SCHOOL OF INFORMATION TECHNOLOGY & ENGINEERING

Review -3

Project Report On Facial Expression Detection And Recognition

Submitted For Course : Artificial Intelligence (ITE2010)

Submitted By:
AKASH KUMAR(17BIT0055)
SLOT – A2+TA2

Under Guidance Of:
Prof. AJIT KUMAR SANTRA

INDEX

1.0 ABSTRACT

In order to detect a facial expression the system should analyze various variability of human faces like color, posture, expression, orientation, lighting etc. Detecting facial features is a pre-requisite to facial emotion recognition.

This is achieved by observing the parts if the face, like eyes, lips movement etc. These are then classified and compared to trained sets of data. In this research, a human facial expression recognition system will be modelled using eigenface approach. Fisher Faces calculation can be utilized for decreasing the high dimensionality of the eigenspace.

2.0 OBJECTIVE

- Detect & recognize facial changes ,patterns ,counters, expression , motions and mood of person
- Expressions help in understanding the overall mood
- Facial expressions are important in facilitating human communication and interactions

3.0 INTRODUCTION

Importance of Emotion Recognition :

- Human face detection plays an important role in applications such as video surveillance, human computer interface, face recognition, and face image database management.
- Facial expressions are important cues for non verbal communication among human beings. This is only possible because humans are able to recognize emotions quite accurately and efficiently. An automatic facial emotion recognition has many commercial uses and can be used as a biological model for cognitive processing and analyzing of human brain.
- Collectively they can enhance their applications like monitoring and surveillance analysis, biomedical image, smart rooms intelligent robots, human computer interfaces and drivers alertness system and can play a vital role in the field of security and crime investigations.

3.1 Literature Review

Computer Aided Diagnosis is gaining significant importance in the day-today life. Specifically, the usage of the computer aided systems for computational biomedical applications has been explored to a higher extent. Medical image analysis is an important biomedical application which is highly computational in nature and requires the aid of the automated systems. These image analysis techniques are often used to detect the abnormalities in the human bodies through scan images.

[1] **IJCSI International Journal of ComputerScience Issues, Vol. 9, Issue 6, No 1, November 2012. Authors: Faizan Ahmad , AaimaNajam and Zeeshan Ahmed.**

A neural network-based algorithm to detect upright, frontal views of faces In grayscale images. It adopts skin model and Adaboost face detection algorithm based on Haar feature to detect face. It is an innovative method that combines a feature- based approach with a holistic one for three-dimensional (3D) face

[2]

Facial Expression Recognition Using Eigenspaces,Vol. 10,2013 Authors: DebasmitaChakrabarti a, DebtanuDuttab.

Describes a generalized view of Face Recognition Technology in latest times. Compares with human perception and discusses problems like illumination. Compares ICA and PCA Algorithms on different distance on facial detection and facial expression. CNN model with the combination of raw pixel data and Histogram of Oriented Gradients (HOG) features.

[3]	**A Human Facial Expression Recognition Model based on Eigen Face Approach,March 2015. Authors : Anurag Dea , AshimSahaa,Dr. M.C Pal.**

For this purpose watershed method is used in combination with edge detection operation. It is a color based detection algorithm using color images in HSV color space. The RGB image is converted to HSV color image by which the image is separated in three regions hue, saturation, and intensity. After contrast enhancement watershed algorithm is applied to the image for each region. Canny edge detector is applied to the output image. After combining the three images final brain tumor segmented image is obtained. The algorithm has been applied on twenty. The developed algorithm has given promising results.

[4] **Convolutional Neural Networks for Facial Expression Recognition, Authors : ShimaAlizadeh, Azar Fazel.**

A self-organizing neural network based automated system for glioma detection is implemented. The main disadvantages of this system are the low classification accuracy and the lack of multiclass analysis. RBF kernel based SVM for brain tumor detection is used. The results of SVM are compared with AdaBoost, a machine learning algorithm. Experimental results illustrated the superior nature of SVM over the other classifiers. Image classification based on fuzzy approach using the pattern discovery algorithm is demonstrated.

[5]	**Face recognition: A literature survey. ACM Computing Surveys, 35(4):399– 458, 2003. Authors: W. Zhao, R. Chellappa, P.J. Phillips, and A. Rosenfeld.** Experiments are conducted on various real-world datasets and the results concluded that the proposed algorithm yield good results when compared with the other classifiers. A hybrid approach for pattern classification is reported. The combination of SVM and fuzzy rules is experimented in this work. The results revealed that the proposed hybrid approach is accurate, fast and robust.

[6]

Recognizing Faces with PCA and ICA, Computer Vision and Image Understanding Authors: B. Draper, K. Baek, M.S. Bartlett, and J.R. Beveridge

Today’s modern medical imaging research faces the challenge of detecting facial expression through open CV. To detect the facial emotion of any person of nay person sitting any where. Due to this anybody image can be detected .For study of facial expression and segmentation the images is very useful in recent years.Different techniques are used to detect and segment of face

[7]

Intra-personal kernel space for face recognition Authors: S. Zhou, R. Chellappa, B. Moghaddam Derives the principal subspace from the intra-personal kernel space by developing a probabilistic analysis of kernel principal components for face recognition.

For study of facial expression and segmentation the images is very useful in recent years.Different techniques are used to detect and segment of face These techniques are SOM clustering, K-mean clustering, Fuzzy C-mean technique, curve let transform.

[8] **Face detection and recognition using LBPH Authors:ChakkaMoun ica, Venugopal P2 Discusses about LBPH algorithm and how it is used to minimize false positive detections, which is beneficial for law enforcement and security purposes.**

Describes a generalized view of Face Recognition Technology in latest times. Compares with human perception and discusses problems like illumination. Compares ICA and PCA Algorithms on different distance on facial detection and facial expression. CNN model with the combination of raw pixel data and Histogram of Oriented Gradients (HOG) features.

[9] **Research on Face Detection Algorithm in Instant Message Robot**

Authors: Yufeng Li, Haijuan Zhang, Yanan Zhang Shenyang Aerospace University

For this purpose watershed method is used in combination with edge detection operation. It is a color based detection algorithm using color images in HSV color space. The RGB image is converted to HSV color image by which the image is separated in three regions hue, saturation, and intensity. After contrast enhancement watershed algorithm is applied to the image for each region. Canny edge detector is applied to the output image. After combining the three images final brain tumor segmented image is obtained. The algorithm has been applied on twenty. The developed algorithm has given promising results.

[10] Neural Network- Based Face Detection Henry A. Rowley, Student Member, IEEE, ShumeetBaluja, and Takeo Kanade, Fellow, IEEE

Clustering approach is widely used in biomedical applications particularly for detection in abnormal magnetic resonance images. The effectiveness of the LSB algorithm in terms of computational rate is improved by modifying the cluster center and membership value updating criterion.

[11] 3D face detection using curvature analysis Authors: Alessandro Colombo, Claudio Cusano, Raimondo Schettini

Describes a generalized view of Face Recognition Technology in latest times. Compares with human perception and discusses problems like illumination. Compares ICA and PCA Algorithms on different distance on facial detection and facial expression. CNN model with the combination of raw pixel data and Histogram of Oriented Gradients (HOG) features.

[12] Convolutional Face Finder: A Neural Architecture for Fast and Robust Face Detection Authors: Christophe Garcia and ManolisDelakis

For study of facial expression and segmentation the images is very useful in recent years.Different techniques are used to detect and segment of face These techniques are SOM clustering, K-mean clustering, Fuzzy C-mean technique, curve let transform.

[13] Histograms of Oriented Gradients for Human Detection NavneetDalal and Bill Triggs

Facial expression & emotion detection & recognition is developed. For this purpose watershed method is used in combination with edge detection operation. It is a color based detection algorithm using color images in HSV color space. The RGB image is converted to HSV color image by which the image is separated in three regions hue, saturation, and intensity. After contrast enhancement watershed algorithm is applied to the image for each region. Canny edge detector is applied to the output image. After combining the three images final brain tumor segmented image is obtained. The algorithm has been applied on twenty. The developed algorithm has given promising results

[14] Face Recognition using Local Binary Patterns (LBP)By Md. Abdur Rahim, Md. Najmul Hossain, Tanzillah Wahid & Md. ShafiulAzam

Clustering approach is widely used in biomedical applications particularly for detection in abnormal magnetic resonance images. The effectiveness of the LSB algorithm in terms of computational rate is improved by modifying the cluster center and membership value updating criterion. In this paper, the application of modified FCM algorithm for is explored. Abnormal brain images from four tumor classes namely metastases, meningioma, glioma and astrocytoma are used.

[15] **Rapid Object Detection using a Boosted Cascade of Simple Features by Paul Viola and Michael Jones**

This work a new method for facial expression & emotion detection & recognition is developed. For this purpose watershed method is used in combination with edge detection operation. It is a color based detection algorithm using color images in HSV color space. The RGB image is converted to HSV color image by which the image is separated in three regions hue, saturation, and intensity. After contrast enhancement watershed algorithm is applied to the image for each region. Canny edge detector is applied to the output image. After combining the three images final brain tumor segmented image is obtained. The algorithm has been applied on twenty. The developed algorithm has given promising results.

[16] Face Detection and Recognition using Viola-Jones algorithm and Fusion of PCA and ANN by Narayan T. Deshpande and Dr. S. Ravishanar

W can be assure of identifying & recognizing ,In this work a new method for facial expression & emotion detection & recognition is developed. For this purpose watershed method is used in combination with edge detection operation. It is a color based detection algorithm using color images in HSV color space. The RGB image is converted to HSV color image by which the image is separated in three regions hue, saturation, and intensity. After contrast enhancement watershed algorithm is applied to the image for each region. Canny edge detector is applied to the output image. After combining the three images final brain tumor segmented image is obtained. The algorithm has been applied on twenty. The developed algorithm has given promising results.

[17] Neural Network- Based Face Detection Henry A. Rowley, Student Member, IEEE, ShumeetBaluja, and Takeo Kanade, Fellow, IEEE

For this purpose watershed method is used in combination with edge detection operation. It is a color based detection algorithm using color images in HSV color space. The RGB image is converted to HSV color image by which the image is separated in three regions hue, saturation, and intensity. After contrast enhancement watershed algorithm is applied to the image for each region. Canny edge detector is applied to the output image. After combining the three images final brain tumor segmented image is obtained. The algorithm has been applied on twenty. The developed algorithm has given

[18]

Intra-personal kernel space for face recognition Authors: S. Zhou, R. Chellappa, B. Moghaddam

It is a color based detection algorithm using color images in HSV color space. The RGB image is converted to HSV color image by which the image is separated in three regions hue, saturation, and intensity. After contrast enhancement watershed algorithm is applied to the image for each region. Canny edge detector is applied to the output image. After combining the three images final brain tumor segmented image is obtained. The algorithm has been applied on twenty. The developed algorithm has given promising results.

[19] **Facial Expression Recognition Using Eigenspaces,Vol. 10,2013 Authors: DebasmitaChakrabarti a, DebtanuDuttab**

In this work a new method for facial expression & emotion detection & recognition is developed. For this purpose watershed method is used in combination with edge detection operation. It is a color based detection algorithm using color images in HSV color space. The RGB image is converted to HSV color image by which the image is separated in three regions hue, saturation, and intensity. After contrast enhancement watershed algorithm is applied to the image for each region. Canny edge detector is applied to the output image. After combining the three images final brain tumor segmented image is obtained. The algorithm has been applied on twenty. The developed algorithm has given promising results.

[20]

A Human Facial Expression Recognition Model based on Eigen Face Approach,March 2015. Authors : Anurag Dea , AshimSahaa,Dr. M.C Pal.

After contrast enhancement watershed algorithm is applied to the image for each region. Canny edge detector is applied to the output image. After combining the three images final brain tumor segmented image is obtained. The algorithm has been applied on twenty. The developed algorithm has given

4.0 METHODOLOGY

Major steps involved :

- **Face Detection**
- **Gathering the dataset**
- **Training dataset**
- **Recognizing dataset**

Convolutional Neural Network

4.1 Description

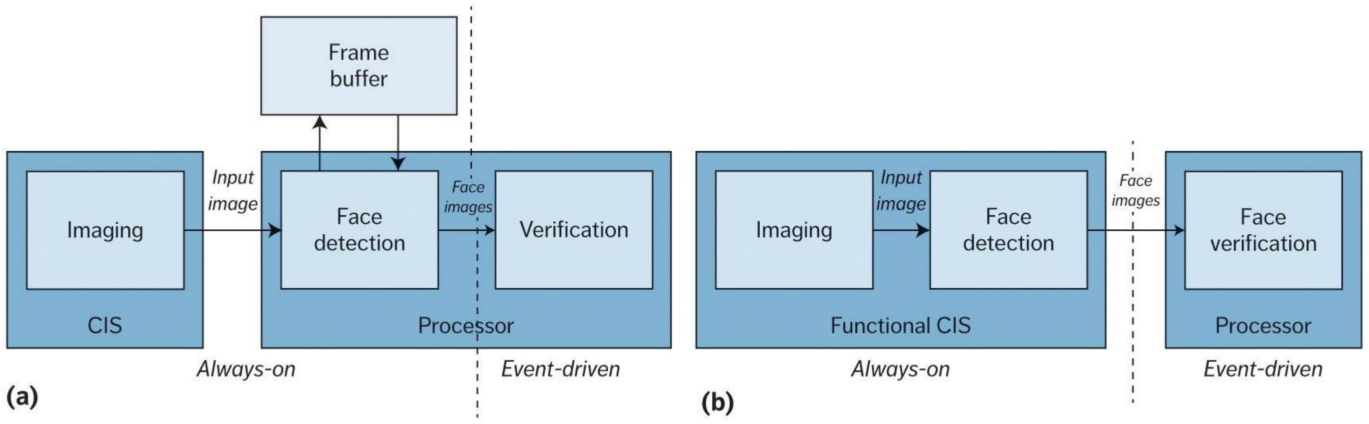
A CNN model can be thought as a combination of two components: feature extraction part and the classification part. The convolution + pooling layers perform feature extraction.

For example given an image, the convolution layer detects features such as two eyes, long ears, four legs, a short tail and so on. The fully connected layers then act as a classifier on top of these features, and assign a probability for the input image being a dog.

The convolution layers are the main powerhouse of a CNN model. Automatically detecting meaningful features given only an image and a label is not an easy task. The convolution layers learn such complex features by building on top of each other. The first layers detect edges, the next layers combine them to detect shapes, to following layers merge this information to infer that this is a nose. To be clear, the CNN doesn't know what a nose is. By seeing a lot of them in images, it learns to detect that as a feature. The fully connected layers learn how to use these features produced by convolutions in order to correctly classify the images.

CNNs use a variation of multilayer perceptrons designed to require minimal pre-processing. They are also known as shift invariant or space invariant artificial neural networks (SIANN), based on their shared-weights architecture and translation invariance characteristics.

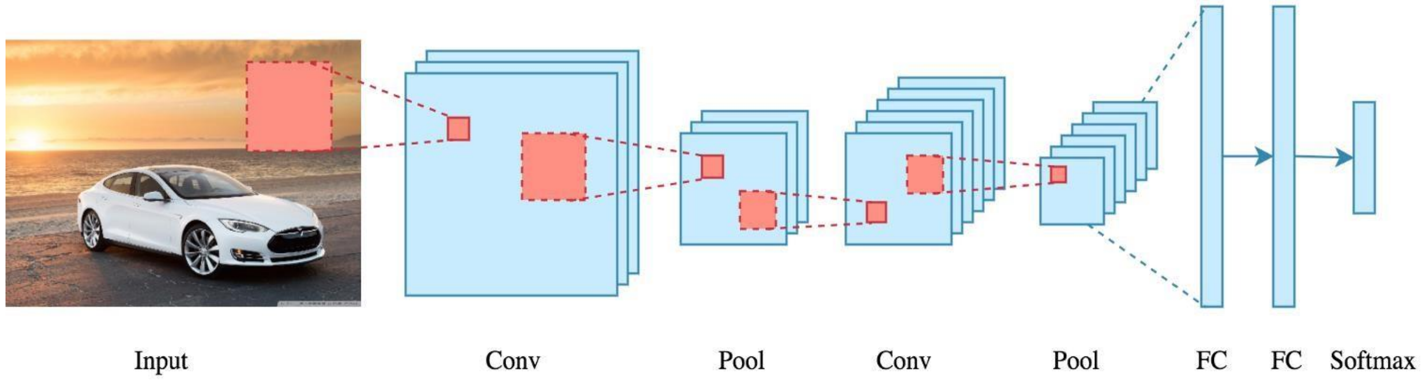
Convolutional networks were inspired by biological processes in which the connectivity pattern between neurons is inspired by the organization of the animal visual cortex. The receptive fields of different neurons partially overlap such that they cover the entire visual field.



main advantage of CNN compared to its predecessors is that it automatically detects the important features without any human supervision. For example, given many pictures of cats and dogs it learns distinctive features for each class by itself.

CNN is also computationally efficient. It uses special convolution and pooling operations and performs parameter sharing. This enables CNN models to run on any device, making them universally attractive.

4.2 Architecture



There is an input image that we're working with. We perform a series convolution + pooling operations, followed by a number of fully connected layers. If we are performing multiclass classification the output is softmax.

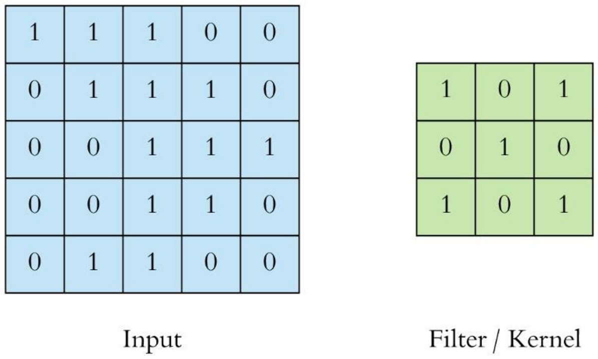
4.3 Convolution Layer

It is a Deep Learning algorithm which can take in an input image, assign importance (learnable weights and biases) to various aspects/objects in the image and be able to differentiate one from the other. The pre-processing required in a Convolution Layer is much lower as compared to other classification algorithms. While in primitive methods filters are hand-engineered, with enough training, Convolution Layer, have the ability to learn these filters/characteristics.

The architecture of a Convolution Layer, is analogous to that of the connectivity pattern of Neurons in the Human Brain and was inspired by the organization of the Visual Cortex. Individual neurons respond to stimuli only in a restricted region of the visual field known as the Receptive Field. A collection of such fields overlap to cover the entire visual area.

Convolution is the first layer to extract features from an input image. Convolution preserves the relationship between pixels by learning image features using small squares of input data. It is a mathematical operation that takes two inputs such as image matrix and a filter or kernel.

The main building block of CNN is the convolutional layer. Convolution is a mathematical operation to merge two sets of information. In our case the convolution is applied on the input data using a *convolution filter* to produce a *feature map*. There are a lot of terms being used so let’s visualize them one by one.



1	1	1	0	0
0	1	1	1	0
0	0	1	1	1
0	0	1	1	0
0	1	1	0	0

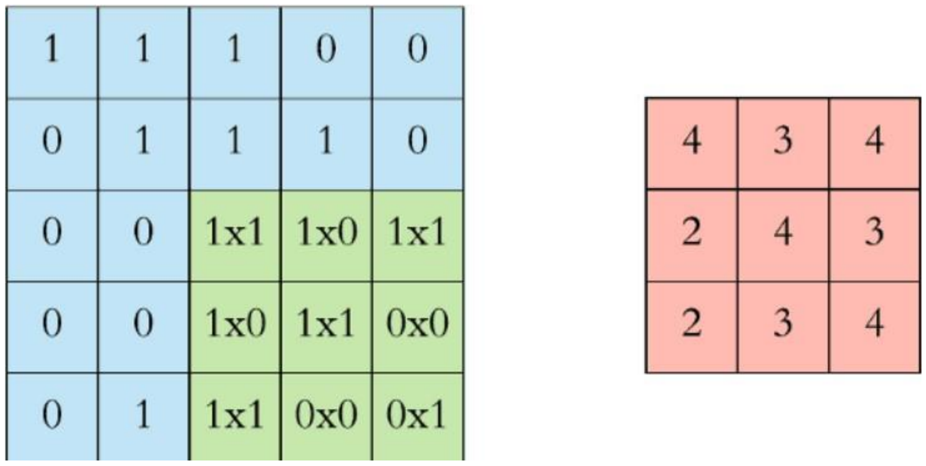
Input

1	0	1
0	1	0
1	0	1

Filter / Kernel

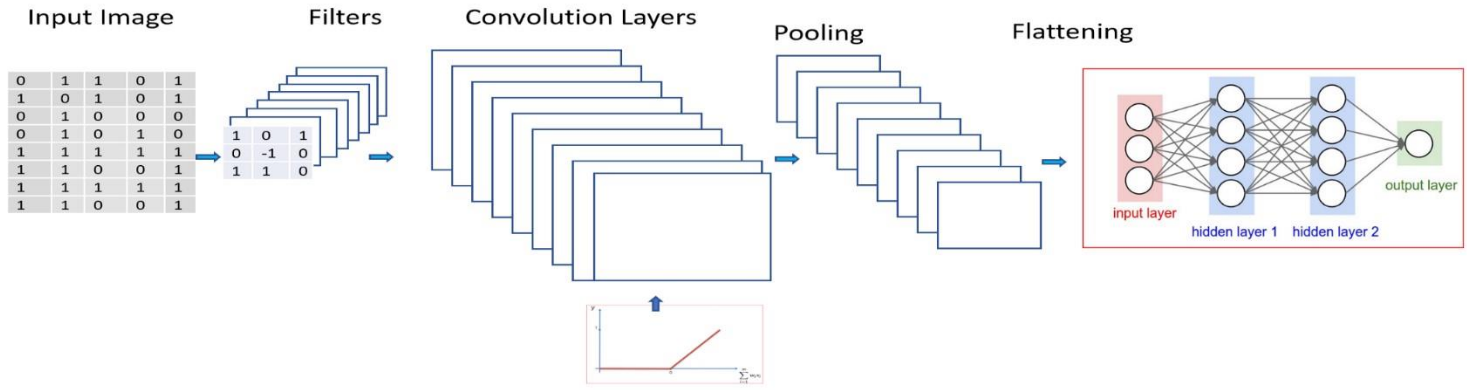
On the left side is the input to the convolution layer, for example the input image. On the right is the convolution *filter*, also called the *kernel*, we will use these terms interchangeably. This is called a *3x3 convolution* due to the shape of the filter.

We perform the convolution operation by sliding this filter over the input. At every location, we do element-wise matrix multiplication and sum the result. This sum goes into the feature map. The green area where the convolution operation takes place is called the *receptive field*. Due to the size of the filter the receptive field is also 3x3.



This was an example convolution operation shown in 2D using a 3x3 filter. But in reality these convolutions are performed in 3D. In reality an image is represented as a 3D matrix with dimensions of height, width and depth, where depth corresponds to color channels (RGB). A convolution filter has a specific height and width, like 3x3 or 5x5, and by design it covers the entire depth of its input so it needs to be 3D as well.

One more important point before we visualize the actual convolution operation. We perform multiple convolutions on an input, each using a different filter and resulting in a distinct feature map. We then stack all these feature maps together and that becomes the final output of the convolution layer. But first let’s start simple and visualize a **convolution using a single filter.**



Non- Linearity

For any kind of neural network to be powerful, it needs to contain non-linearity. Both the ANN and autoencoder we saw before achieved this by passing the weighted sum of its inputs through an activation function, and CNN is no different. We again pass the result of the convolution operation through *relu* activation function. So the values in the final feature maps are not actually the sums, but the relu function applied to them. We have omitted this in the figures above for simplicity. But keep in mind that any type of convolution involves a relu operation, without that the network won't achieve its true potential.

Stride and Padding

Stride specifies how much we move the convolution filter at each step.

We can have bigger strides if we want less overlap between the receptive fields. This also makes the resulting feature map smaller since we are skipping over potential locations.

We see that the size of the feature map is smaller than the input, because the convolution filter needs to be contained in the input. If we want to maintain the same dimensionality, we can use *padding* to surround the input with zeros

Now the dimensionality of the feature map matches the input. Padding is commonly used in CNN to preserve the size of the feature maps, otherwise they would shrink at each layer, which is not desirable. The 3D convolution figures we saw above used padding, that’s why the height and width of the feature map was the same as the input (both 32x32), and only the depth changed.

Fully Connected

After the convolution + pooling layers we add a couple of fully connected layers to wrap up the CNN architecture. This is the same fully connected ANN architecture we already talked .

Remember that the output of both convolution and pooling layers are 3D volumes, but a fully connected layer expects a 1D vector of numbers. So we *flatten* the output of the final pooling layer to a vector and that becomes the input to the fully connected layer. Flattening is simply arranging the 3D volume of numbers into a 1D vector, nothing fancy happens here.

4.4 Training CNN algorithm for face recognition:

CNN is trained the same way like ANN, backpropagation with gradient descent. Due to the convolution operation it’s more mathematically involved

- We first take a pre-trained convolutional neural network.
- Detect/identify faces in an image (using a face detection model) — for simplicity, this tutorial will only use images with one face/person in it, not more/less
- Predict face poses/landmarks (for the faces identified in step 2)
- Using data from step 2 and the actual image, calculate face encodings(numbers that describe the face)
- Compare the face encodings of known faces with those from test images to tell who is in the Picture
- Then,model is retrained. We train the last layer of the network based on number of classes that need to be detected.
- The next step is to get Region of Interest for each image. We reshape all these regions so they can match CNN size
- After getting the regions, we train SVM to classify objects and background. For each class, we train one binary SVM.
- Finally, we train a linear regression model to generate tighter bounding boxes for each identified object in the image .
- compare the unique features of that face to all the people you already kno

Backpropagation In Convolutional Neural Networks

Convolutional neural networks (CNNs) are a biologically-inspired variation of the multilayer perceptrons (MLPs). Neurons in CNNs share weights unlike in MLPs where each neuron has a separate weight vector. This sharing of weights ends up reducing the overall number of trainable weights hence introducing sparsity.

Existing between the convolution and the pooling layer is an activation function such as the ReLu layer; a non-saturating activation is applied element-wise, i.e. $f(x)=\max(0,x)$ and $f(x)=\max(0,x)$ thresholding at zero. After several convolutional and pooling layers, the image size (feature map size) is reduced and more complex features are extracted.

Cross-correlation

Given an input image "I" and a filter (kernel) "K" of dimensions $k_1 \times k_2$, the crosscorrelation operation is given by:

$$(I \otimes K)ij = \sum^{K}{}_{m}{}^{1}{}_{=}{}^{-}{}_{0}{}^{1} \qquad \sum^{k}{}_{n}{}^{2}{}_{=}{}^{-}{}_{0}{}^{1} I\,(i+m\,,\,j+n)\;K\,(m,n)$$
(1)

Convolution

Given an input image "I" and a filter (kernel) "K" of dimensions $k_1 \times k_2$, the convolution operation is given by:

$$(I * K)_{i\,j} = \sum Km1=-01 \qquad \sum^{k}{}_{n}{}^{2}{}_{=}{}^{-}{}_{0}{}^{1} I\,(i-m\,,\,j-n)\;K\,(m,n) \qquad (2)$$

$$= \sum^{K}{}_{m}{}^{1}{}_{=}{}^{-}{}_{0}{}^{1} \qquad \sum^{k}{}_{n}{}^{2}{}_{=}{}^{-}{}_{0}{}^{1} I\,(i+m\,,\,j+n)\;K\,(-m,-n) \qquad (3)$$

From Eq. 33 it is easy to see that convolution is the same as cross-correlation with a flipped kernel i.e: for a kernel K where $K(-m,-n) == K\,(m,n)$.

Convolution Neural Networks – CNNs

CNNs consists of convolutional layers which are characterized by an input map "I", a bank of filters '’K’ and biases "B".

In the case of images, we could have as input an image with height "H", width "W" and $C=3$ channels (red, blue and green) such that $I \in R^{HxWxC}$. Subsequently for a bank of D filters we have $K \in R^{k1xk2xCxD}$ and biases $b \in R^{D}$, one for each filter.

The output from this convolution procedure is as follows:

$$(I*K)_{i\,j} = \sum^{K}{}_{m}{}^{1}{}_{=}{}^{-}{}_{0}{}^{1} \sum{}_{n}{}^{k2}{}_{=}{}^{-}{}_{0}{}^{1} \sum^{C}{}_{c=1} K\ m, n, c * \ I\ i + m, j + n, c + b \ (4)$$

The convolution operation carried out here is the same as cross-correlation, except that the kernel is "flipped" (horizontally and vertically).

For the purposes of simplicity we shall use the case where the input image is grayscale i.e single channel C=1. The Eq. 44 will be transformed to:

$$(I*K)_{i\,j}$$

$$= \sum K m 1 = -0 1 \qquad \sum{}^{k}{}_{n}{}^{2}{}_{=}{}^{-}{}_{0}{}^{1}\ K\ m, n * \ I\ i + m, j + n + b \ (5)$$

Notation

To help us explore the forward and backpropagation, we shall make use of the following notation:

1. "I" is the lth layer where l=1 is the first layer and l=L is the last layer.
2. Input x is of dimension H×W and has i by j as the iterators
3. Filter or kernel w is of dimension k1×k2 has mm by n as the iterators
4. $W^{l}{}_{m,n}$ is the weight matrix connecting neurons of layer l with neurons of layer l−1.
5. B^{l} is the bias unit at layer l.
6. $X^{l}{}_{i,j}$, is the convolved input vector at layer l plus the bias represented as:

$$X^{l}{}_{i,j} = \sum_{m=0} \qquad \sum_{m=0} W\ l\ m\ , n \quad O\ l - 1\ i + m, j + n + b\ l$$

7. $O^l_{i,j}$ is the output vector at layer l given by

$$O^l_{i,j} = f(x^l_{i,j})$$

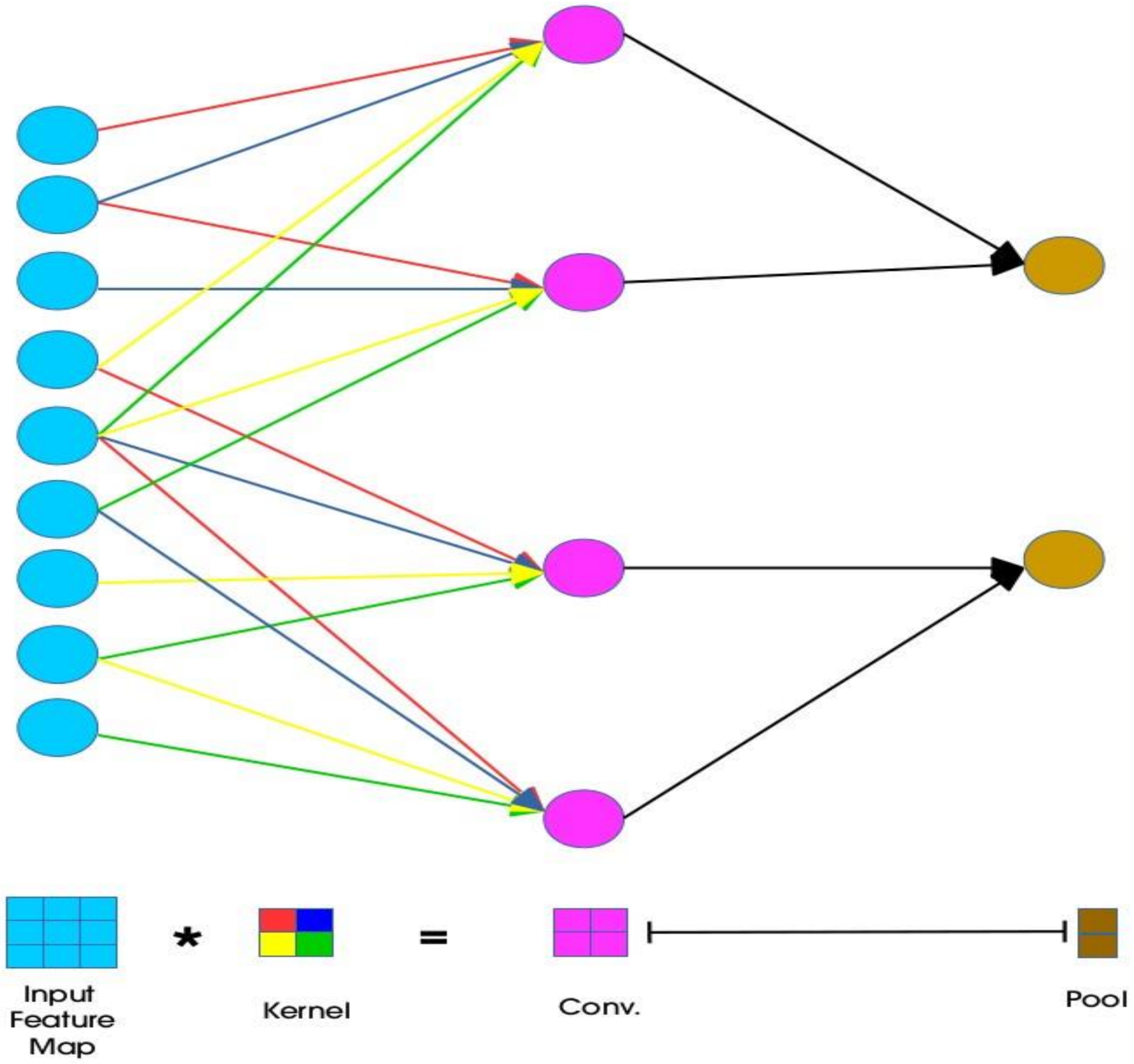
$f(\cdot)$ is the activation function. Application of the activation layer to the convolved input vector at layer l is given by $f(x^l_{i,j})$

Foward Propagation

To perform a convolution operation, the kernel is flipped 180° and slid across the input feature map in equal and finite strides. At each location, the product between each element of the kernel and the input input feature map element it overlaps is computed and the results summed up to obtain the output at that current location.

This procedure is repeated using different kernels to form as many output feature maps as desired. The concept of weight sharing is used as demonstrated in the diagram below:

Units in convolutional layer illustrated above have receptive fields of size 4 in the input feature map and are thus only connected to 4 adjacent neurons in the input layer. This is the idea of **sparse connectivity** in CNNs where there exists local connectivity pattern between neurons in adjacent layers.



Backpropagation

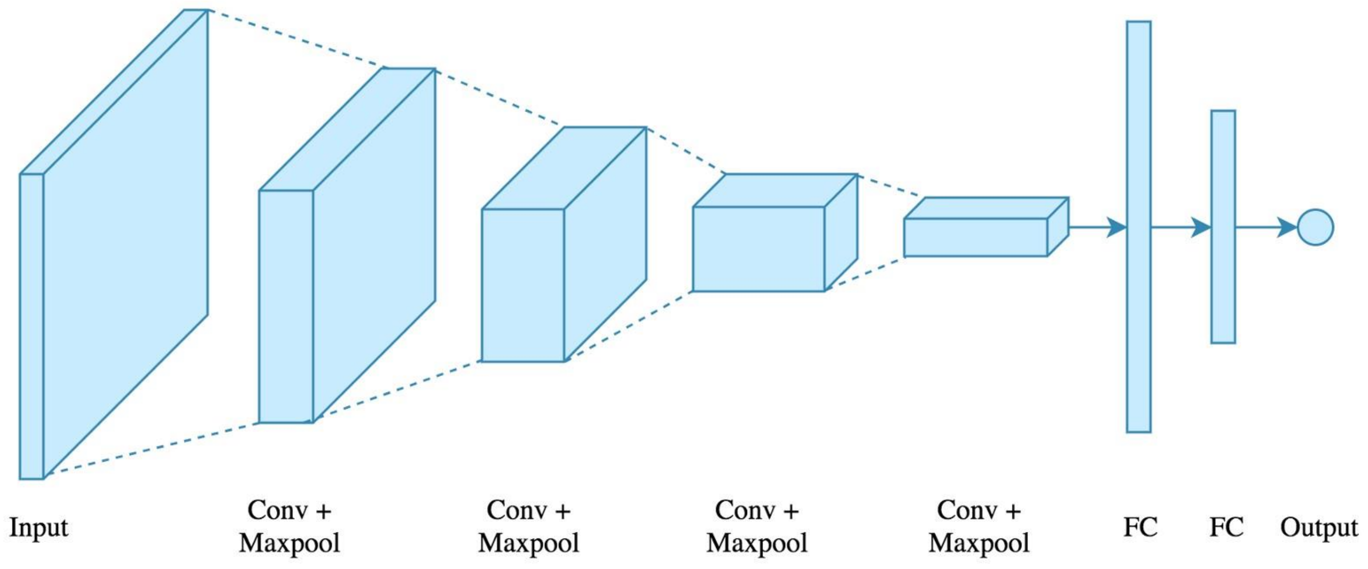
For backpropagation there are two updates performed, for the weights and the deltas. Lets begin with the weight update.

We are looking to compute $\partial E / \partial w^{l}_{m`,n`}$ which can be interpreted as the measurement of how the change in a single pixel $w_{m`,n`}$ in the weight kernel affects the loss function E.

Implementation

After this lengthy explanation let’s code up our CNN. We will use the Dogs vs Cats dataset from Kaggle to distinguish dog photos from cats.

We will use the following architecture: 4 convolution + pooling layers, followed by 2 fully connected layers. The input is an image of a cat or dog and the output is binary.



4.5 Code for CNN

```
Model = Sequential() model.add(Conv2D(32, (3, 3), activation='relu',
padding='same', name='conv_1',                    input_shape=(150, 150, 3)))
model.add(MaxPooling2D((2, 2), name='maxpool_1')) model.add(Conv2D(64, (3,
3), activation='relu', padding='same', name='conv_2'))
model.add(MaxPooling2D((2, 2), name='maxpool_2')) model.add(Conv2D(128,
(3, 3), activation='relu', padding='same', name='conv_3'))
model.add(MaxPooling2D((2, 2), name='maxpool_3')) model.add(Conv2D(128,
(3, 3), activation='relu', padding='same', name='conv_4'))
model.add(MaxPooling2D((2, 2), name='maxpool_4')) model.add(Flatten())
model.add(Dropout(0.5)) model.add(Dense(512, activation='relu',
name='dense_1')) model.add(Dense(128, activation='relu', name='dense_2'))
model.add(Dense(1, activation='sigmoid', name='output'))
```

- *Conv2D*: this method creates a convolutional layer. The first parameter is the filter count, and the second one is the filter size. For example in the first convolution layer we create 32 filters of size 3x3. We use *relu* non-linearity as activation. We also enable padding. In Keras there are two options for padding: *same* or *valid*. Same means we pad with the number on the edge and valid means no padding. Stride is 1 for convolution layers by default so we don’t change that. This layer can be customized further with additional parameters, you can check the documentation here.

- *MaxPooling2D*: creates a maxpooling layer, the only argument is the window size. We use a 2x2 window as it’s the most common. By default stride length is equal to the window size, which is 2 in our case, so we don’t change that.

- *Flatten*: After the convolution + pooling layers we flatten their output to feed into the fully connected layers as we discussed above.

- *Dropout*: we will explain this in the next section.

4.6 Pooling Layer

After a convolution operation we usually perform *pooling* to reduce the dimensionality. This enables us to reduce the number of parameters, which both shortens the training time and combats overfitting. Pooling layers down sample each feature map independently, reducing the height and width, keeping the depth intact.

The most common type of pooling is *max pooling* which just takes the max value in the pooling window. Contrary to the convolution operation, pooling has no parameters. It slides a window over its input, and simply takes the max value in the window. Similar to a convolution, we specify the window size and stride.

Here is the result of max pooling using a 2x2 window and stride 2. Each color denotes a different window. Since both the window size and stride are 2, the windows are not overlapping.

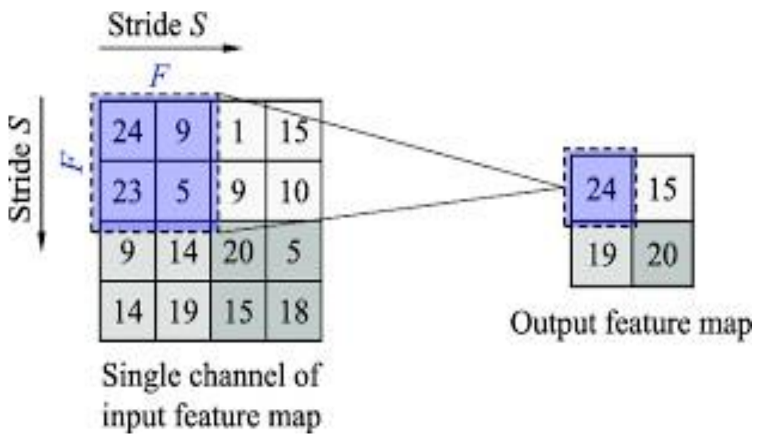
In CNN architectures, pooling is typically performed with 2x2 windows, stride 2 and no padding. While convolution is done with 3x3 windows, stride 1 and with padding.

The function of the pooling layer is to progressively reduce the spatial size of the representation to reduce the amount of parameters and computation in the network, and hence to also control overfitting. No learning takes place on the pooling layers [2].

To keep track of the "winning unit" its index noted during the forward pass and used for gradient routing during backpropagation. Gradient routing is done in the following ways:

- **Max-pooling** - the error is just assigned to where it comes from - the "winning unit" because other units in the previous layer's pooling blocks did not contribute to it hence all the other assigned values of zero

- **Average pooling -** the error is multiplied by 1N×N1N×N and assigned to the whole



pooling block (all units get this same value).

Max Pooling

Max pooling take the largest element from the rectified feature map. Taking the largest element could also take the average pooling. Sum of all elements in the feature map call as sum pooling.

$f(x) = \max(0,x)$.

Formular for the output after Max-pooling:

- $(N — F)/ S + 1$; where N = Dimension of input to pooling layer
- F = Dimension of filter
- S = Stride

5.0 PLATFORM

Python 3v

Libraries include :
OpenCV
Numpy

6.0 Sample Code

Program

```
import sys, os import pandas as pd import numpy as np
from sklearn.model_selection import train_test_split from keras.models import Sequential
from keras.layers import Dense, Dropout, Activation, Flatten from keras.layers import
Conv2D, MaxPooling2D, BatchNormalization
from keras.losses import categorical_crossentropy from keras.optimizers import Adam
from keras.regularizers import l2
from keras.callbacks import ReduceLROnPlateau, TensorBoard, EarlyStopping,
ModelCheckpoint
from keras.models import load_model
from keras.models import model_from_json
```

num_features = 64 num_labels = 7 batch_size = 64 epochs = 100 width, height = 48, 48

```
x = np.load('./fdataX.npy')
y = np.load('./flabels.npy')

x -= np.mean(x, axis=0)
x /= np.std(x, axis=0)


X_train, X_test, y_train, y_test = train_test_split(x, y, test_size=0.1, random_state=42)
X_train, X_valid, y_train, y_valid = train_test_split(X_train, y_train, test_size=0.1,
random_state=41)

np.save('modXtest', X_test)
np.save('modytest', y_test)

model = Sequential()

model.add(Conv2D(num_features, kernel_size=(3, 3), activation='relu',
input_shape=(width, height, 1), data_format='channels_last',
kernel_regularizer=l2(0.01)))
model.add(Conv2D(num_features, kernel_size=(3, 3), activation='relu', padding='same'))
model.add(BatchNormalization())
model.add(MaxPooling2D(pool_size=(2, 2), strides=(2, 2))) model.add(Dropout(0.5))

model.add(Conv2D(2*num_features, kernel_size=(3, 3), activation='relu',
padding='same'))
model.add(BatchNormalization())
model.add(Dense(num_labels, activation='softmax'))
```

```
#model.summary()

model.compile(loss=categorical_crossentropy,
              optimizer=Adam(lr=0.001, beta_1=0.9, beta_2=0.999, epsilon=1e-7),
metrics=['accuracy'])

model.fit(np.array(X_train), np.array(y_train),
          batch_size=batch_size,          epochs=epochs,          verbose=1,
          validation_data=(np.array(X_valid), np.array(y_valid)),          shuffle=True)

fer_json = model.to_json() with open("fer.json", "w") as json_file:
    json_file.write(fer_json) model.save_weights("fer.h5") print("Saved model to disk")
```

7.0 TEST CASES

7.1 Test Case -1

a.) Test case for Face Detection :

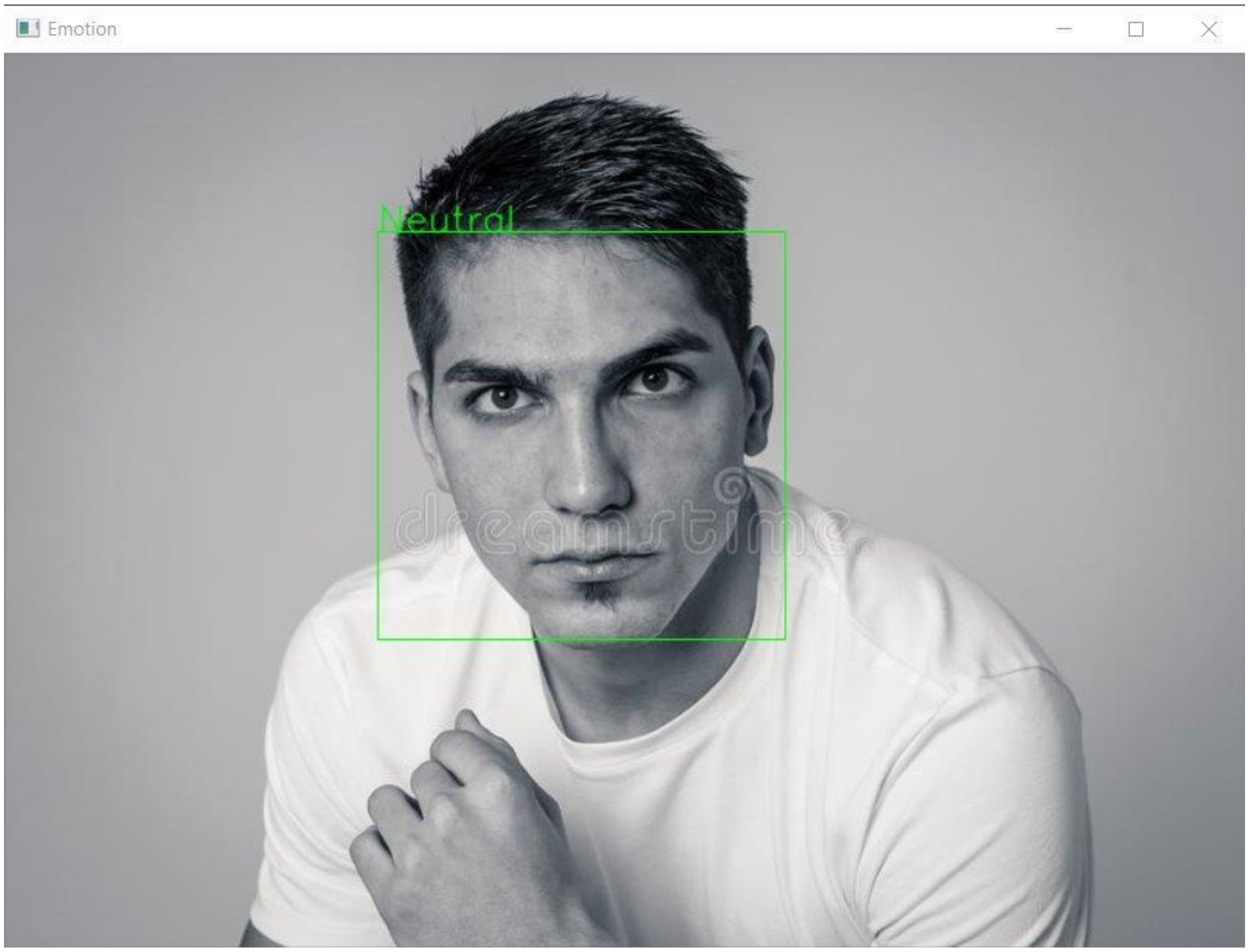
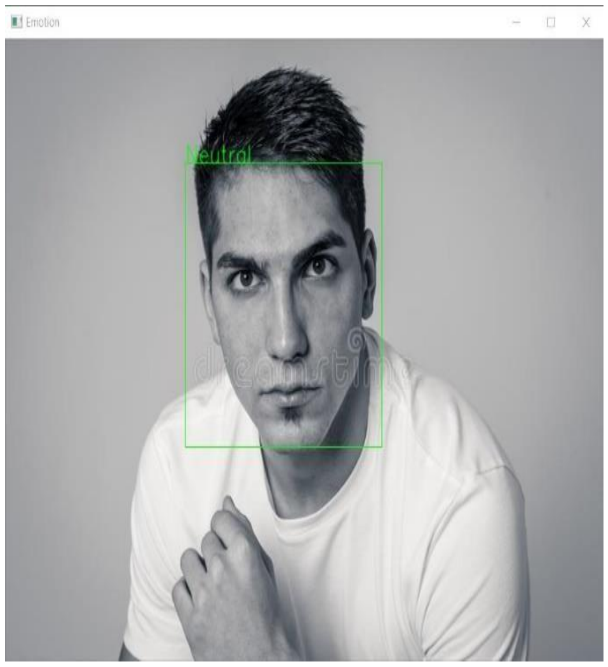
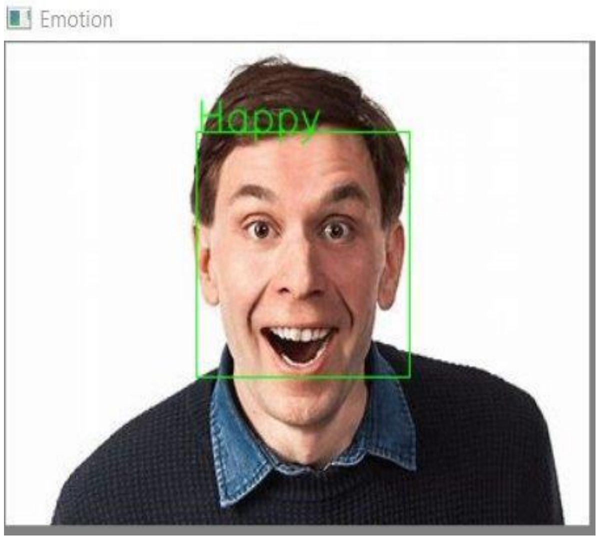


Fig 1. Output of face detection using HAAR

Sr. No.	Emotion	Normal Image	Output Image
1.	Neutral		
2.	Happy		

7.2 Test Case -2

b.) Test Cases Involving text and face recognition :

8.0 Result & Discussion

The computational models, which were implemented in this project, were chosen after extensive research, and the successful testing results confirm that the choices made by the researcher were reliable.The system with manual face detection and automatic face recognition did not have a recognition accuracy over 90%, due to the limited number of eigenfaces that were used for the PCA transform. This system was tested under very robust conditions in this experimental study and it is envisaged that real-world performance will be far more accurate.The fully automated frontal view face detection system displayed virtually perfect accuracy and in the researcher's opinion further work need not be conducted in this area.

We are checking the facial expression & emotion of of any person ,whose is neutral , meaning neither he is happy or angry .From this shows that , if any image person is shown to camera , it can show his/her expression & emotion. Due to this , we can able to identify mood of any person.

We are checking the facial expression & emotion of of any person ,whose is happy om given image , meaning neither he is neutral or angry .From this shows that , if any image person is shown to camera , it can show his/her expression & emotion. Due to this , we can able to identify mood of any person.

9.0 CONCLUSION & FUTURE WORK

Facial expression recognition is a challenging problem in the field of image analysis and computer vision that has received a great deal of attention over the last few years because of its many applications in various domains. This paper proposes a human facial expression recognition model based on eigenface approach in which the various emotions are recognized by calculating the Euclidean distance between the input test image and the mean of the eigenfaces of the training dataset. The field of research in expression recognition is an area which can be further explored and improved. This project focuses on directly transforming the dataset images to their eigen faces so as to depict a general sense in which these expressions are formed. The training is done by fisher face classifier and despite the small size of dataset an accuracy of **91%** is achieved. In order to optimize the training, we omitted 4 emotions out of 7 namely fear, sadness and contempt and disgust. We will expand the rest of the emotions with greater datasets for future work. These emotions can be recognized with greater accuracy when dataset with more number of images is used.

This project can be further enhanced by bringing in more emotions like disgust, fear, dissatisfaction , confused etc. which can be done by increasing the dataset.
Authorization of every registered human can be made stronger by training over larger dataset so that there is no chance of wrong identification.
With every emotion analyzed in various situations for a particular person , a hypothesis of his mental health can be produced to further use this in medical diagnosis and psychological experiments.

10.0 REFERENCES

[1] IJCSI International Journal of Computer Science Issues, Vol. 9, Issue 6, No 1, November 2012.Authors: Faizan Ahmad , AaimaNajam and Zeeshan Ahmed

[2] Facial Expression Recognition Using Eigenspaces,Vol. 10,2013 Authors: DebasmitaChakrabartia, DebtanuDuttab

[3] A Human Facial Expression Recognition Model based on Eigen Face Approach,March 2015.Authors : Anurag Dea , AshimSahaa,Dr. M.C Pal.

[4] Convolutional Neural Networks for Facial Expression Recognition,Authors : ShimaAlizadeh, Azar Fazel

[5] Convolutional Neural Networks for Facial Expression Recognition,Authors : ShimaAlizadeh, Azar Fazel

[6] Face Recognition using Local Binary Patterns (LBP). Authors : Md. Abdur Rahim, Md. Najmul Hossain, Tanzillah Wahid & Md. ShafiulAzam

[7] Face Detection and Recognition using Viola-Jones algorithm and Fusion of PCA and ANN Authors : Narayan T. Deshpande and Dr. S. Ravishankar

[8] Face Recognition using Local Binary Patterns (LBP) . Authors : Md. Abdur Rahim, Md. Najmul Hossain, Tanzillah Wahid & Md. ShafiulAzam

[9] Histograms of Oriented Gradients for Human Detection.sAuthors : Navneet Dalal and Bill Triggs

[10] .Fasel2003: Fasel, B. and Luettin, J., Automatic FacialExpression Analysis: A Survey. Pattern Recognition, 2003. 36(1). p:259-275

[11]. Ioannou, S., et al., Emotion recognition through facialexpression analysis based on a neurofuzzy network.NeuralNetworks, 2005. 18(2005 Special Issue): p. 423-435.

[12]. Yeasin, M., B. Bullot, and R. Sharma, Recognition of facialexpressions and measurement of levels of interest from video.Multimedia, IEEE Transactions on, 2006. 8(3): p. 500-508.

[13]. Sebe, N., et al. Emotion Recognition Based on Joint Visual andAudio Cues. in 18th International Conference on PatternRecognition 2006.

[14]. Takeo Kanade. Computer recognition of human faces, volume 47. Birkh¨auser Basel, 1977.

[15]. Lawrence Sirovich and Michael Kirby. Low-dimensional procedure for the characterization of human faces. Josa a, 4(3):519–524, 1987.

[16]. P. N. Belhumeur, J. P. Hespanha, and D. J. Kriegman. Eigenfaces vs. fisherfaces: recognition using class specific linear projection.

[17]. Rainer Lienhart and Jochen Maydt. An extended set of haar-like features for rapid object detection. In Image Processing. 2002. Proceedings. 2002 International Conference on, volume 1, pages I–I. IEEE, 2002.

[18]. Lawrence Sirovich and Michael Kirby. Low-dimensional procedure for the characterization of human faces. Josa a, 4(3):519–524, 1987

[19]. Human and Machine Vision, Proceedings of Workshop on Motion: Representation

[20]. Bichsel, M. (1991). Strategies of Robust Objects Recognition for Automatic Identification of Human Faces. PhD thesis, , Eidgenossischen Technischen Hochschule, Zurich.