N.M.A.M. INSTITUTE OF TECHNOLOGY
(An Autonomous Institution affiliated to Visvesvaraya Technological University, Belagavi)
Nitte – 574 110, Karnataka, India
(ISO 9001:2015 Certified), Accredited with 'A' Grade by NAAC
☎: 08258 - 281039 – 281263, Fax: 08258 – 281265

Department of Computer Science and Engineering
B.E. CSE Program Accredited by NBA, New Delhi from 1-7-2018 to 30-6-2021

Report on Mini Project

"FACE EMOJI CREATOR"

Course Code: 18CS605

Course Name: Machine Learning Lab

Semester: 6 Section: D

Submitted To

Ms.Keerthana B.Chigateri

Asst Prof Gd II, Dept of CSE, NMAMIT

Submitted By

Name: Shreyas K Shetty USN: 4NM18CS181
Name: Shashank V Jogi USN: 4NM18CS166

Date of submission: 15/06/2021

Signature of Course Instructor

NITTE EDUCATION TRUST | N.M.A.M. INSTITUTE OF TECHNOLOGY
(An Autonomous Institution affiliated to Visvesvaraya Technological University, Belagavi)
Nitte – 574 110, Karnataka, India

(ISO 9001:2015 Certified), Accredited with ‘A’ Grade by NAAC
��: 08258 - 281039 – 281263, Fax: 08258 – 281265

Department of Computer Science and Engineering

B.E. CSE Program Accredited by NBA, New Delhi from 1-7-2018 to 30-6-2021

CERTIFICATE

Certified that the project work carried out by **Shashank V Jogi (4NM18CS166)** and **Shreyas K Shetty(4NM18CS181)** bonafide students of NMAM Institute of Technology, Nitte in fulfilment for the Machine Learning lab in Computer Science and Engineering during the academic year 2020-2021.

Signature of the Examiners:Signature of the

Guide:

1.

2.

ACKNOWLEDGEMENT

The satisfactions that accompany the successful completion of any task would be incomplete without the mention of the people who made it possible. So, we acknowledge all those whose guidance and encouragement served as a beacon of light and crowned our efforts with success.

We are thankful to our project guide, Ms. Keerthana B. Chigateri, Assistant Professor Gd II, Department of Computer Science and Engineering for her valuable guidance and advice. Her willingness tomotivate us contributed tremendously to our project.

We would like to place on record our deep sense of gratitude to Dr. Jyothi Shetty, Head of Department of Computer Science and Engineering, NMAMIT, Nitte for his generous guidance, help and useful suggestions. We also acknowledge and express our sincere thanks to our beloved Dr. Niranjan. N. Chiplunkar, Principal, NMAMIT, Nitte who is a source of inspiration to us.

We thank all the Teaching and Non-Teaching staff members of the department of Computer Science and Engineering for providing resources for the completion of the project. A special thanks goes to our parents, friends and relatives forsupporting and encouraging us in all ways thus making our project successful. Finally, we thank all those who have contributed directly or indirectly in making this project a grand success.

Shashank V Jogi
(4NM18CS166)

Shreyas K Shetty
(4NM18CS181)

ABSTRACT:

These Human facial expressions convey a lot of information visually rather than articulately. Facial expression recognition plays a crucial role in the area of human-machine interaction. Automatic facial expression recognition system has many applications including, but not limited to, human behavior understanding, detection of mental disorders, and synthetic human expressions. Recognition of facial expression by computer with high recognition rate is still a challenging task.

Two popular methods utilized mostly in the literature for the automatic FER systems are based on geometry and appearance. Facial Expression Recognition usually performed in four-stages consisting of pre-processing, face detection, feature extraction, and expression classification.

In this project we applied various Machine learning methods to identify the key seven human emotions: anger, disgust, fear, happiness, sadness, surprise and neutrality

TABLE OF CONTENTS

TITLE **PAGE NO.**

---

1. <u>INTRODUCTION:</u>

"2018 is the year when machines learn to grasp human emotions" --Andrew Moore, the dean of computer science at Carnegie Mellon.

With the advent of modern technology our desires went high and it binds no bounds. In the present era a huge research work is going on in the field of digital image and image processing. The way of progression has been exponential and it is ever increasing. Image Processing is a vast area of research in present day world and its applications are very widespread.

Image processing is the field of signal processing where both the input and output signals are images. One of the most important application of Image processing is Facial expression recognition. Our emotion is revealed by the expressions in our face. Facial Expressions plays an important role in interpersonal communication. Facial expression is a non verbal scientific gesture which gets expressed in our face as per our emotions. Automatic recognition of facial expression plays an important role in artificial intelligence and robotics and thus it is a need of the generation. Some application related to this include Personal identification and Access control, Videophone and Teleconferencing, Forensic application, Human-Computer Interaction, Automated Surveillance, Cosmetology and so on.

The objective of this project is to develop Automatic Facial Expression Recognition System which can take human facial images containing some expression as input and recognize and classify it into seven different expression class such as :

I. Neutral
II. Angry
III. Disgust
IV. Fear
V. Happy
VI. Sadness
VII. Surprise

fig 1. Basic Human Emotion

Several Projects have already been done in this fields and our goal will not only be to develop an Automatic Facial Expression Recognition System but also improving the accuracy of this system compared to the other available systems.

2. LITERATURE SURVEY:

As per various literature surveys it is found that for implementing this project four basic steps are required to be performed.

i. Preprocessing

ii. Face registration

iii. Facial feature extraction

iv. Emotion classification

Description about all these processes are given below-

i. Preprocessing:

Preprocessing is a common name for operations with images at the lowest level of abstraction both input and output are intensity images. Most preprocessing steps that are implemented are –

a. Reduce the noise

b. Convert the Image to Binary/Grayscale.

c. Pixel Brightness Transformation.

d. Geometric Transformation.

fig 2. Pre-processing

ii. Face Registration:

Face Registration is a computer technology being used in a variety of applications that identifies human faces in digital images. In this face registration step, faces are first located in the image using some set of landmark points called “face localization” or “face detection”. These detected faces are then geometrically normalized to match some template image in a process called “face registration”.

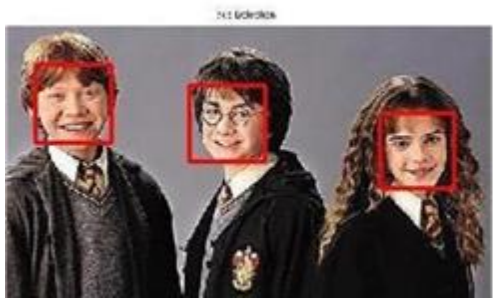


fig 3. Face Registration

iii. Facial Feature Extraction:

Facial Features extraction is an important step in face recognition and is defined as the process of locating specific regions, points, landmarks, or curves/contours in a given 2-D image or a 3D range image. In this feature extraction step, a numerical feature vector is generated from the resulting registered image. Common features that can be extracted are-

a. Lips

b. Eyes

c. Eyebrows

d. Nose tip

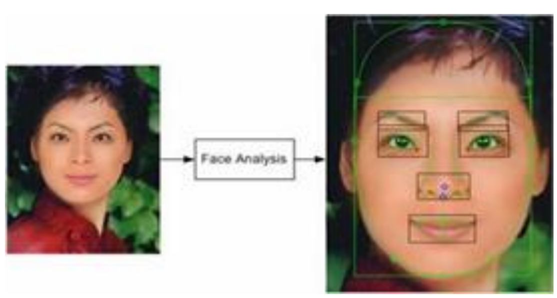


fig 4. Facial Feature Extraction

iv. Emotion Classification :

In the third step, of classification, the algorithm attempts to classify the given faces portraying one of the seven basic emotions.

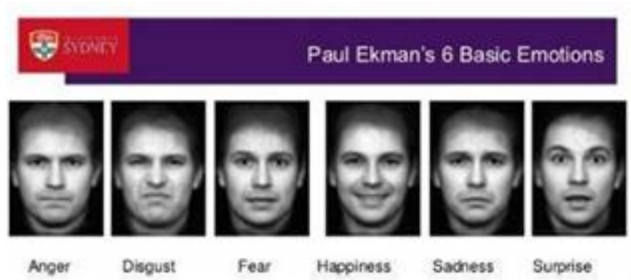


fig 5. Emotion Classification

Paul Ekman (born February 15, 1934) is an American psychologist and professor emeritus at the University of California, San Francisco who is a pioneer in the study of emotions and their relation to facial expressions. He has created an "atlas of emotions" with more than ten thousand facial expressions.

Different approaches which are followed for Facial Expression Recognition:

Neural Network Approach:

The neural network contained a hidden layer with neurons. The approach is based on the assumption that a neutral face image corresponding to each image is available to the system. Each neural network is trained independently with the use of on-line back propagation.

Neural Network will be discussed later.

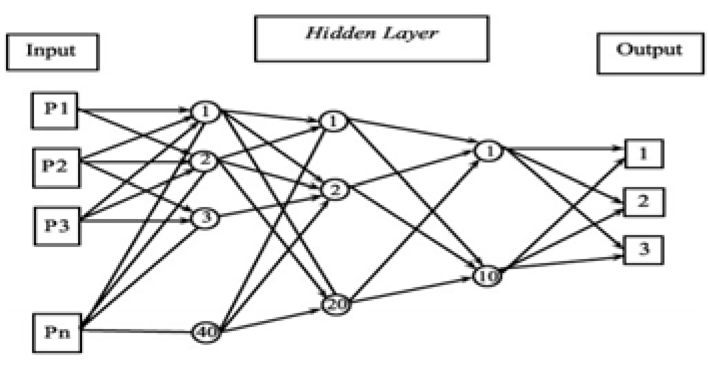


fig 6. Neural Network

Principal of Component Analysis:

Principal component analysis (PCA) is a statistical procedure that uses an orthogonal transformation to convert a set of observations of possibly correlated variables into a set of values of linearly uncorrelated variable called Principal Components.

Gabor Filter:

In image processing, a **Gabor filter**, named after Dennis Gabor, is a linear filter used for texture analysis, which means that it basically analyses whether there are any specific frequency content in the image in specific directions in a localized region around the point or region of analysis. Frequency and orientation representations of Gabor filters are claimed by many contemporary vision scientists to be similar to those of the human visual system, though there is no empirical evidence and no functional rationale to support the idea. They have been found to be particularly appropriate for texture representation and discrimination. In the spatial domain, a 2D Gabor filter is a Gaussian kernel function modulated by a sinusoidal plane wave.

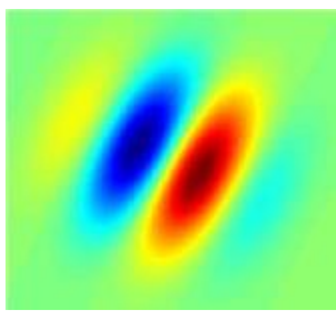


fig 7. Gabor Filter

Gabor filters are directly related to Gabor wavelets, since they can be designed for a number of dilations and rotations. However, in general, expansion is not applied for Gabor wavelets, since this requires computation of bi-orthogonal wavelets, which may be very time-consuming. Therefore, usually, a filter bank consisting of Gabor filters with various scales and rotations is created. The filters are convolved with the signal, resulting in a socalled Gabor space. This process is closely related to processes in the primary visual cortex. Jones and Palmer showed that the real part of the complex Gabor function is a good fit to the receptive field weight functions found in simple cells in a cat's striate cortex

Support Vector Machine :

In machine learning, **support vector machines** (**SVMs**, also **support vector networks**) are supervised learning models with associated learning algorithms that analyze data used for classification and regression analysis. Given a set of training examples, each marked as belonging to one or the other of two categories, an SVM training algorithm builds a model that assigns new examples to one category or the other, making it a non-probabilistic binary model (although methods such as Platt scaling exist to use SVM in a probabilistic classification setting). An SVM model is a representation of the examples as points in space, mapped so that the examples of the separate categories are divided by a clear gap that is as wide as possible. New examples are then mapped into that same space and predicted to belong to a category based on which side of the gap they fall.

In addition to performing linear classification, SVMs can efficiently perform a nonlinear classification using what is called the kernel trick implicitly mapping their inputs into high-dimensional feature spaces.

When data are not labeled, supervised learning is not possible, and an unsupervised learning approach is required, which attempts to find natural clustering of the data to groups, and then map new data to these formed groups. The **support vector clustering** algorithm created by Hava Siegelmann and Vladimir Vapnik, applies the statistics of support vectors, developed in the support vector machines algorithm, to categorize unlabeled data, and is one of the most widely used clustering algorithms in industrial applications.

Training & Testing Dataset:

In machine learning, the study and construction of algorithms that can learn from and make predictions on data is a common task. Such algorithms work by making data driven predictions or decisions, through building a mathematical model from input data.

The data used to build the final model usually comes from multiple datasets. In particular, three data sets are commonly used in different stages of the creation of the model.

The model is initially fit on a **training dataset**, that is a set of examples used to fit the parameters (e.g. weights of connections between neurons in artificial neural networks) of the model. The model (e.g. a neural net or a naive Bayes classifier) is trained on the training dataset using a supervised learning method (e.g. gradient descent or stochastic gradient descent). In practice, the training dataset often consist of pairs of an input vector and the corresponding *answer* vector or scalar, which is commonly denoted as the *target*. The current model is run with the training dataset and produces a result, which is then compared with the *target*, for each input vector in the training dataset. Based on the result of the comparison and the specific learning algorithm being used, the parameters of the model are adjusted. The model fitting can include both variable selection and parameter estimation.

Successively, the fitted model is used to predict the responses for the observations in a second dataset called the **validation dataset**. The validation dataset provides an unbiased evaluation of a model fit on the training dataset while tuning the model's hyperparameters (e.g. the number of hidden units in a neural network). Validation datasets can be used for regularization by early stopping: stop training when the error on the validation dataset increases, as this is a sign of overfitting to the training dataset. This simple procedure is complicated in practice by the fact that the validation dataset's error may fluctuate during training, producing multiple local minima. This complication has led to the creation of many ad-hoc rules for deciding when overfitting has truly begun.

Finally, the **test dataset** is a dataset used to provide an unbiased evaluation of a *final* model fit on the training dataset.

Various facial datasets available online are:

1. Japanese Female Facial Expression (JAFFE)

2. FER

3. CMU MultiPIE

4. Lifespan

5. MMI

6. FEED

7. CK

Accuracy of various databases :

Traing	*Testing*	*Accu*
FER2013	CK+	76.05
FER2013	CK+	73.38
JAFFE	CK+	54.05
MMI	CK+	66.20
FEED	CK+	56.60
FER2013	JAFFE	50.70
FER2013	JAFFE	45.07
CK+	JAFFE	55.87
BU-3DFE	JAFFE	41.96
CK	JAFFE	45.71
CK	JAFEE	41.30
FEED	JAFFE	46.48
FEED	JAFFE	60.09

3. DESIGN:

DATA FLOW DIAGRAM

A data flow diagram (DFD) is a graphical representation of the "flow" of data through an information system, modelling its process aspects. A DFD is often used as a preliminary step to create an overview of the system without going into great detail, which can later be elaborated. DFDs can also be used for the visualization of data processing.

A DFD shows what kind of information will be input to and output from the system, how the data will advance through the system, and where the data will be stored. It does not show information about process timing or whether processes will operate in sequence or in parallel, unlike a traditional structured flowchart which focuses on control flow, or a UML activity workflow diagram, which presents both control and data flows as a unified model.

Data flow diagrams are also known as bubble charts. DFD is a designing tool used in the top down approach to Systems Design.

DFD levels and layers

A data flow diagram can dive into progressively more detail by using levels and layers, zeroing in on a particular piece. DFD levels are numbered 0, 1 or 2, and occasionally go to even Level 3 or beyond. The necessary level of detail depends on the scope of what you are trying to accomplish.

DFD Level 0 is also called a Context Diagram. It‟s a basic overview of the whole system or process being analyzed or modeled. It‟s designed to be an at-a-glance view, showing the system as a single high-level process, with its relationship to external entities. It should be easily understood by a wide audience, including stakeholders, business analysts, data analysts and developers.

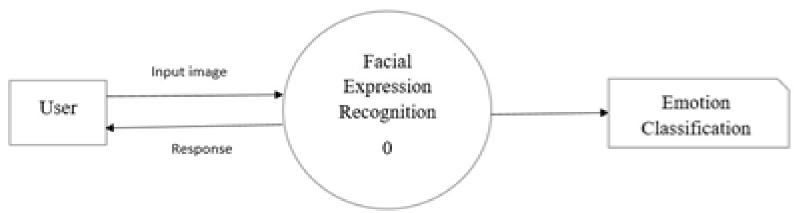
DFD Level 1 provides a more detailed breakout of pieces of the Context Level Diagram. You will highlight the main functions carried out by the system, as you break down the high-level process of the Context Diagram into its subprocesses.

DFD Level 2 then goes one step deeper into parts of Level 1. It may require more text to reach the necessary level of detail about the system’s functioning.

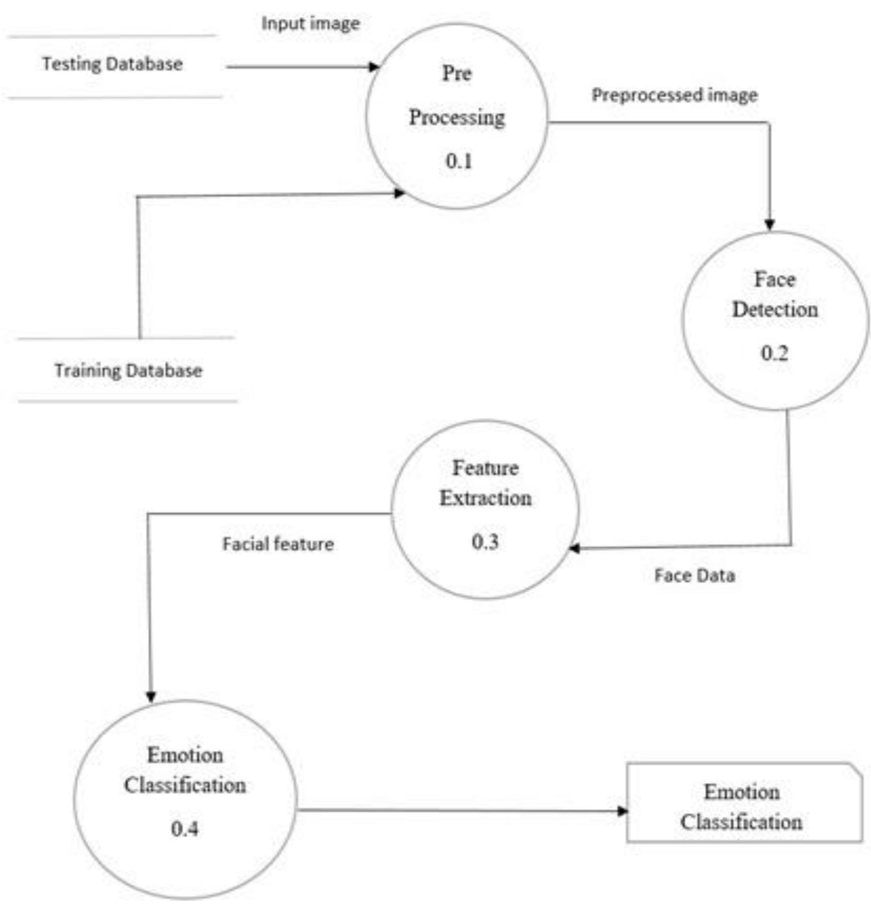
Progression to Levels 3, 4 and beyond is possible, but going beyond Level 3 is uncommon. Doing so can create complexity that makes it difficult to communicate, compare or model effectively.

Using DFD layers, the cascading levels can be nested directly in the diagram, providing a cleaner look with easy access to the deeper dive.

Level 0

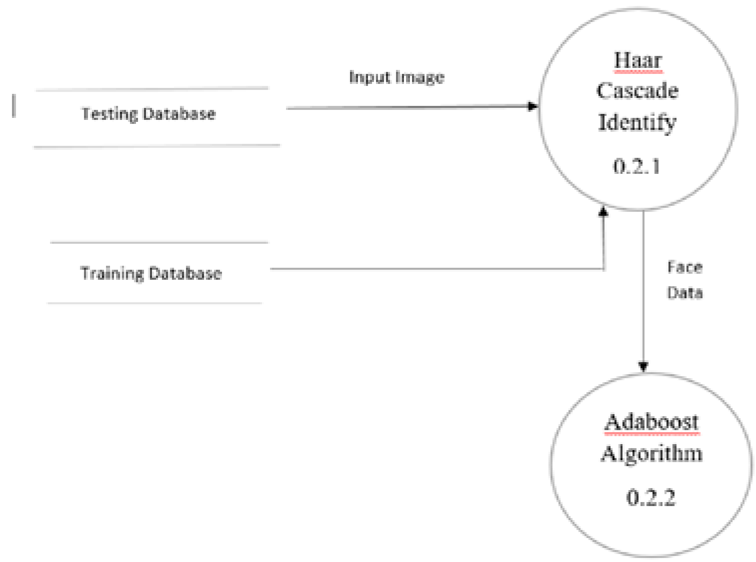


Level 1

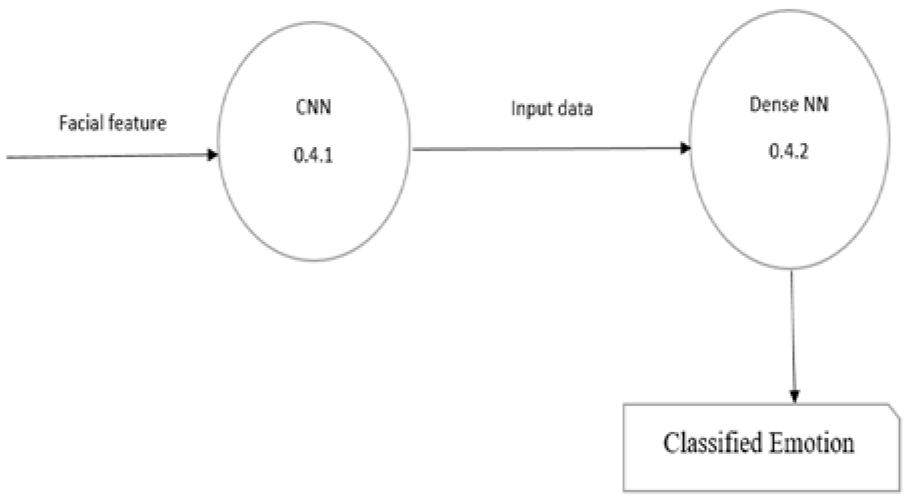


Level 2

Face Detection-



Emotion Classification-



4. IMPLEMENTATION DETAILS:

The Dataset:

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

The training set consists of 28,709 examples. The public test set used for the leaderboard consists of 3,589 examples. The final test set, which was used to determine the winner of the competition, consists of another 3,589 examples.

Emotion labels in the dataset:

0: -4593 images- ***Angry***

1: -547 images- ***Disgust***

2: -5121 images- ***Fear***

3: -8989 images- ***Happy***

4: -6077 images- ***Sad***

5: -4002 images- ***Surprise***

6: -6198 images- ***Neutral***

fig 8. FER2013 Images

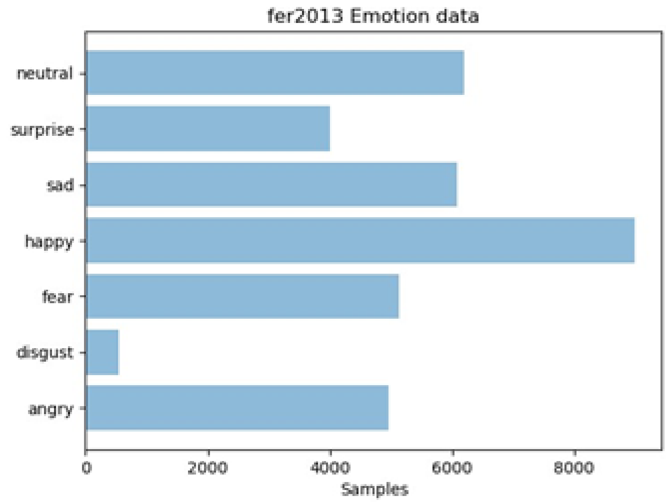


fig 9. FER2013 Samples

The dataset we used for training the model is from a Kaggle Facial Expression Recognition Challenge a few years back (FER2013). It comprises a total of **35887 pre-cropped, 48-by-48-pixel grayscale images** of faces each labeled with one of the 7 emotion classes: **anger, disgust, fear, happiness, sadness, surprise, and neutral**.

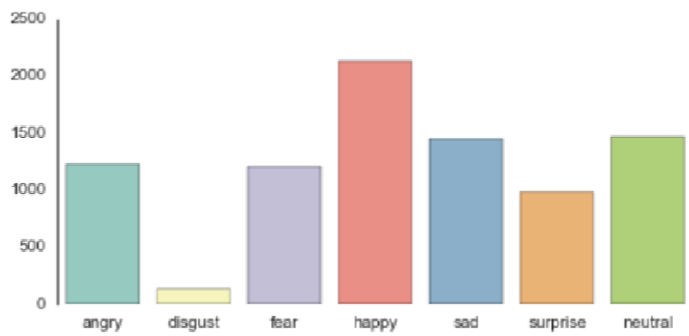


Fig 10. Overview Of FER2013 Database

As we were exploring the dataset, we discovered an imbalance of the "disgust" class compared to many samples of other classes. We decided to merge disgust into anger given that they both represent similar sentiment. To prevent data leakage, we built a data generator **fer2013datagen.**py that can easily separate training and hold-out set to different files. We used 28709 labeled faces as the training set and held out the remaining two test sets (3589/set) for after-training validation. The resulting is a **6-class, balanced dataset**, that contains angry, fear, happy, sad, surprise, and neutral. Now we are ready to train.

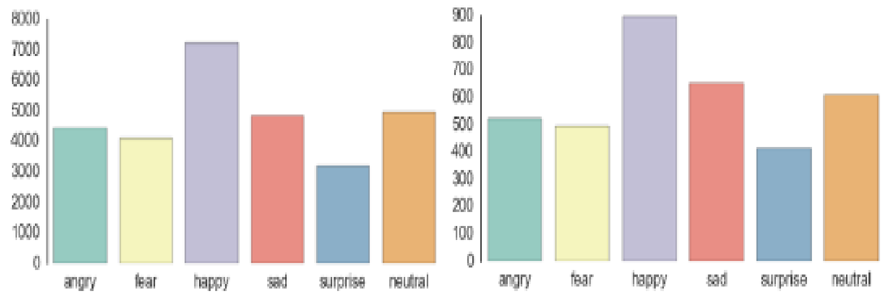


Fig 11. Training & Validation Data Distribution

The Model:

Deep learning is a popular technique used in computer vision. We chose Convolutional Neural Network (CNN) layers as building blocks to create our model architecture. CNNs are known to imitate how the human brain works when analyzing visuals. We have used a picture of Mr. Bean as an example to explain how images are fed into the model, **because who doesn't love Mr. Bean?**

A typical architecture of a convolutional neural network contains an input layer, some convolutional layers, some dense layers (aka. fully-connected layers), and an output layer. These are linearly stacked layers ordered in sequence. In Keras, the model iscreated as `Sequential()` and more layers are added to build architecture.

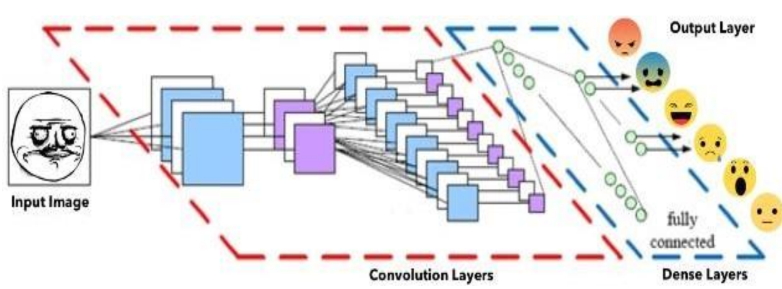


Fig 12. FER CNN Architecture

- **Input Layer:**

The input layer has pre-determined, fixed dimensions, so the image must be **pre-processed** before it can be fed into the layer. We used OpenCV, a computer vision library, for face detection in the image. The haar-cascade_frontalface_default.xml in OpenCV contains pre-trained filters anduses

Adaboost to quickly find and crop the face.

The cropped face is then converted into grayscale using cv2.cvtColor and resized to 48-by-48 pixels with cv2.resize. This step greatly reduces the dimensions compared to the original RGB format with three color dimensions (3, 48, 48). The pipeline ensures every image can be fed into the input layer as a (1, 48, 48) numpy array.

- **<u>Convolutional Layers:</u>**

The numpy array gets passed into the Convolution2D layer where we specify the number of filters as one of the hyperparameters. The set of filters (aka. kernel) are unique with randomly generated weights. Each filter, (3, 3) receptive field, slides across the original image with shared weights to create a feature map. Convolution generates feature maps that represent how pixel values are enhanced, for example, edge and pattern detection. A feature map is created by applying filter 1 across the entire image. Other filters are applied one after another creating a set of feature maps.

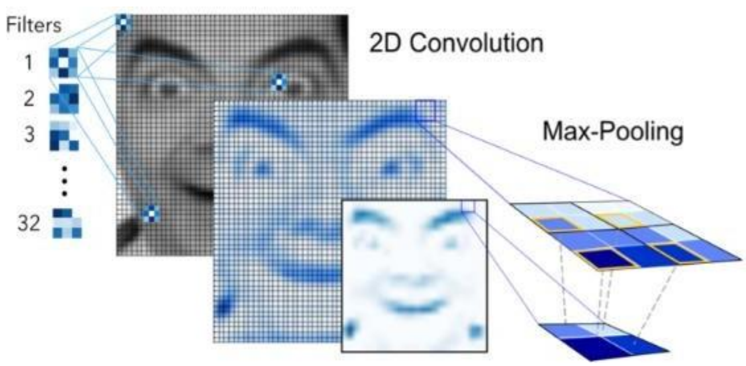


Fig 13. Convolutional & Maxpooling of Neural Network

Pooling is a dimension reduction technique usually applied after one or several convolutional layers. It is an important step when building CNNs as adding more convolutional layers can greatly affect computational time. We used a popular pooling method called MaxPooling2D that uses (2, 2) windows across the feature map only keeping the maximum pixel value. The pooled pixels form an image with dimensions reduced by 4.

- **<u>Dense Layers:</u>**

The dense layer (aka fully connected layers), is inspired by the way neurons transmit signals through the brain. It takes a large number of input features and transform features through layers connected with trainable weights.

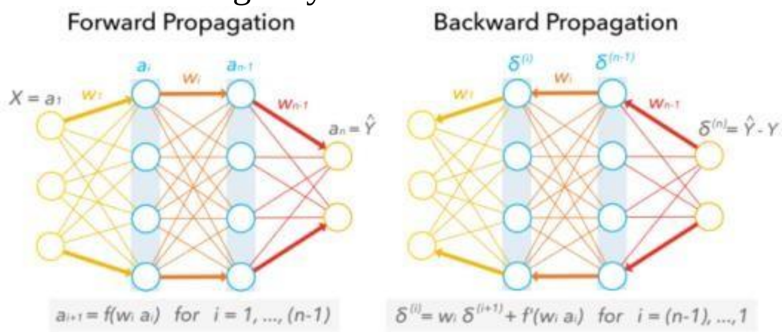


Fig 14.

These weights are trained by forward propagation of training data then backward propagation of its errors. **Back propagation** starts from evaluating the difference between prediction and true value, and back calculates the weight adjustment needed to every layer before. We can control the training speed and the complexity of the architecture by tuning the hyper-parameters, such as **learning rate** and **network density**. As we feed in more data, the network is able to gradually make adjustments until errors are minimized. Essentially, the more layers/nodes we add to the network the better it can pick up signals. As good as it may sound, the model also becomes increasingly prone to overfitting the training data. One method to prevent overfitting and generalize on unseen data is to apply **dropout**. Dropout randomly selects a portion (usually less than 50%) of nodes to set their weights to zero during training. This method can effectively control the model's sensitivity to noise during training while maintaining the necessary complexity of the architecture.

- **Output Layer:**

Instead of using sigmoid activation function, we used **softmax** at the output layer. This output presents itself as a probability for each emotion class. Therefore, the model is able to show the detail probability compositionof the emotions in the face. As later on, you will see that it is not efficient to classify human facial expression as only a single emotion. Our expressions areusually much complex and contain a mix of emotions that could be used to accurately describe a particular expression.

Deep Learning we built a simple CNN with an input, three convolution layers, one dense layer, and an output layer to start with. As it turned out, the simple model performed poorly. The low accuracy of 0.1500 showed that it was merely random guessing one of the six emotions. The simple net architecture failed to pick up the subtle details in facial expressions. This could only mean one thing…

This is where deep learning comes in. Given the pattern complexity of facial expressions, it isnecessary to build with a deeper architecture in order to identify subtle signals. So, we fiddled combinations of three components to increase model complexity:

Models with various combinations were trained and evaluated using GPU computing g2.2xlarge on Amazon Web Services (AWS). This greatly reduced training time and increased efficiency in tuning the model. In the end, our final net architecture was 9 layers deep in convolution with one max-pooling after every three convolution layers as seen below.

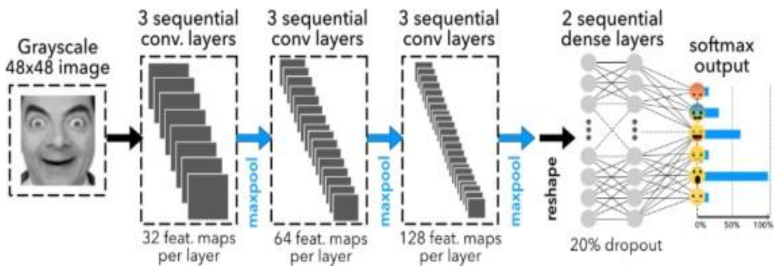
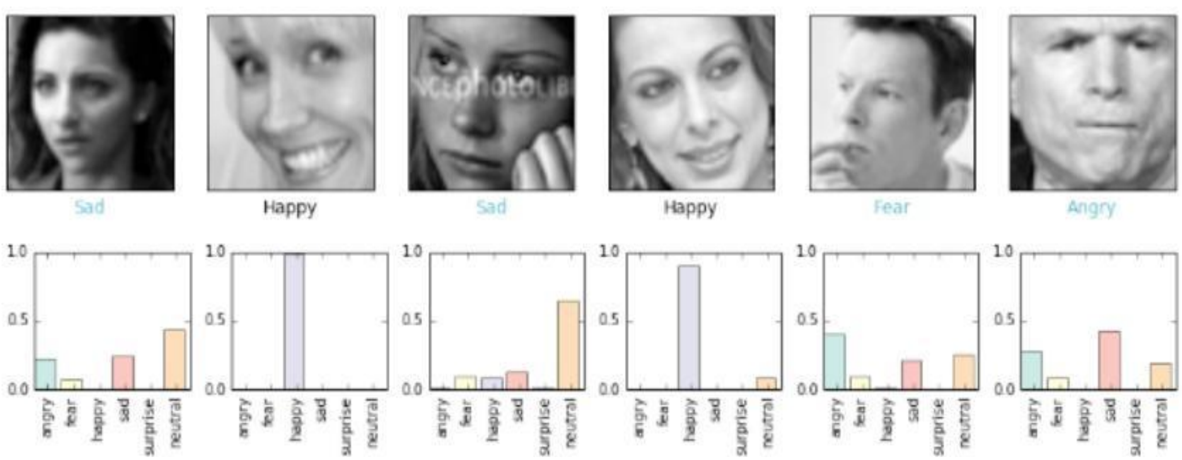


Fig 15. CNN Forward & Backward Propagation

- <u>**Model Validation:**</u>

Performance As it turns out, the final CNN had a **validation accuracy of 58%**. Thisactually makes a lot of sense. Because our expressions usually consist a combinationof emotions, and *only* using one label to represent an expression can be hard. **In thiscase, when the model predicts incorrectly, the correct label is often the second most likely emotion as seen in figure below.**



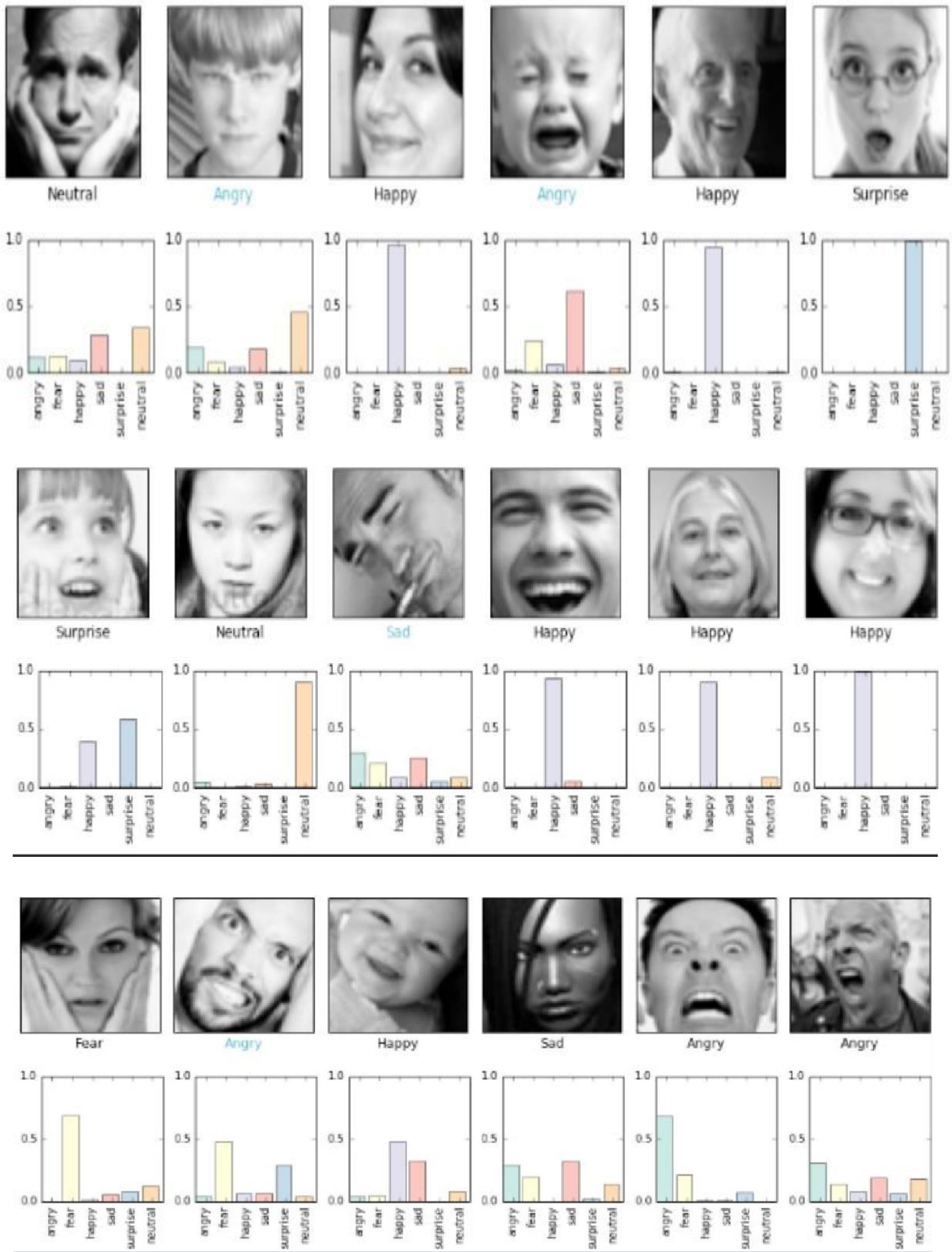


Fig 16. Prediction Of Example Faces from Database

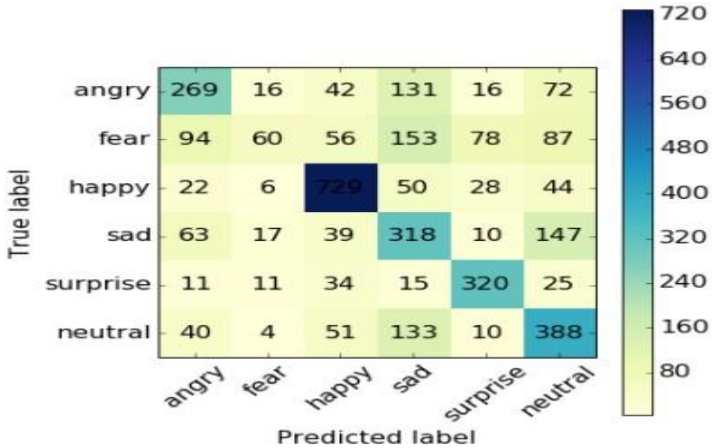


Fig 17. Confusion Matrix

The confusion matrix gives the counts of emotion predictions and some insights to theperformance of the multi-class classification model:

- The model performs really well on classifying **positive emotions** resulting in relatively high precision scores for happy and surprised. **Happy** has a precision of 76.7% which could be explained by having the most examples (~7000) in the training set. Interestingly, **surprise** has a precision of 69.3% having the least examples in the training set. There must be very strong signals in the surprise expressions.
- Model performance seems weaker across **negative emotions** on average. In particularly, the emotion **sad** has a low precision of only 39.7%. The model frequently misclassified angry, fear and neutral as sad. In addition, it is most confused when predicting sad and neutral faces because these two emotions are probably the least expressive (excluding crying faces).
- Frequency of prediction that misclassified by less than 3 ranks.

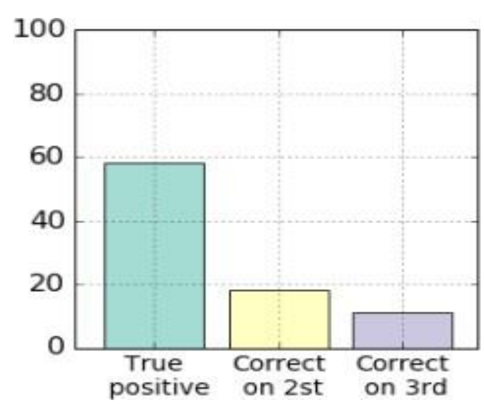


Fig 18. Correct Prediction On 2^{nd}& 3^{rd} Highest Probable Emotion

Computer Vision As a result, the feature maps become increasingly abstract down the pipeline when more pooling layers are added. This gives an idea of what the machine sees in feature maps after 2nd and 3rd max-pooling.

Fig 19. CNN Feature Maps After 2nd Layer Of Maxpooling

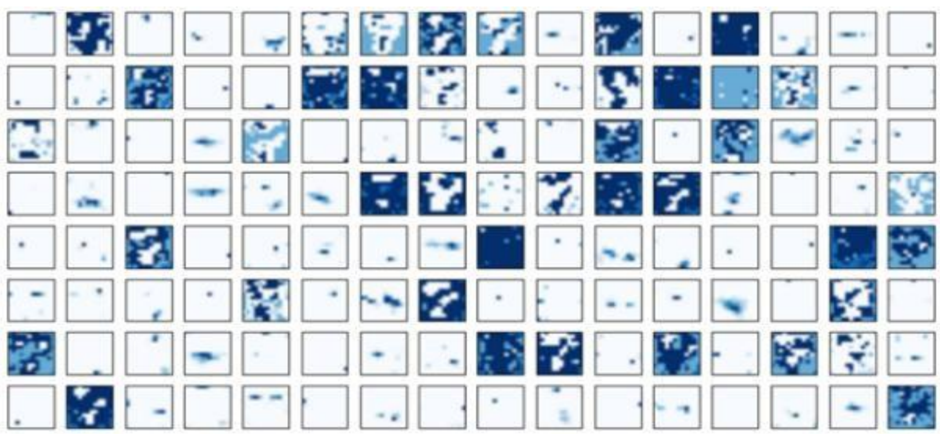
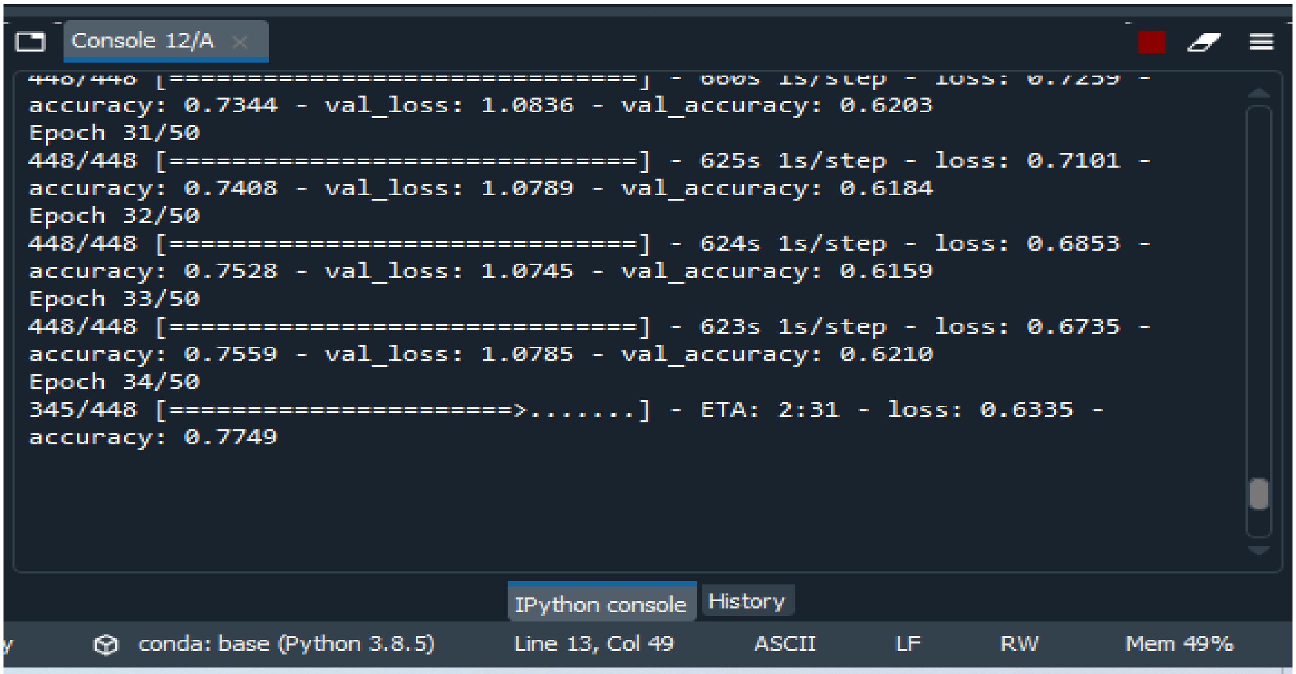


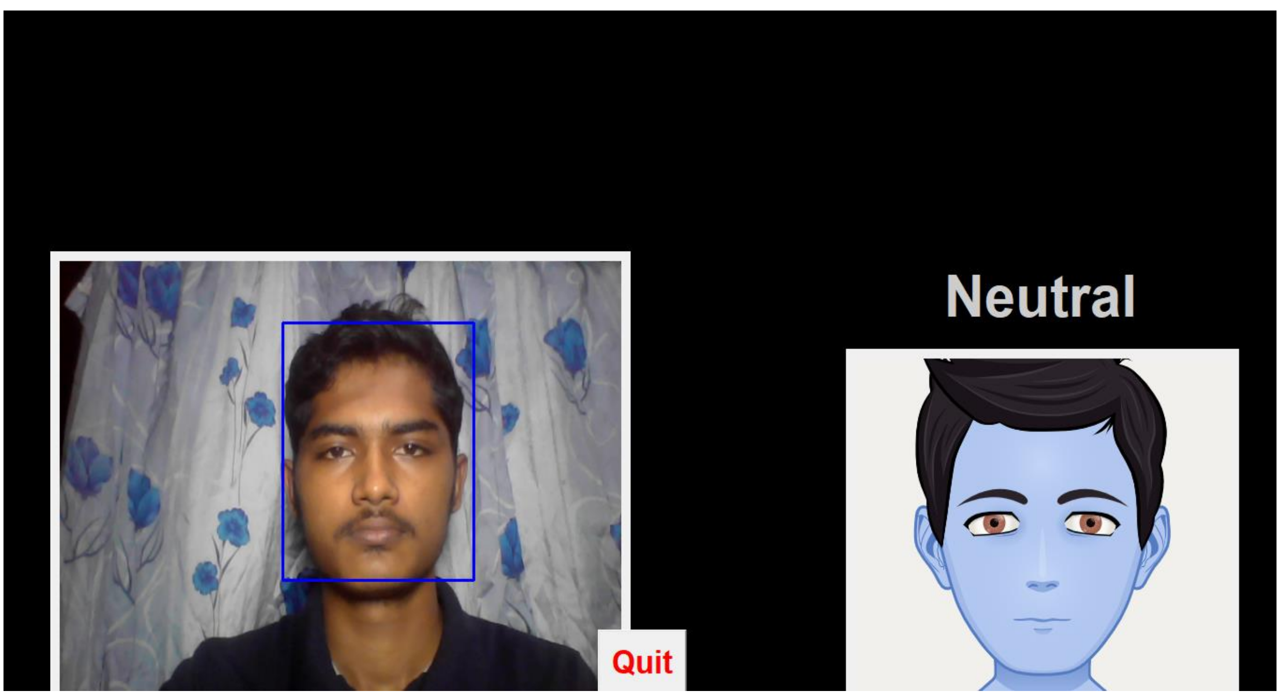
Fig 20. CNN Feature Maps After 3rd Layer OfMaxpooling

5. RESULT:

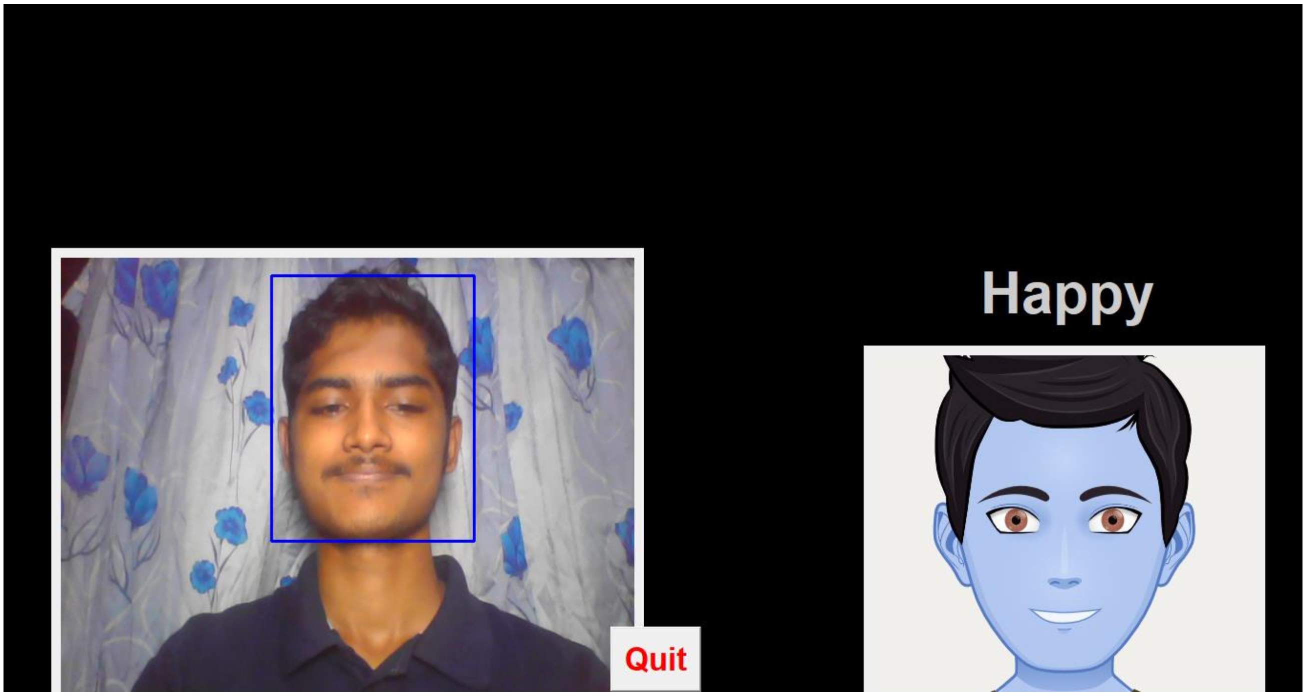
Dataset Training:



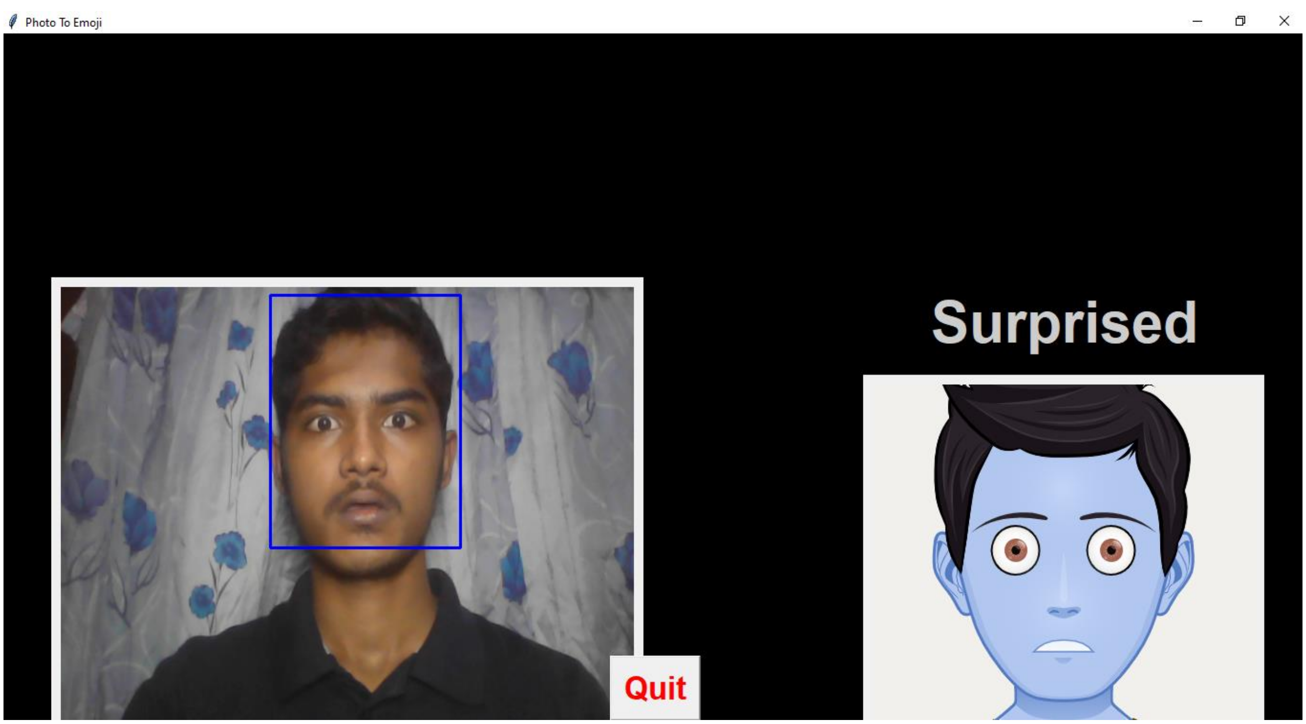
Output Sample 1



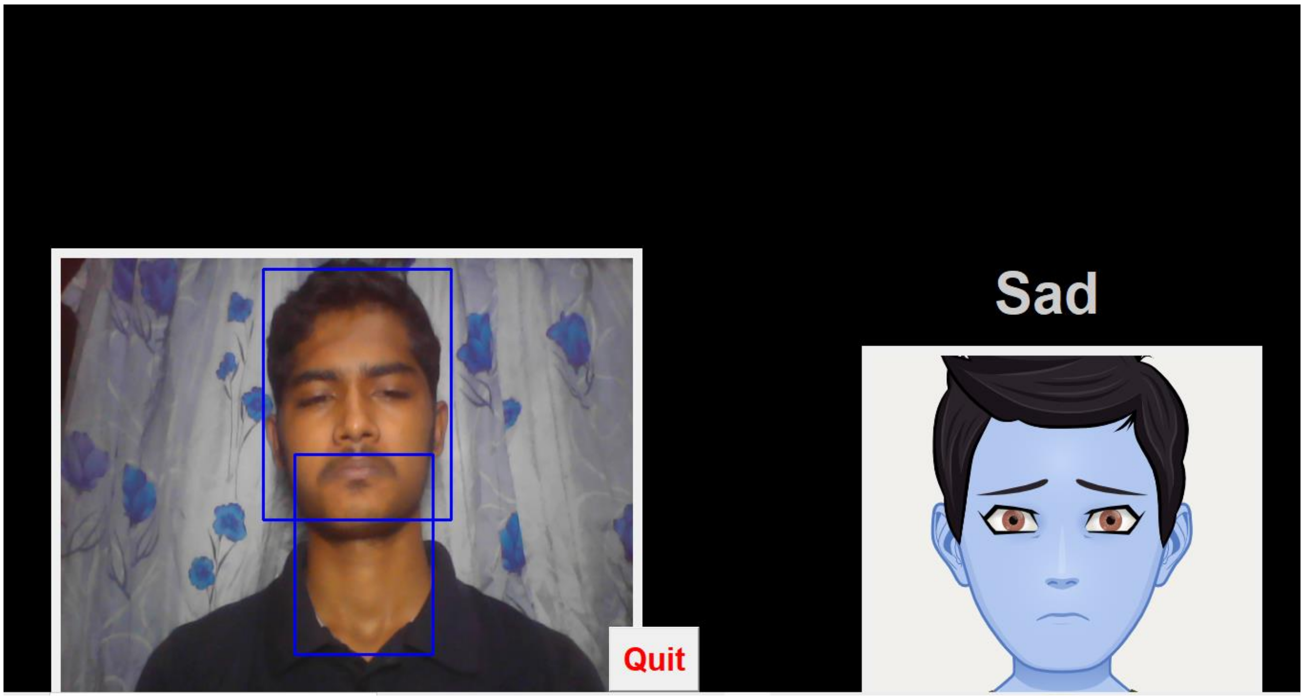
Output Sample 2



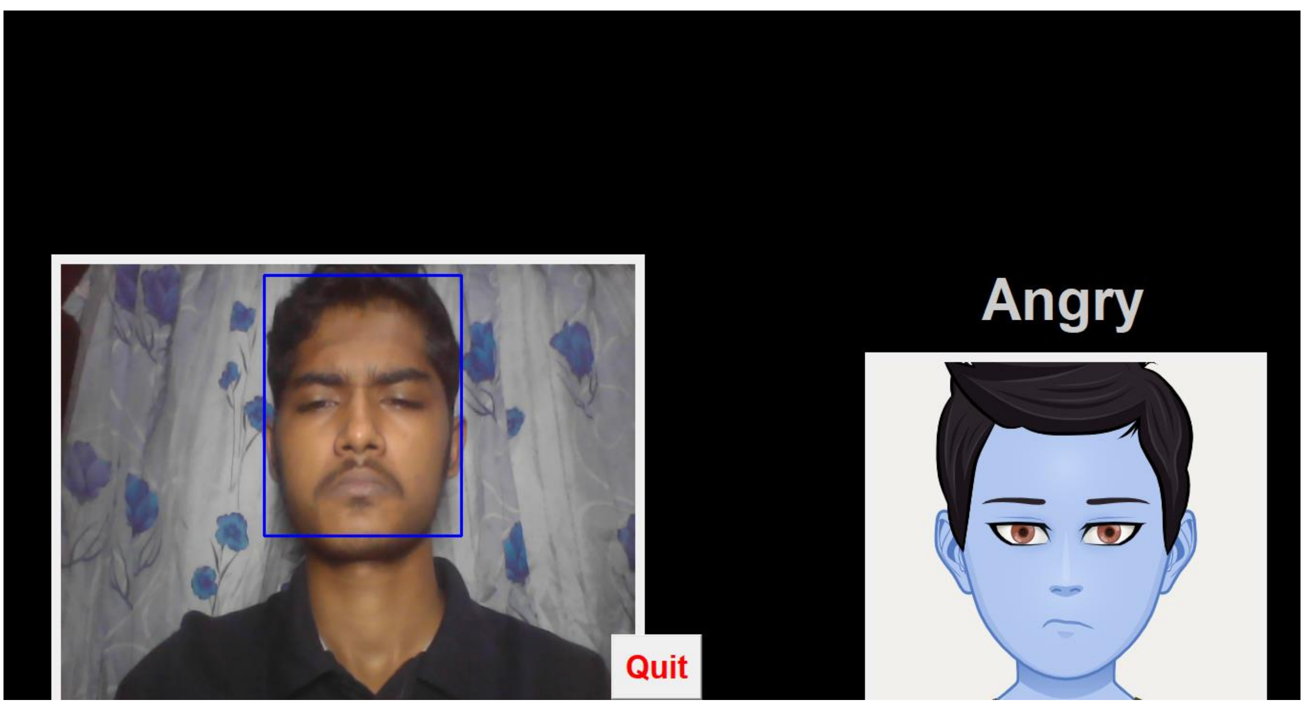
Output Sample 3



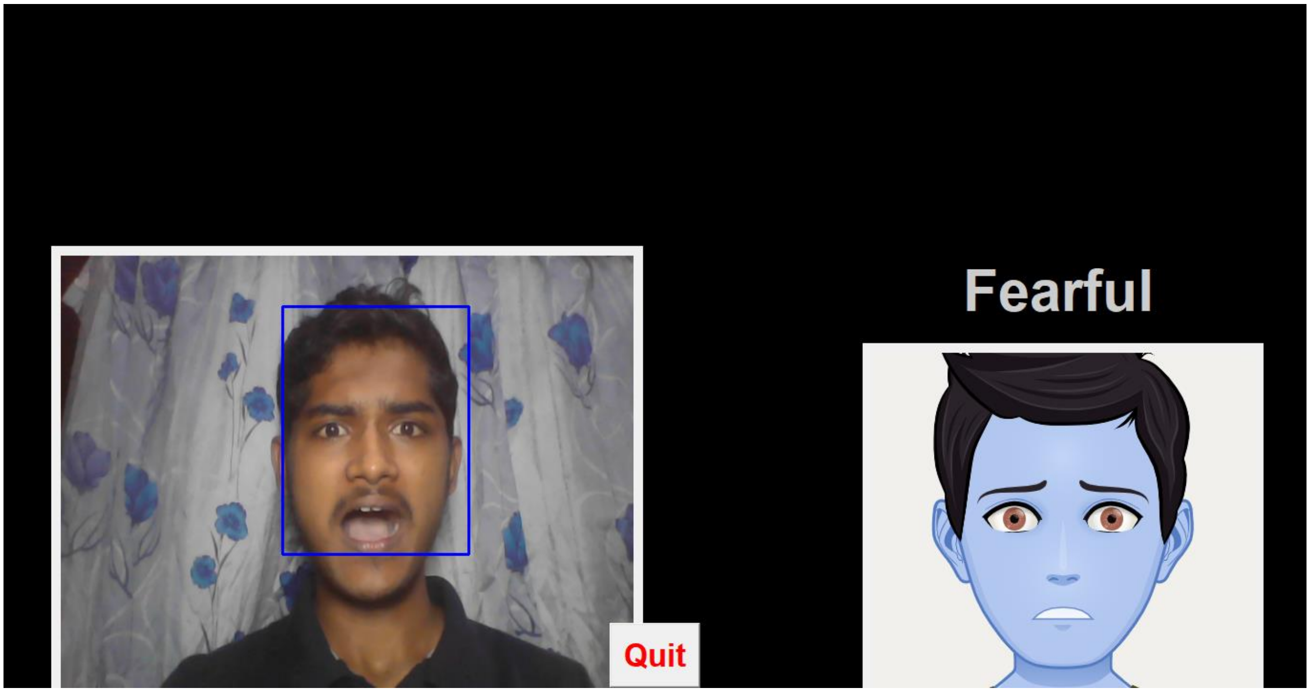
Output Sample 4



Output Sample 5



Output Sample 6



6. CONCLUSION:

The facial expression recognition system presented in this mini project contributes a resilient face recognition model based on the mapping of behavioral characteristics with the physiological biometric characteristics. The physiological characteristics of the human face with relevance to various expressions such as happiness, sadness, fear, anger, surprise and disgust are associated with geometrical structures which restored as base matching template for the recognition system.

The behavioral aspect of this system relates the attitude behind different expressions as property base. The property bases are alienated as exposed and hidden category in genetic algorithmic genes. The gene training set evaluates the expressional uniqueness of individual faces and provide a resilient expressional recognition model in the field of biometric security.

The design of a novel asymmetric cryptosystem based on biometrics having features like hierarchical group security eliminates the use of passwords and smart cards as opposed to earlier cryptosystems. It requires a special hardware support like all other biometrics system. This research work promises a new direction of research in the field of asymmetric biometric cryptosystems which is highly desirable in order to get rid of passwords and smart cards completely. Experimental analysis and study show that the hierarchical security structures are effective in geometric shape identification for physiological traits.

7. REFERENCES:

- https://www.geeksforgeeks.org/convolutional-neural-network-cnn-in-machine-learning/
- https://www.baeldung.com/java-cnn-deeplearning4j
- https://data-flair.training/blogs/deep-learning/
- https://www.kaggle.com/msambare/fer2013