}^{S,m})$$

weight and bias variation of row x , column y in m th feature pattern ,sub sample layer S

$$\Delta W^{S,m} = \sum_{x=0}^{fh}\sum_{y=0}^{fw} d(I_{\lfloor x/2 \rfloor,\lfloor y/2 \rfloor}^{S,m}) O_{x,y}^{C,m}$$

among them, C represents convolution layer.

$$\Delta Bias^{S,m} = \sum_{x=0}^{fh}\sum_{y=0}^{fw} d(O_{x,y}^{S,m})$$

output bias of row x ,column y in k th feature patter ,convolution layer C

$$d(O_{x,y}^{C,k}) = d(I_{\lfloor x/2 \rfloor,\lfloor y/2 \rfloor}^{S,k}) W^{k}$$

output bias of row x ,column y in k th feature patter ,convolution layer C

$$d(I_{x,y}^{C,k}) = \varphi'(v_k) d(O_{x,y}^{C,k})$$

weight variation of row r ,column c in m th convolution core, corresponding to k th feature pattern in l th layer ,convolution C

$$\Delta W_{r,c}^{k,m} = \sum_{x=0}^{fh}\sum_{y=0}^{fw} d(I_{x,y}^{C,k}) O_{x+r,y+c}^{l-1,m}$$

total bias variation of the convolution core

$$\Delta Bias^{C,k} = \sum_{x=0}^{fh}\sum_{y=0}^{fw} d(I_{x,y}^{C,k})$$