

VISVESVARAYA TECHNOLOGICAL UNIVERSITY

Jnana Sangama, Belagavi – 590 018.



A PROJECT REPORT

on

"EMOJIFIER"

Submitted in partial fulfillment of the requirement for the award of the degree

Bachelor of Engineering

in

Computer Science and Engineering

by

RANJITH SINGH	:	1VI19CS081
ROHAN TELKAR	:	1VI19CS086
SANTHOSH S	:	1VI19CS092
SYED MOHAMMED IMAD	:	1VI19CS113

Under the supervision of

Ms. Rachitha M V
Assistant professor



DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING
VEMANA INSTITUTE OF TECHNOLOGY

BENGALURU – 560 034

2022 - 23

Karnataka ReddyJana Sangha®

VEMANA INSTITUTE OF TECHNOLOGY

(Affiliated to Visvesvaraya Technological University, Belagavi)

Koramangala, Bengaluru-34.



**Department of Computer Science and
Engineering**

Certificate

This is to certify that the project (phase-II) entitled “**EMOJIFIER**” is a bonafide work carried out jointly by **Ranjith Singh (1VI19CS081)**, **Rohan Telkar (1VI19CS086)**, **Santhosh S (1VI19CS092)** and **Syed Mohammed Imad (1VI19CS113)** are bonafide students of **Vemana Institute of Technology** in partial fulfillment for the award of **Bachelor of Engineering in Computer Science and Engineering** of the **Visvesvaraya Technological University, Belagavi** during the year 2022-23. It is certified that all corrections/suggestions indicated for internal assessment have been incorporated in the report. The project report has been approved as it satisfies the academic requirements in respect of the project work prescribed for the said degree.

Supervisor

Ms. Rachitha MV

HOD

Dr. M. Ramakrishna

Principal

Dr. Vijayasimha Reddy. B. G

Submitted for the university examination (viva-voce) held on

External Viva

Internal Examiner

1. _____

2. _____

External Examiner

.....

.....

DECLARATION BY THE CANDIDATES

We the undersigned solemnly declare that the project report “EMOJIFIER” is based on own work carried out during the course of our study under the supervision of ‘Ms. RACHITHA M V’.

We assert the statements made and conclusions drawn are an outcome of our project work. We further certify that,

- a. The work contained in the report is original and has been done by us under the general supervision of my supervisor.
- b. The work has not been submitted to any other Institution for any other degree/diploma/certificate in this university or any other University of India or abroad.
- c. We have followed the guidelines provided by the university in writing the report.
- d. Whenever we have used materials (data, theoretical analysis, and text) from other sources, we have given due credit to them in the text of the report and their details are provided in the references.

Date:

Place:

Project Team Members:

Ranjith Singh	1VI19CS081
Rohan Telkar	1VI19CS086
Santhosh S	1VI19CS092
Syed Mohammed Imad	1VI19CS113

ACKNOWLEDGEMENT

First we would like to thank our parents for their kind help, encouragement and moral support.

We thank **Dr. Vijayasimha Reddy. B. G**, Principal, Vemana Institute of Technology, Bengaluru for providing the necessary support.

We would like to place on record our regards to **Dr. M. Ramakrishna**, Professor and Head, Department of Computer Science and Engineering for his continued support.

We would like to thank our project coordinators **Mrs. Mary Vidya John**, Assistant Professor and **Mrs. J Brundha Elci**, Assistant Professor, Dept. of CSE for their support and coordination.

We would like to thank our project guide **Ms. Rachitha M V**, Assistant Professor, Dept. of CSE for her continuous support, valuable guidance and supervision towards successful completion of the project work.

We also thank all the Teaching and Non-teaching staff of Computer Science and Engineering Department, who have helped us to complete the project in time.

Project Team Members:

RANJITH SINGH	(1VI19CS081)
ROHAN TELKAR	(1VI19CS086)
SANTHOSH S	(1VI19CS092)
SYED MOHAMMED IMAD	(1VI19CS113)

ABSTRACT

Emojis are small images that are commonly included in social media text messages. The combination of visual and textual content in the same message builds up a modern way of communication. Emojis or avatars are ways to indicate nonverbal cues. These cues have become an essential part of online chatting, product review, brand emotion, and many more. It also led to increasing data science research dedicated to emoji-driven storytelling. With advancements in computer vision and deep learning, it is now possible to detect human emotions from images. The core components of the Emojifier system include robust facial expression classification algorithms, natural language processing techniques for text-based emotion analysis, and sophisticated speech recognition algorithms for emotion detection from speech inputs. The fusion of these components enables the Emojifier to accurately interpret and map a diverse range of human emotions to appropriate visual representations. In this project, we will classify human facial expressions to filter and map corresponding emojis or avatars. Emojifier is a software which deals with the classification of facial expressions, text and speech into Emojis or Avatars.

Keywords: Facial expression classification, Emoji-driven storytelling, Computer vision and deep learning, Emojifier software, Human emotion detection.

TABLE OF CONTENTS

Chapter No.	Title	Page No.
	Acknowledgement	i
	Abstract	ii
	List of Tables	iii-v
	List of Figures	vi
	List of Abbreviations	vii
	List of Symbols	viii
1.	INTRODUCTION	01-07
	1.1 Scope	03
	1.2 Objectives	04
	1.2.1 Plan of action for the project	05
	1.2.2 Current status of project	05
	1.2.3 Proposed plan for completion	06
	1.2.4 Outline of the chapters	06
2.	LITERATURE SURVEY	08-21
	2.1 Literature survey 1	08
	2.2 Literature survey 2	09
	2.3 Literature survey 3	10
	2.4 Literature survey 4	11
	2.5 Literature survey 5	12
	2.6 Literature survey 6	13

	2.7 Literature survey 7	14
	2.8 Literature survey 8	15
	2.9 Literature survey 9	17
	2.10 Literature survey 10	18
	2.11 Comparative Analysis	18
3.	SYSTEM REQUIREMENTS	22
	3.1 Functional Requirements	22
	3.2 Non-Functional Requirements	22
	3.3 Hardware Requirements	22
	3.4 Software Requirements	22
4.	DESIGN METHODOLOGY	23-24
	4.1 Dataflow Diagram	23
	4.2 System Architecture	24
5.	MODULE DESCRIPTION	25-29
	5.1 Module 1: Expression to Emoji	25
	5.2 Module 2: Text to Emoji Converter	27
	5.3 Module 3: Speech to Emoji converter	28
6.	SYSTEM IMPLEMENTATION	30-37
	6.1 Introduction	30
	6.2 Use case diagram	32
	6.3 Screenshots	33
7.	SYSTEM TESTING	38-39

	7.1 Test cases	38
	7.2 Tests conduction	39
8.	SUMMARY	40
	CONCLUSION AND	
	FUTURE WORK	41
	REFERENCES	42
	APPENDIX A	43
	APPENDIX B	51

LIST OF FIGURES

Fig. No.	Title	Page No.
1.1	Emoji Categories	01
1.2	Emoji face stimuli	04
4.1	System Architecture	23
4.2	Data Flow Diagram	24
5.1	Data flow diagram for module 1	26
5.2	Data flow diagram for module 2	27
5.3	Data flow diagram for module 3	29
6.1	Use case diagram	32
6.2	Home page	33
6.3	providing input to speech	34
6.4	Acquiring the input given through speech	34
6.5	Displaying emoji for speech input	35
6.6	Acquiring the input given through text	35
6.7	Displaying emoji for text input	36
6.8	Acquiring the input given through real time image	36
6.9	Acquiring the input given through real time image	37

LIST OF TABLES

Table No.	Title	Page No.
1.2	Time Line Chart	05
2.1	Comparative Analysis	19-21
7.1	Checking for Redundancy	38
7.2	User Interface	38

LIST OF ABBREVIATIONS

Acronyms

Abbreviations

AR	Augmented Reality
API	Application Programming Interface
Bi-LSTM	Bi-directional long term short memory
CNN	Convolutional Neural Network
GPU	Graphical Processing Unit
ICML	International Conference On Machine Learning
ML	Machine learning
NLP	Natural Language Processing
Open CV	Open source computer vision
OST	Optical See - Through
RNN	Recurrent neural network
TPU	Tensor processing unit
VS code	Visual studio code

CHAPTER 1

INTRODUCTION

Emojis are the modern-day equivalent of emoticons. The necessity for them developed from the difficulty of conveying emotions and expressions in written form. The Unicode Consortium standardized these two-dimensional visual representations of commonplace things as part of Unicode 6.0 in 2010. Emoji spread rapidly over the world, becoming an integral component of Western popular culture in particular. Practically every network and messaging app now uses it. One of the most important functions of emojis in internet communication is the expression of emotion. The number of emojis has grown to 2,666 as of June 2017, the latest data available on Emojipedia. This presents difficulties for programs that list them on portable devices with limited screen real estate, such as mobile phones. To get around this problem, most devices' emoji keyboards have the classifications shown in Fig. 1.1.

























Category	Emoji Examples
Smiley and People	 ,  , 
Animals and Nature	 ,  , 
Food and Drink	 ,  , 
Activity	 ,  , 
Travel and Places	 ,  , 
Objects	 ,  , 
Symbols	 ,  , 
Flags	 ,  , 

Fig. 1.1: Emoji Categories

The use of emojis, which are standardized sets of small graphical glyphs showing anything from smiling faces to foreign flags and were first presented in 1997, has witnessed a significant surge in usage in social media over the past decade. Emojis were first introduced in 1997. The 'Face with Tears of Joy' emoji was selected as the Oxford Dictionary's Word of the Year for 2015. This was due to the fact that the use of emojis increased by more than 800% during the course of the year. The Oxford Dictionary declared 2015 the year of the emoji. Over ten percent of the postings on Twitter and more than half of the content on Instagram contain at least one emoji. Because of their widespread adoption and popularity, they have been the focus of a significant amount of scholarly and non-scholarly

investigation in the fields of language and social communication, as well as natural language processing (NLP).

The lack of manually annotated data hinders the progress of many natural language processing jobs. As a result, models have been using co-occurring emotional expressions for distant supervision in social media sentiment analysis and related tasks in order to acquire helpful text representations before modelling these tasks directly. For instance, cutting-edge techniques for analyzing social media data rely on a library of positive and negative emoticons to teach their algorithms how to interpret user feedback. Similarly, prior studies have mapped hashtags like #angry, #joy, #happytweet, #ugh, #yuck, and #fml into emotional categories for study.

In the field of social sciences, a significant amount of attention has been paid to the practice of employing emojis on mobile platforms as a means of conveying one's emotional state. Emojis are commonly regarded as a means of conveying emotions; however, recent research has shown that they are also being used as tools to express relationally helpful roles in discussion. This finding is rather intriguing. Emojis have been shown to be culturally and contextually bound, and it has been demonstrated that this leaves them subject to reinterpretation as well as misinterpretation. These discoveries have led the way for several formal examinations of the semantic features of emojis.

In many cases, a model can achieve higher performance on the goal task with distant supervision. In this research, we demonstrate that improving the models' performance on benchmarks for identifying sentiment, emotions, and sarcasm may be achieved by expanding the distant supervision to a more diverse set of noisy labels.

Concurrently, we have seen a rise in interest in the application of natural language processing to data derived from social media. Many of the contemporary NLP systems that are applied to social media rely on representation learning and word embeddings. Such systems frequently rely on pre-trained word embeddings, which can, for example, be obtained via word2vec or GloVe. These two resources are both available online. However, neither resource has the full set of Unicode emoji representations, which leads one to believe that a great number of social NLP applications could be enhanced by the incorporation of more robust emoji representations. In this work, we announce the availability of emoji2vec, which is an embedding for emoji Unicode symbols that was

learned from the description of those symbols in the Unicode emoji standard. In addition, we offer a qualitative study by researching some examples of emoji analogies and visually representing the area in which emojis can be embedded.

The use of emojis is one approach to convey nonverbal messages. These indicators have rapidly become an indispensable component of a wide variety of activities, including online talking, product reviews, brand emotions, and many others. Additionally, it resulted in an increase in the amount of data science research that was devoted to emoji-driven storytelling. This technology allows us to grasp the emotions and sentiments of a person without having to focus on their face and can also be used to generate image filters. It does both of these things by utilizing a neural network. The CNN recognition technique was utilized in the training process for the Machine Learning model.

1.1 SCOPE

Emotional and semantic emoji evolved from emoticons. Individual conditions, culture, and platforms affect emoji use. Ambiguity and misunderstanding can emerge in different cultural contexts. From the perspectives of many fields (communication, computing, behavioral science, marketing, and education), this project comprehensively combs the research topics, methods, and tools used in emoji studies, systematically summarizes the research status of emoji in various fields, and proposes new perspectives for future emoji research, such as emotional association, use preference, new modalities, social impacts. and sentiments of a person without having to focus on their face and can also be used to generate image filters. It does both of these things by utilizing a neural network.



Fig. 1.2 : Emoji face stimuli

1.2 OBJECTIVES

- To open up conversations about emotion recognition systems: from the science behind the technology to their social impacts.
- To develop a model using filtered data gives better predictions.
- To promote public understanding of these technologies and citizen involvement in their development and use.
- To bring public understanding and awareness in the development of the emotion recognition systems and their social impact.
- To identify the intensity of pain of a deaf patient. For example: It can be used by Video conferencing doctors.

1.2.1 Plan of action for the project

Table 1.1 : Timeline Chart

Task	October	November	December	January	February	March	April	May
Problem Formulation								
Synopsis Submission								
Research and Tools procurement								
Building Prototype								
Testing the prototype								
Customizing the prototype								
Final Testing of prototype								
Report Submission								

Stage 1 – In this stage, we done planning of doing conversion of Facial Expression To Emoji, Speech To Emoji, Text To Emoji ML project.

Stage 2 – In this stage, the team identified the software requirements for the project, drafted a plan and collected 4 reference papers to gain information that is required to start the project.

Stage 3- In this stage, the team gained knowledge on neural networks and how to use them. Artificial Neural networks for raw data and Convolutional Networks.

Stage 4 - The project was proposed to the college and reviewed by the panelists.

Stage 5 – In this stage, we started with the documentation work and drafted the report for project phase-1.

Stage 6- Developing a working prototype and check the initial accuracy.

Stage 7- Customizing the prototype based on the results we get during the initial test.

Stage 8- Final testing is done after implementing all the changes

Stage 9- Submitting the final report of the project.

1.2.2 Current status of project

The deadlines set to complete the project phase-1 is end of December. As mentioned earlier, the project phase-1 consists of devising a problem statement, analyzing the requirements,

and drafting the report, this is to be successfully completed by December. The initial framework is to be completed by the end of December. This gives us more time to focus on the implementation of the main project so that we can get better and accurate results as we planned to spend four months to build the project.

1.2.3 Proposed plan for completion

- **Data collection:** Storing the facial expressions from premade datasets like FER-2013 and also by adding our own facial expressions, using it as a training set to predict Emojis.
- **Data Pre-processing:** This was converted into a dictionary format where the dictionary key is the year and, the data in step 1 was combined with data of this

Data Cleaning: The CSV file created in step 2 was cleaned to remove null values and improper data. A new resultant CSV file was created.
- **Feature Engineering and Model Creation:** Tried various algorithms, like Convolutional Neural Network, Bi-directional Long short term memory, HAAR cascade classifier gave best performance.

1.2.4 Outline of the chapters

- **Chapter 1:** Introduction - In this chapter we have a brief introduction to the project. It includes scope of the problem statement and the objectives of the project like plan of action, current status of project and proposed plan for completion of project.
- **Chapter 2:** Literature Survey - In this chapter we look at different survey/reference papers that we have considered that has helped for completion of our project. We have identified the advantages and limitations of each survey paper. This includes a comparative analysis of the papers to understand the how each proposal is different from the other.
- **Chapter 3:** System Requirements - In this chapter we have determined all the requirements of the project that include, functional, non-functional, hardware and software requirements.
- **Chapter 4:** Design methodology - In this chapter we understand the system architecture and the dataflow diagrams to visualize the workflow.
- **Chapter 5:** Module Description - In this chapter we elucidate the working of each module that is built to develop the system.

- **Chapter 6:** System implementation - In this module we include the Use case diagram and the different algorithms what we have used and the outputs of the code.
- **Chapter 7:** System Testing – In this module we include the different test cases performed and the test conducted with respect to different algorithms.
- **Chapter 8:** Summary- In this chapter we have summarized all the work that we have done in the first phase of the project.

CHAPTER 2

LITRERATURE SURVEY

[2.1] Emotion Recognition Based Emoji Retrieval Using Deep Learning by Swati Srivastava , Prateek Gupta and Pravendra Kumar Department of Computer Engineering & Application GLA University Mathura, India.

Here Swati Srivastava et al. have proposed a deep learning model to classify facial expressions from the images. Then mapped the classified emotion to a corresponding emoji or an avatar. They built a convolution neural network to acknowledge facial emotions. They trained their model on the FER2013 dataset. Then mapping of those emotions with the corresponding emojis or avatars have been performed. Using OpenCV's HAAR cascade xml, they obtained the bounding box of the faces within the webcam. Then they feed these boxes to the trained model for classification. In this will build a convolution neural network structure and train the model on FER2013 dataset for Emotion recognition from images. The FER2013 dataset (facial expression recognition) consists of 48*48 pixel grayscale face images. The images are centered and filled with an equal amount of space. This dataset consists of facial emotions of following descriptions; angry, disgust, fear, happy, sad, surprise, natural. The training set consists of 25000+ images and the public test set consists of 3500 images.

Here it uses image processing techniques such as reshape, resizing, converting to greyscale, and standardization. NumPy and pandas library are used for implementation. With the help of power vectorization array and pandas data frames, images have been processed whenever required the images with the help of NumPy arrays and generate the output by pandas. Preprocessing results in an improved image that suppresses undesired distortions or enhances some image features relevant for further processing and analysis task. First they have reduced the noise of images with the help of auto-encoding algorithm which help in reduction of unwanted data. They have used image segmentation through non-contextual thresholding and contextual segmentation so that it can convert part of an image into multiple parts and separating foreground from background so that emotion on the face in an image can be easily detected. With geometric transformation, positions of pixels in an image are modified.

Methodology : Here they developed an improved emotion recognition model for Facial

expression images. It uses a convolutional neural network based deep learning model which is tested on FER2013 dataset.

Advantages: The proposed model has a accuracy of 87%. The image detection is really fast with a very high success rate. This proposed model which will detect the facial expressions and map those emotions on the faces with the corresponding emojis or avatars.

Disadvantages: It only works For Still images and does not give accurate results for motion pictures/videos. The efficiency involved in the preprocessing steps is not up to the mark.

[2.2] Emojify –Create your own emoji with Deep Learning by Payas Doshi, Priyanshi Sethi, Pulkit Sharma, Mahaveer Jain International Research Journal of Engineering and Technology.

Here Payas Doshi et al. have built a convolution neural network to recognize facial emotions. It had trained their model on the FER2013 dataset. Then we will map those emotions with the corresponding emojis or avatars. Fer2013 contains approximately 30,000 facial RGB images of different expressions with size restricted to 48×48, and the main labels of it can be divided into 7 types: 0=Angry, 1=Disgust, 2=Fear, 3=Happy, 4=Sad, 5=Surprise, 6=Neutral. The Disgust expression has the minimal number of images – 600, while other labels have nearly 5,000 samples each. In this deep learning project, they had classified human facial expressions to filter and map corresponding emojis or avatars. Created a folder named emojis and saved the emojis corresponding to each of the seven emotions in the dataset. The trained model is tested on a set of images. Random images are introduced to the network and the output label is compared to the original known label of the image. Parameters used for evaluation are F1 score, precision and recall. Precision is the proportion of predicted positives that are truly positives.

Methodology:

1. Build a CNN architecture. Here It is importing all the required libraries needed for their model and then they are initialising the training and validation generators i.e., they are first rescaling all their images needed to train their model and then converting them to grayscale images.
2. Train the model on Fer2013 dataset. Here they are training their network on all the images they have i.e., FER2013 dataset and then saving the weights in model for the future predictions. Then using OpenCV harassed xml to detect the bounding boxes of face in webcam and predict the emotions.

Advantages: It is designed to receive voice commands, reduce typing difficultly, and

provide output in both text and voice formats. The predicting the emoji based on the emotion at an accuracy of 90%. use of emojiify in chatting world. people can communicate with their own customisable emoticon. This project will recognize one's current emotion and convert that emotion's emoji so that the customer gets emoji of their face and use it in chatting.

Disadvantages: The Graphical user interface is not user friendly and the accuracy attained is not very high. It can only be used with images.

Tools Used: Google Collab supports both GPU and TPU instances, which makes it a perfect tool to train the model on large datasets. It also has various data science related libraries like koralas, TensorFlow, OpenCV, matplotlib, NumPy etc. For the purpose of building the keras model they have used sequential modelling technique. VS Code is used for overall development as a standard platform.

[2.3] Anon-Emoji: An Optical See-Through Augmented Reality System for Children with Autism Spectrum Disorders to promote Understanding of Facial Expressions and Emotions by Ran Sun Harald Haraldsson¹ from Cornell Tech Yuhang Zhao Serge Belongie from Cornell University.

Based on their observations, Ran Sun Harald et al. have designed the system of A non Emoji. The system consists of a head tracker that the subject wears on the head and a system running on the Magic Leap One headset. When the headset picks up the head tracker either through image tracking or electromagnetic tracking, The system tracks the position and orientation of the person's head and renders an emotion presenting 3D emoji around it so as to occlude his/her face. They chose emojis for two main reasons: its nature in expressing non-verbal information; its strength in concealing effects under OST AR settings. The emotional information they bear is widely recognized and easy to comprehend. As they are in the rough shape of a head and make sense to the viewers when overlayed on the head of a subject. Finally, emojis are generally bright yellow in color, which their experience tells us is ideal for occluding objects in OST AR displays. In the future, the authors will implement a separate process runs in the background on the device which aims to perform facial landmark detection and sentimental analysis to extract the facial expression of the subject, map it to a certain emoji, and change the renderings.

Methodology: The system consists of an AR headset and a head tracker. Viewers are required to wear the AR headsets, while subjects only wear head tracker around their heads. The head tracker helps identify the position and rotation of the subject's head. Once the

tracker is picked up by the system, a virtual rendering of 3D emoji model will be rendered around the subject's head. The emoji model will obscure most visual information of the subject's face from all directions.

Advantages: It can be used for kids with Autism disorder and help them learn better through use of emojis. The system tracks the position and orientation of the person's head and renders an emotion presenting 3D emoji around it so as to occlude(close up) his/her face.

Disadvantages: OST headsets are designed for indoor purposes. The design would work well only in indoor settings without strong light coming through the headset. Although the general intention of AR(Augmented reality) is to add objects to a real environment, it can also remove objects. However, including real world objects is not one of the strengths of OST AR displays.

Tools used: Magic Leap One Headset, OS version 0.95.2, MLSDK 0.20.0, Unity, OpenCV for Unity.

[2.4] Facial Emoji Recognition by N. Swapna Goud(Assistant Professor), K. Revanth Reddy, G. Alekhya, G. S. Sucheta CSE Department, Anurag Group of Institutions, Telangana, India.

1. Decomposing an image.

Images are composed of pixels and these pixels are nothing more than numbers. Often it is considered that the Coloured images can be divided into three colour channels, which are :red, green, and blue and each channel is represented by a grid (2-dimensional array). Each cell in the grid stores a number between 0 and 255 which denotes the intensity of that cell.

2. Importing Necessary Libraries

The libraries like Numpy, Pandas, Matplotlib and SKlearn are imported in this project.

3. Define Data Loading Mechanism

It had defined the load_data() function which will efficiently parse the data file and extract necessary data and then convert it into a usable image format. All the images in their dataset are 48x48 in dimensions. Since these images are gray-scale, there is only one channel. It will extract the image data and rearrange it into a 48x48 array. Then convert it into unsigned integers and divide it by 255 to normalize the data. 255 is the maximum possible value of a single cell and by dividing every element by 255, It ensures that all values range between 0 and 1. They have checked the Usage column and store the data in separate lists, one for training the network and the other for testing it.

4. Defining the model.

They have used Keras to create a Sequential Convolutional Network. Which means that neural network will be a linear stack of layers.

5. Callback functions

Callback functions are those functions which are called after every epoch during the training process.

6. Training the model

The Model has been trained based on the Datasets provided.

7. Test the model

The above model is started off by defining a loading mechanism and loading the images. Then a training set and a testing set are created. After this a fine model and a few callback functions are defined. The basic components of a convolutional neural network are considered and then they training is done to the network.

Methodology: Facial expressions can be described as the arrangement of facial muscles to convey a certain emotional state to the observer in simple words. Emotions can be divided into six broad categories—Anger, Disgust, Fear, Happiness, Sadness, Surprise, and Neutral. Train a model to differentiate between these, train a convolutional neural network using the FER2013 dataset and will use various hyper-parameters to fine-tune the model.

Advantages: It can be used to denote a person's stress level in a hassle free way. Robust spontaneous expression recognizers in the model can be developed and deployed in real-time systems.

Disadvantages: This works only for images and can not classify text or speech like their project.

[2.5] Real-Time Facial Expression Recognition Based on CNN by Keng-Cheng Liu, Chen-Chien Hsu, Wei-Yen Wang, Hsin-Han Chiang Department of Electrical Engineering National Taiwan Normal University Taipei, Taiwan.

In this paper, Keng-Cheng Liu et al. have proposed method aims to improve the accuracy of real-time facial expression recognition. The method is divided into three steps:

- 1) obtain and input the images to the CNN architecture for recognition;
- 2) obtain the results of facial expression recognition;
- 3) average the recognition result with the previous output and re-output.

In this way, real-time recognition can be achieved through integrating the CNN architecture

and the webcam. Moreover, overall robustness and accuracy of facial expression recognition are greatly improved by using the proposed method compared to those obtained by only the convolution neural network (CNN).

For face detection, they used a Logitech C920 webcam with the Open CV face capture program and the parameters from the face training set `haarcascade_frontalface_alt.xml`. The detection was programmed in Python on the Anaconda platform to achieve a detection rate of about 10 images per second. This step was followed by grayscale conversion and resizing for each image to immediately and quickly input the captured images into the trained framework.

Methodology: They derived from the average of the current recognition result and the previous recognition result because the system considers both the CNN architecture and the influence of the environment of the captured image. Each image has 7 weights indicating different expressions, which are recorded and averaged with the weights of the previous image. Subsequently, the largest output is therefore regarded as the prediction result. It is worth noting that the proposed average weighting method is used when the weights are produced from the second identification. Because the weights are averaged between the current and the prior recognition, the error is accordingly reduced. The accuracy of the original CNN architecture complements each other to enhance the overall performance. Furthermore, if a webcam is employed, an effective real-time facial expression recognition system can therefore be achieved.

Advantages: Because of the average weighting method, facial expression recognition results become more robust from frame to frame. As a result, the accuracy of facial expression recognition is improved. Experimental results have shown that the proposed facial expression recognition system is more reliable than the traditional CNN approach. Max pooling is employed for the pooling of each layer.

Disadvantages: This model has less accuracy and low quality image recognition.

[2.6] Exploiting Deep Neural Networks for Tweet-based Emoji Prediction

by Andrei Catalin Coman, Giacomo Zara¹, Alessandro Moschitti
University of Trento, Trento, Italy and Yaroslav Nechaev , Gianni
Barlacchi Fondazione Bruno Kessler, Trento, Italy.

The model CNN has been used to solve various types of problems. One of the first applications of this neural-architecture was in the visual domain, where It is employed for image and video recognition. However, the versatility of the CNN has led them to be used

in Recommendation Systems and even in Natural Language Processing tasks. In contrast to classic Multilayer Perceptron Neural Networks, CNNs have distinguished themselves through three essential characteristics, namely:

Number of parameters to train: in classic neural network architectures, i.e. those with a lot of dense hidden layers and units, the number of parameters to be trained grows really fast. With CNNs instead, the amount of train parameters is given by the size and quantity of convolution filters, which in some cases are drastically less.

Parameters sharing: a feature detector that's useful in one part of the sentence is probably useful in another part of the sentence.

Sparsity of connections: In each convolution layer, each output value is depends only on a small number of inputs.

Methodology: In present Neural Network models Andrei Catalin Coman et al. used to perform their emoji predictions. The first model is an architecture based on RNNs, where the recurrent cell consists of a Bi-LSTM. As second model they decided to implement a CNN, since over the years it has proved to be effective in the Natural Language Processing area. Some other concepts are also exposed including the last layer used for predictions, the loss function applied for the train, the regularization techniques employed to contain overfitting and finally the strategy adopted to stop the train.

Advantages: The main advantage is it uses FastText embeddings over Word2Vec is to take into account the internal structure of words while learning word representations, which could be very useful for morphologically rich languages, and also for words that occur rarely. The Deep learning model can tackle problems that traditional machine learning models cannot.

Disadvantages: Not much flexibility. No option to speed up using GPU. Can be used only for text classification and word embeddings. The Deep learning works only with large amounts of data.

[2.7] Transformation of Facial Expression into corresponding Emoticons by Ankur Ankit, Dhananjay Narayan, Alok Kumar International Journal of Engineering and Advanced Technology (IJEAT)

The proposed model will detect a face using API and feature extraction is done through HAAR Cascade. Emotions are classified from the extractions through SVM. The Emojis are later superimposed over the faces according to the matching emotion exhibited by the subject.

API implementation: An API acts as an interface between an operating system, application and the user. The API design plays a significant role on its usage . An API is designed in such a way that it hides the background details of modules from the users who do not have the knowledge of complexity of the modules. Thus, API facilitates the user-friendly interface. A camera-based API can be used which automatically detects the face of the subject(s) regardless of the background and send this image to the model for processing after which the emoji will be superimposed over the face.

The image that is supplied by the API is then provided to the HAAR cascade in which some dataset has been given for training the data. For the development of a working model, they will use two datasets: Cohn-Kanade (CK+) and Japanese Female Facial Expression (JAFPE) . HAAR-Like features have high accuracy to detect faces from different angles.

Support vector machine: Support Vector Machine is a supervised machine learning algorithm that is used for classification as well as regression problems. The SVM is used in many pattern analysis tasks with support of binary classifier that differentiates between the classifications of the expressions. It works by classifying data through the use of assessment of an optimal hyper plane which separates one class's data points from the other.

Methodology: The idea of the proposed system is to employ an API that will detect the face after which the image can be processed using HAAR cascade for facial feature extraction. SVM Classifier is then used to categorize the emotions into its seven distinct types. Using HAAR of OpenCV package, the corresponding emojis of the emotions can get superimposed over the subjects' faces. In any camera module of any leading social networking apps, the use of APIs can reduce the processing time for face detection for which they have their in-built face detection algorithm which can detect the face smoothly and followed by which the emoticons can be implemented over the faces as filters.

Advantages: very fast at computing features due to the use of integral images. It is also very efficient for feature selection through the use of the AdaBoost algorithm. It works really well with a clear margin of separation. It is effective in high dimensional spaces.

Disadvantages: The downside is that it tend to be prone to false-positive detections. It doesn't perform well when they have large data set because the required training time is higher.

[2.8] Mapping of human facial expressions to emojis using Deep learning by Parth Vishe, Student, Dept. of Electronics and Telecommunication.

Engg, Pune Institute of Computer Technology, Pune, Maharashtra, India.

Input the dataset: The sample images from the FER 2013 dataset that are used to classify emotions. These Images are categorized based on the emotion shown in the facial expressions such as happiness, neutral, sadness, anger, surprise, disgust, fear.

Data pre-processing and applying augmentation Strategies: Image data augmentation is used to expand the training dataset in order to improve the performance and ability of the model to generalize. Images are rescaled from [0,255] to [0,1] using the Image Data Generator python module. Benefits of this are It treats all images in the same manner and some images are high pixel range while some are low pixel range. The images are all sharing the same model, weights and learning rate. The high range image tends to create stronger loss while low range creates weak loss and the sum of them will all contribute to the back propagation update. Using typical learning rate: When Parth Vishe et al. reference the learning rate from other's work, they can reference their learning rate directly if both works do the scaling preprocessing over images data set. Otherwise, higher pixel range image results in higher loss and should use a smaller learning rate, lower pixel range image will need a larger learning rate.

Neural Network architecture: After pre-processing the database, the next step is to build a convolutional neural network. The convolution layer consists of input layers, hidden layers, and output layers. Depending on the structure of the neural network add convolutional layers with filters.

Methodology: Their system treats all images in the same manner: some images are high pixel range while some are low pixel range. The images are all sharing the same model, weights and learning rate. The high range image tends to create stronger loss while low range creates weak loss and the sum of them will all contribute to the back propagation update .Using typical learning rate: When they reference the learning rate from other's work, they can reference their learning rate directly if both works do the scaling preprocessing over images data set.

Advantages: It possible to use the software on multiple GPUs or CPUs. The google colab used allows anybody to modify the code according to their will.

Disadvantages: No windows support. It has a very limited set of features for Windows users. The tensorflow used is comparatively slower and less usable compared to its competing frameworks. Google colab has a limited space and time.

[2.9] Emojify: Emoji Prediction from Sentence by Chen Huang, Xueying (Shirley) Xie, Boyu (Bill) Zhang Stanford University.

The model extracts 167,685 sentence emoji pairs for training. However, the dataset is heavily unevenly distributed, where emoji has over 50% representation power of the whole dataset. So they have done several data preprocessing steps before training. Noise removal, filter out emojis which has less than 10000 correspondence sentence. Stop emojis, like the stop words technique in NLP, remove high frequent emojis which are everywhere and do not have specific semantic meaning. Unbiased data, equalize the number of sentences for each emoji. After the data pre-processing, the dataset is reduced to 13251 valid examples which belongs to 13 emoji categories.

Word Embedding: BoW-TFIDF Instead of representing the sentences with matrix of corpus and counts like we've seen in class, Chen Huang et al. can employ Bag of Words representation with TD-IDF trying to capture more indicative keywords in a sentence. The dictionary size is 1834 after stop words and stemming. They can feed this into Their traditional methods for Naive Bayes and SVM.

Multinomial Naive Bayes Classifier: By using a word dictionary they transform texts into word vector via bag of words with TD-IDF as described above. Finally here they apply a Multinomial Naive Bayes Classifier to train the model for text classification. Since Naive Bayes is a simple classifier to use, it acts as a good baseline and starting point to tackle this problem.

Bi-LSTM Classifier: For the RNN model, the first layer is embedding layer, embedding layer will represent each word as a matrix. The embedding layer is loaded from a Pre-trained GLoVe model and the weight is fixed during training.

Methodology: Predicting emojis is more like a sentence classification problem. First investigate the word embedding techniques to use, then try machine learning algorithms such as Multinomial Naive Bayes, SVM and Bi-LSTM to do the emoji prediction.

Advantages: It is suitable for solving multi-class prediction problems. If its assumption of the independence of features holds true, the naive bayes can perform better than other models and requires much less training data. Naive Bayes is better suited for categorical input variables than numerical variables. It is easy to implement as they only have to calculate probability.

Disadvantages: The naive Bayes classifier used relies on an often-faulty assumption of equally important and independent features which results in biased posterior probabilities.

[2.10] Emojify-Create Your Own Emojis With Deep Learning by Sagar Chilivery, Sandeep Pukale, Yashraj Sonawane Vasantdada Patil Pratishthan's College Of Engineering And Visual Arts, India.

Here Sagar Chilivery et al. have notion of creating personal custom designed emojis. Emojify is a software program which offers with the advent of Emoji's or Avatars. The neural community has been an rising software in numerous regions as instance of cease to cease studying this paper primarily based totally on a gadget which implements Convolutional Neural Network and Fer2013 Dataset to hit upon feelings from facial expressions and changing them to personalized emojis. They are constructing a convolution neural community to apprehend facial feelings. It might be education version at FER2013 dataset. Then we'll map the ones feelings with the corresponding emojis or avatars. Fer2013 carries about 30,000 facial RGB pics of various expressions with length limited to 48×48, and the principle labels of it is labelled to be divided into five types: 0=Sad, 1=Surprise, 2=Fear, 3=Happy, 4=Neutral.

Methodology:

1. Build a CNN Architecture. Here uploading all of the required libraries wished for their version after which we're initializing the education and validation mills i.e., we're first rescaling all their pics had to teach their version after which changing them to grayscale pics.

2. Train the version on Fer2013 dataset. Here educating the community on all the pics they have got i.e. Fer2013 dataset after which saving weights in version for the destiny predictions. Then the use of OpenCV hit upon bounding containers of face in webcam and are expecting emotion.

Advantages: It's ability to execute feature engineering by itself. The predicting the emoji based on the emotion at an accuracy of 90%.

Disadvantages: A lot of training data is needed for it to be effective and they fail to encode the position and orientation of objects.

Tools used: They have used diverse records technology associated libraries like keras, TensorFlow, OpenCV, NumPy etc. For the motive of constructing the keras version they've got used sequential modelling technique. VS Code and Anaconda Prompt is used for usual improvement as a common platform.

2.11 COMPARATIVE ANALYSIS

Comparative analysis is the process of comparing items to one another and distinguishing their similarities and differences. When a business wants to analyze an idea, problem, theory

or question, conducting a comparative analysis allows it to better understand the issue and form strategies in response.

Table 2.1 Comparative Analysis

Reference	Algorithm/ Technique	Platform used	Performance Metrics	Advantage	Drawback
[2.1]	CNN Algorithm with Deep learning	Jupyter Notebook	Throughput	The image detection is really fast with a very high success rate.	It only works for Still images and does not give accurate results for Motion pictures/videos.
[2.2]	CNN Algorithm.	Google Collab, VS Code	Accuracy	It is designed to receive voice commands, reduce typing difficulty, and provide output in both text and voice formats.	The Graphical User Interface is not user friendly and the accuracy attained is not very high .It can only be used with images.
[2.3]	OST AR displays	OpenCV for Unity 2.3.4	Accuracy	It can be used for kids with Autism disorder and help them learn better through the use of emojis.	The design would work well only in settings without strong light coming through the headset.

[2.4]	CNN Algorithm.	Anaconda Platform	Robustness	It can be used to denote a person's stress level in a hassle free way.	This works only for images and can not classify text or speech like their project.
[2.5]	CNN Algorithm.	Anaconda Platform with OpenCV.	Accuracy	It denotes the robustness of facial emoji recognition.	This paper does not have a high accuracy
[2.6]	Bi-LSTM, CNN Algorithm , fastText.	Jupyter Notebook	Functionality	Simple deep neural networks can also reach quite good results by improving them through fine-tuning of hyper- parameters, regularization, and optimization of the models themselves.	The Deep Learning approach has generally out performed both the baselines.
[2.7]	SVM, HAAR	Google Collab, VS Code	Accuracy	Their approach of using APIs instead of neural networks, makes the implementation convenient.	The Accuracy is less compared to the existing deep learning models.
[2.8]	Multinomial Naive Bayes, LSTM	Anaconda Platform	Accuracy	Naïve Bayes performed verywell with a high accuracy of 95% of the text classification	This works only for text classification and not for image or speech classification.

[2.9]	CNN Architecture	TensorFlow, Keras , OpenCV, Google Collab,	Accuracy	It treats all images in the same manner some images are high pixel range while some are low pixel range. The images are all sharing the same model, weights and learning rate.	Achieves an accuracy of only about 60-65%. Does not work well on people with a beard.
[2.10]	CNN Algorithm.	Anaconda prompt, VScode	Accuracy	It denotes the robustness of facial emoji recognition.	This paper has less accuracy.

CHAPTER 3

SYSTEM REQUIREMENTS

3.1 Functional Requirements

Functional requirements for an emojiifier project are

Input Processing: The emojiifier should process the input text, analyze its content, and identify keywords, sentiments, or other relevant information that will be used to determine appropriate emojis.

Emotion Detection: The emojiifier should detect emotions expressed in the input, such as happiness, sadness, anger, or surprise.

Emoji Selection: Based on the analyzed input and detected emotions, the emojiifier should select suitable emojis to represent the content.

3.2 Non-Functional Requirements

- **Reliability:** Providing a reliable service for prediction of emojis.
- **Security:** The data will not be compromised as it will be stored on a securecloud service.
- **Performance:** Aiming to provide a model which is highly accurate.

3.3 Hardware Requirements

Processor :	INTEL Core I5
Ram :	8 GB
Processor Speed :	2.4 GHz
System Type :	64-bit/32-bit Operating System
Display :	1920*1080

3.4 Software Requirements

Programming Language :	PYTHON
Operating System :	Windows 10
Software :	Jupyter Notebook Using Python 3.7

CHAPTER 4

DESIGN METHODOLOGY

4.1 System Architecture

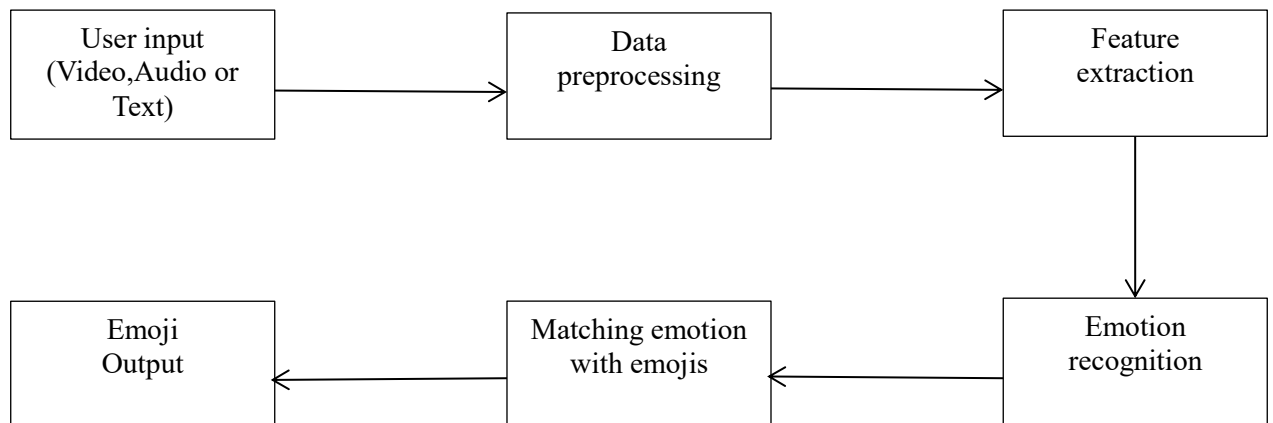


Fig. 4.1 - System architecture

The above fig. 4.1 shows that The architecture includes image capture, pre-processing, CNN training model, and emotion-to-emoji mapping. The camera captures the live picture, then image pre-processing suppresses the extracted live input image data and extracts features for subsequent processing. The saved model classifies the collected feature and detects emotion. The system also takes input in the form of text and speech. The speech input is pre processed with the help of NLTK and mapped to a dictionary containing the emojis. The text model works similarly, with the only difference being input is in text instead of voice.

4.2 Data Flow Diagram

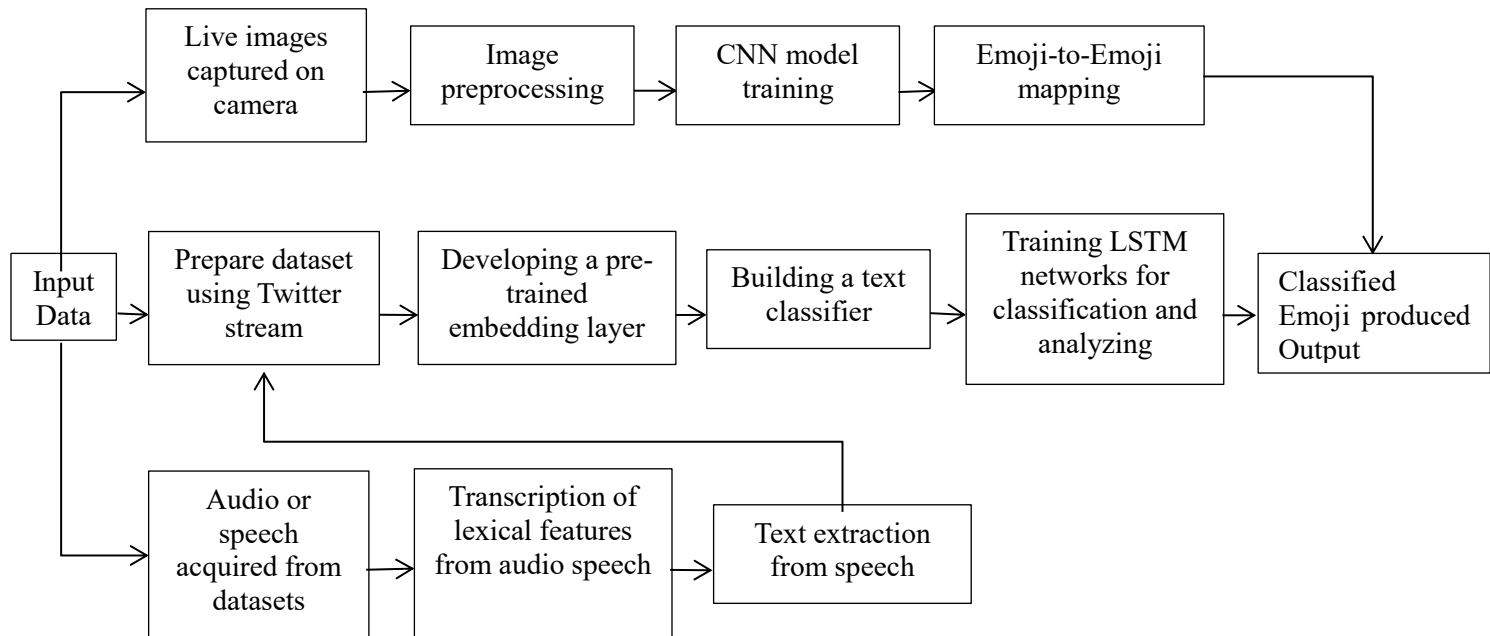


Fig. 4.2 – Data Flow diagram

The above Fig. 4.2 shows how the 3 different modules work in tandem with each other to give us the final emoji output. It shows how the data is converted from raw input, its features are extracted and finally classified across the 3 different modules to finally produce an output consisting of a single emoji.

CHAPTER 5

MODULE DESCRIPTION

Different Modules Of Emojifier

Module 1: Facial Expression To Emoji Converter

Module 2: Text To Emoji Converter

Module 3: Speech To Emoji Converter

5.1 Module 1: Facial Expression To Emoji Converter

The architecture includes image capture, pre-processing, CNN training model, and emotion-to-emoji mapping. The camera captures the live picture, then image pre-processing suppresses the extracted live input image data and extracts features for subsequent processing. The saved model classifies the collected feature and detects emotion. The identified emotion finds the matching emoji from the stored photos. Finally, the output emoji is shown and transmitted instantaneously when an emotion changes. OpenCV defines HAAR Cascade face detection. HAAR Cascade detects faces. A rectangle box captures the face. HAAR Cascade detects faces in each camera frame.

The trained neural network recognises facial expressions and classifies them into one of the seven fundamental emotions. It classifies input images into expressions (angry, sad, happy, surprise, disgust, fear and neutral). Python's Keras package simplifies CNN construction. Computers see pixels. Image pixels frequently relate. It maps user-provided live input image characteristics to emoji. Tkinter forms the GUI module. GUI applications use buttons, sliders, search fields, labels, and textboxes. Python's standard GUI is Tkinter. Open-sTheirce Python framework Streamlit. Python's standard Tk GUI toolkit interface is Tkinter. Python with Tkinter makes GUI apps fastest and easiest.

It is accomplished by capturing the face inside a rectangular box using Haar Cascade and the system is structured in such a way that the live picture is caught through the camera. The system is constructed in such a manner that the live picture can be captured. The steps that are to be followed are:

- i. The HAAR Cascade approach is applied to each each frame of the camera feed in order to detect faces.

ii. The portion of the image that contains the face is cropped to a resolution of 48 by 48 pixels before being sent as input to the CNN.

iii. After that, the image is preprocessed, which consists of suppressing the live input image data that was retrieved earlier and extracting features in preparation for subsequent processing.

- The identified feeling then looks through the collection of saved emoji for the one that best represents it.
- When a person changes the expression on their face, the emoji that corresponds to that alteration is instantaneously displayed, and it is also shared.
- The captured feature is trained, categorized, and then transmitted to the comparison model, which is where the emotion is detected from the saved model (i.e., there is already trained data there), and the results of the comparison are compared with those of the extracted features.
- This procedure is shown clearly in the below fig. 5.1.

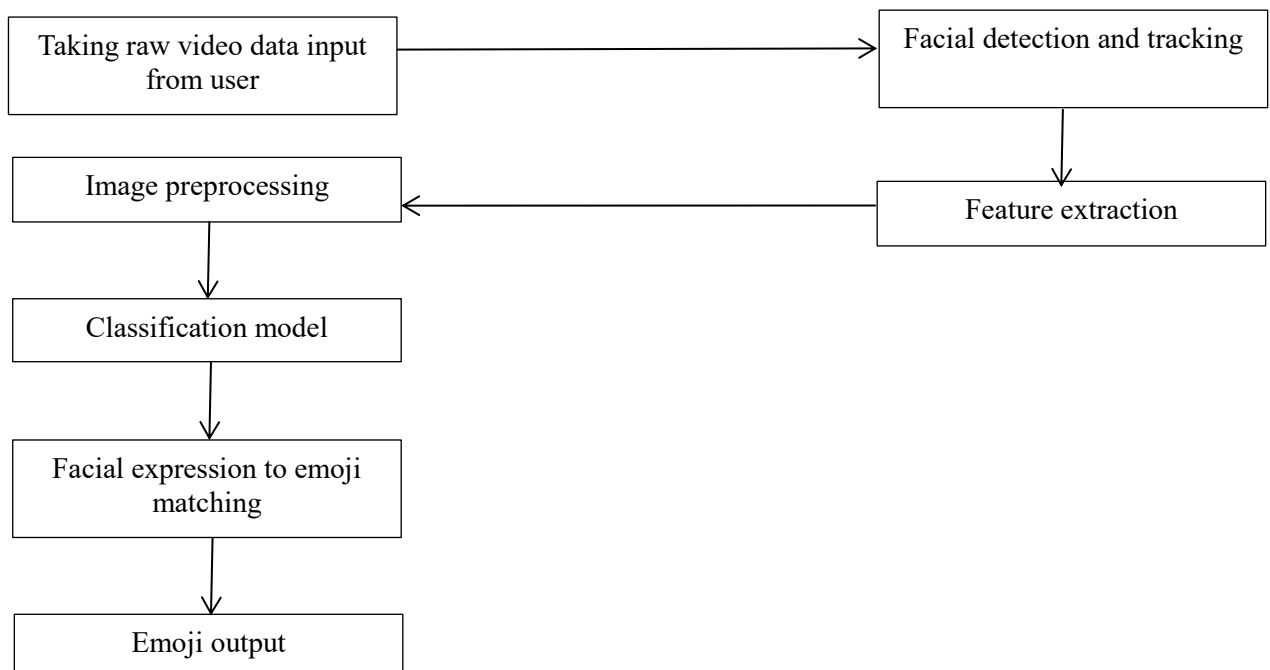


Fig. 5.1- Flow diagram for facial expression to emoji converter

5.2 Module 2: Text To Emoji Converter

This phase assists us in categorizing the text and provides an emoji in exchange. This means that we will construct a text classifier that, when presented with an English sentence, will make an appropriate emoji prediction. In order to prepare the dataset, we are going to use the Twitter streaming API. By using this, we gather both the tweets and the emoticons that go along with them. There are two.csv files in the folder containing the source code that are labelled train emoji.csv and test emoji.csv. These files allow us to download the dataset. The two files, train emoji.csv and test emoji.csv, contain the datasets that we will use to train and test the model that we develop. The layer, known as glove.6B.50d, functions primarily as a pre-trained embedding layer. It is going to be put to use in the process of embedding the words. Emoji Prediction Using Machine Learning is a python file that contains the entirety of the emoji prediction project's implementation . In light of the fact that we are going to construct the LSTM model. Because of this, gaining an understanding of it will be beneficial. The type of recurrent neural network known as long short-term memory, or LSTM, networks are able to learn order dependency in sequence prediction problems. Classifying, analyzing, and making predictions on the basis of text or time series data are all possible applications for LSTM networks. These steps are clearly shown in the below fig. 5.2.

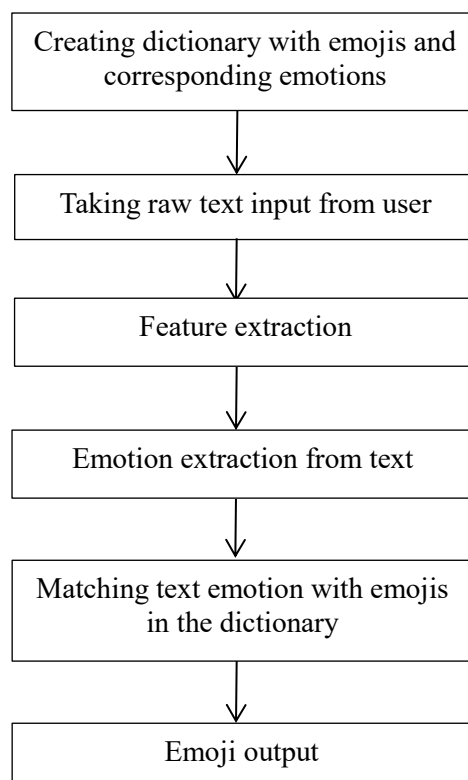


Fig 5.2 - Flow diagram for module text to emoji converter

5.3 Module 3: Speech To Emoji Converter

Humans naturally express themselves through words. We use emoticons to express their feelings in emails and text messages since we depend on it so much. In today's digital world of distant communication, detecting and analyzing emotions is crucial. Emotions are subjective, making emotion detection difficult. They're not standardized. Speech Emotion Recognition systems analyze and classify speech data to identify emotions.

An interactive voice-based assistant or caller-agent conversation analysis system can be used in many applications. This study analyses the acoustic aspects of recorded speech to discover underlying emotions. Speech has lexical, visual, and auditory aspects (sound properties like pitch, tone, jitter, etc.).

Speech to emoji converters typically use natural language processing (NLP) and machine learning techniques to analyze the speech and determine the most suitable emojis to represent the content. These tools often rely on pre-built databases or models that associate specific words or phrases with corresponding emojis.

These time audio, following lexical features requires a transcript and text extraction from speech. Analyzing visual aspects would require access to the video of the discussions, which may not be possible in all cases, but we can analyze features in real time using simply the audio data. This paper analyses textual features. Emotions can be shown in two ways:

- 1) Classifying emotions as anger, happiness, boredom, etc.
- 2) Representing emotions with emojis like sad, happy, angry, laughing etc.

The dimensional technique is more complex and provides more context for prediction, but it is harder to execute and lacks dimensional audio data. Dimensional representation gives predictive context, yet discrete categorization is simpler and easier to apply. For lack of dimensionally annotated public data, we used discrete categorization in this investigation.

It's worth noting that the accuracy and effectiveness of speech to emoji converters can vary depending on the specific tool or software used, as well as the quality of its underlying algorithms and data.

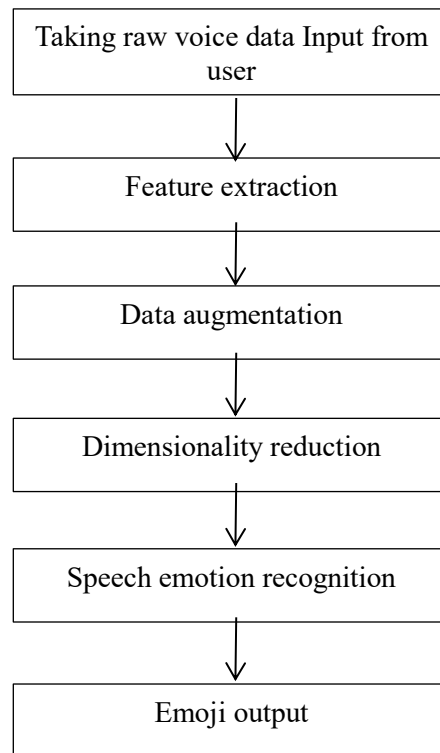


Fig 5.3 - Flow diagram for speech to emoji converter

CHAPTER 6

SYSTEM IMPLEMENTATION

6.1 Introduction

Data analysis is an essential capability in computational analytics and machine learning. In addition, statistics rely on detailed explanations and data estimates. Data visualization is an effective toolkit to achieve a contextual understanding. It can be helpful as you research and study a dataset which can help to detect trends, missing results, outliers and several other items. Data visualizations are able to be utilized with a little technical awareness to communicate and explain core relations in more graphic and involved plots and maps than cumulative or concrete measures.

6.1.1 Natural Language Toolkit

Natural Language Toolkit (NLTK) is a popular open-source platform for building Python programs to work with human language data. It is a comprehensive library of tools and resources for natural language processing (NLP) tasks such as tokenization, stemming, lemmatization, parsing, and more.

NLTK is widely used by researchers, educators, and developers to explore and analyze human language data for a variety of applications, including machine learning, text classification, sentiment analysis, language translation, and more. NLTK provides easy-to-use interfaces and libraries for a wide range of NLP tasks, as well as comprehensive documentation, tutorials, and community support. It also includes a large collection of corpora, or sets of text data, for training and testing NLP models.

Overall, NLTK is a powerful and versatile tool for anyone interested in working with human language data using Python.

Here are some advantages of using NLTK:

- User-friendly and easy to learn: NLTK is designed to be easy to use and learn, even for those with little or no prior experience in NLP.
- Extensible and customizable: NLTK is highly extensible, allowing users to create their own custom tools and libraries.
- Supports multiple languages: NLTK supports multiple languages, including English, Spanish, French, and more.

6.1.2 Convolutional Neural Network Algorithm

A Convolutional Neural Network (CNN) is a type of deep neural network that is commonly used for image and video processing tasks. It is designed to automatically learn spatial hierarchies of features from raw image data. CNNs are inspired by the structure of the visual cortex in the brain and have proven to be highly effective in tasks such as image classification, object detection, and segmentation.

The key idea behind CNNs is to apply filters (also known as kernels or convolutional layers) to the input image in order to extract meaningful features. These filters are typically small matrices that slide over the image, performing element-wise multiplication and then summing the results. The output of this operation is known as a feature map, which highlights areas of the image that are most relevant to the task at hand. Multiple filters can be applied in parallel, resulting in multiple feature maps. CNNs also use pooling layers to reduce the spatial dimensions of the feature maps while retaining the most important information. Pooling can be performed in various ways, including max pooling and average pooling, which reduce the size of the feature maps by taking the maximum or average value of a group of adjacent pixels.

Finally, the feature maps are fed into one or more fully connected layers, which perform classification or regression tasks, depending on the problem being solved. CNNs can be trained using a variety of optimization techniques, including stochastic gradient descent and its variants, and can be regularized using techniques such as dropout and weight decay to prevent overfitting.

Overall, CNNs are a powerful tool for image and video processing tasks and have achieved state-of-the-art performance on a wide range of benchmarks.

6.1.3 Haar Cascade

The Haar cascade algorithm is a machine learning-based approach used for object detection in images or videos. It was proposed by Viola and Jones in 2001 and is based on the Haar wavelet transform. The algorithm uses a set of Haar-like features that are computed from rectangular subregions of the image. These features are used to train a classifier that can distinguish between positive and negative samples.

The training process involves feeding the classifier with a set of positive and negative samples. Positive samples are images containing the object that needs to be detected, while negative samples are images that do not contain the object. The classifier learns to identify the object by comparing the Haar-like features of the positive and negative samples.

Once the classifier is trained, it can be used to detect objects in new images or videos. The algorithm scans the image or video frame-by-frame, applying the classifier to each sub region of the image. If the sub region is classified as containing the object, the algorithm marks it as a potential detection. To reduce false positives, the algorithm applies a technique called non-maximum suppression, which eliminates overlapping detections. The Haar cascade algorithm is widely used for object detection in applications such as face detection, pedestrian detection, and vehicle detection. It is computationally efficient and can achieve high detection rates with low false positives. However, it may not work well with objects that have complex shapes or textures.

6.2 Use Case Diagram

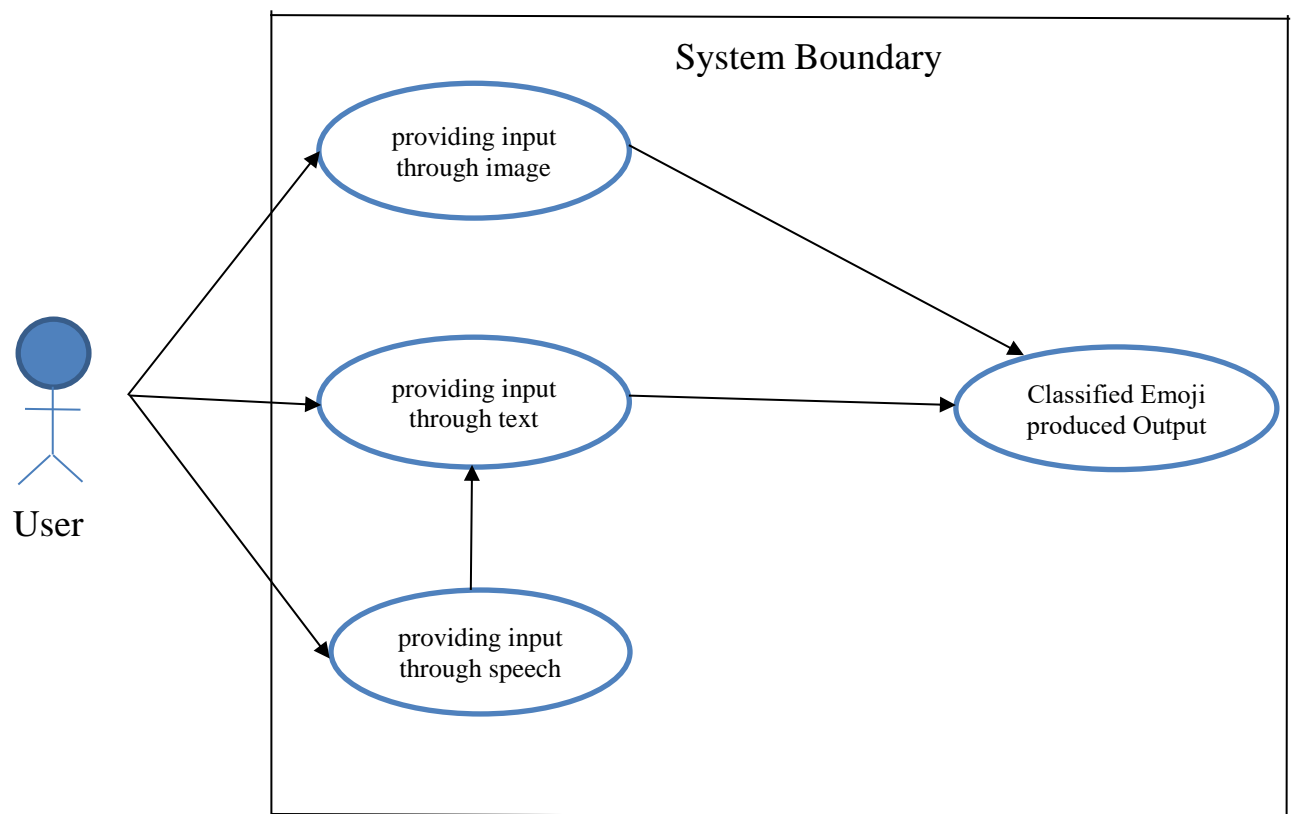


Fig 6.1 Use case diagram

6.2.1 providing input through image

The User and Admin provides the input and it uses the image recognition algorithm to identify the content of the image, and then map the identified object or concept to an emoji.

For example, if the image recognition algorithm identifies a dog in the image, the mapping could be to the 🐶 emoji. Similarly, if the algorithm identifies a sunset, the mapping could be to the 🌅 emoji.

6.2.2 providing input through text

The User and Admin provides the input through text like Smiling, laughing and thinking etc. These are just a few examples, and there are many more emojis you can use. To input emojis on a computer, you can use a keyboard shortcut or an emoji keyboard. Based on the given input provides the relative emoji.

6.2.3 providing input through speech

The User and Admin provides the input through speech. It displays please say something and when you provide the input like "I am happy" or "I am sad" and the device will recognize your command and insert the corresponding emoji in the message. However, it's worth noting that the availability of this feature and the specific commands used to activate it may vary depending on the device and software you are using.

6.3 Screenshots

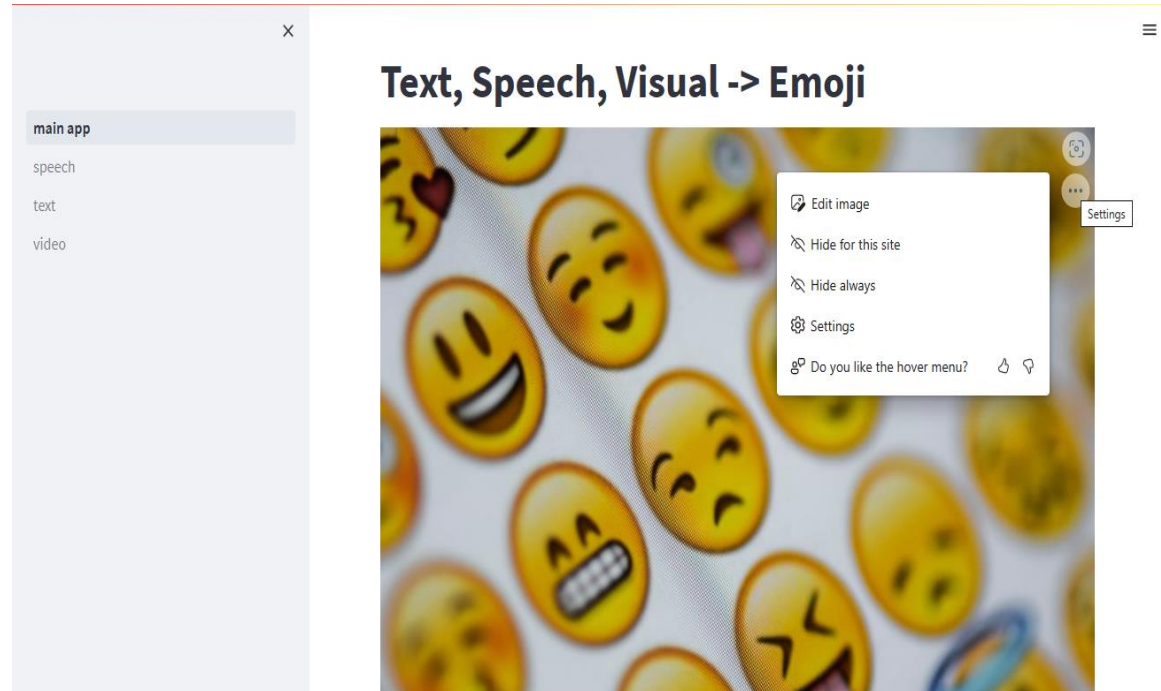


Fig 6.2 home page

This is the homepage of our project here we can choose what type of input we have to provide among the speech, text and video.

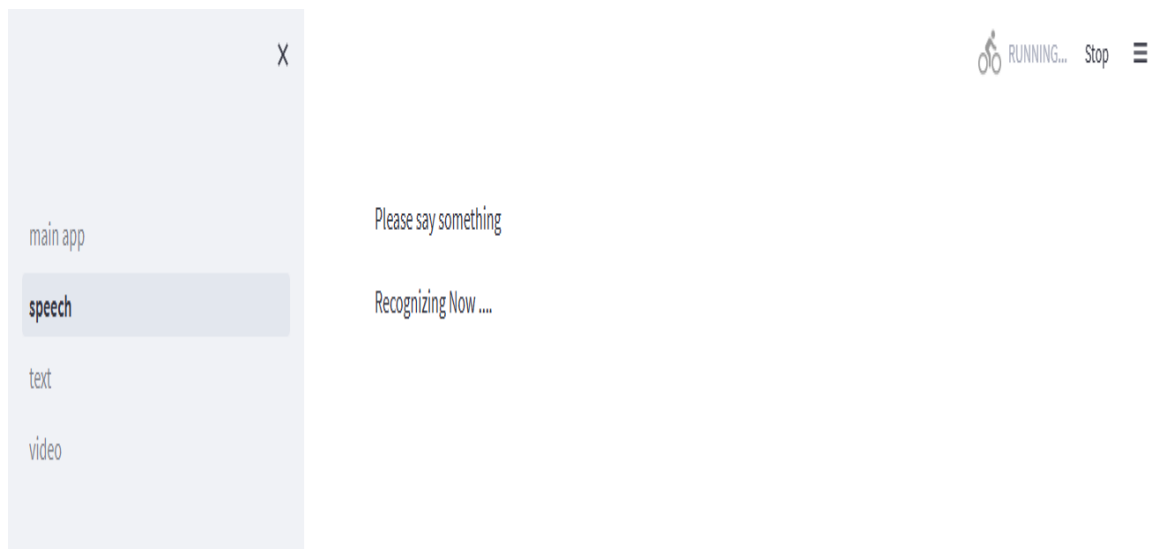


Fig 6.3 providing input to speech

Here it is waiting for the user to provide the input through the speech by displaying the message “Please say something”.

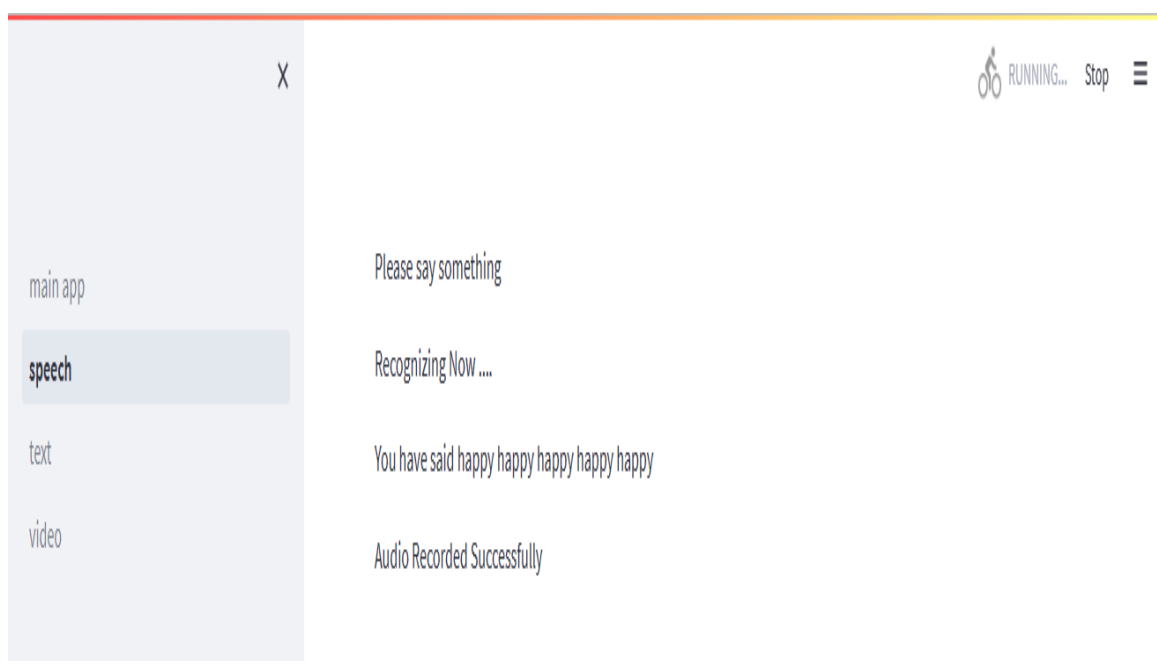


Fig 6.4 Acquiring the input given through speech

Here it is acquiring the input provided by the user through speech and after acquiring the input it displays “Audio Recorded Successfully”.

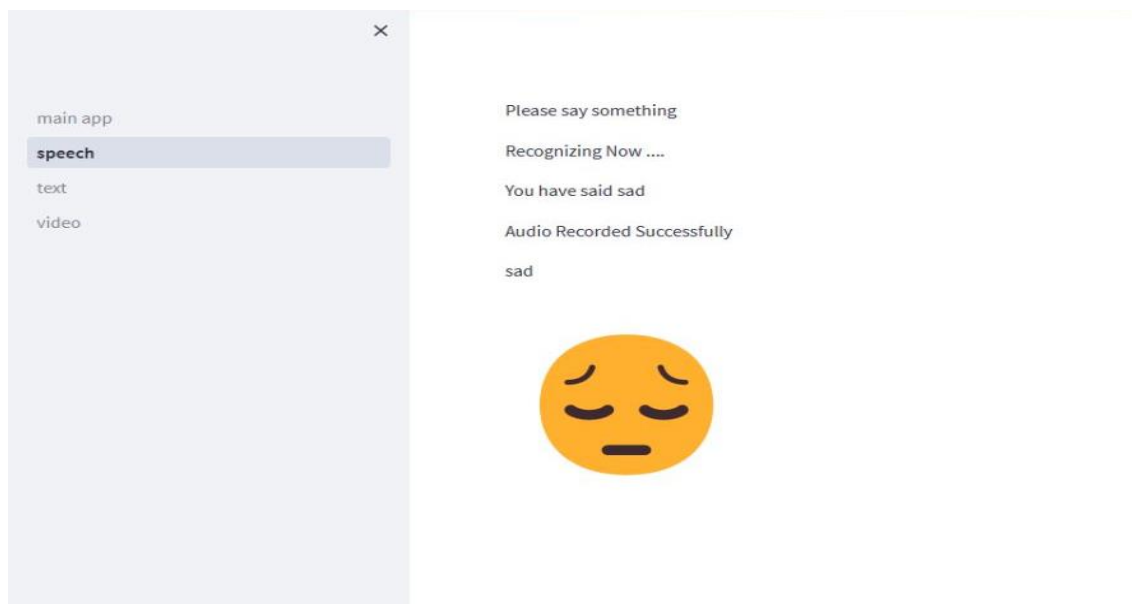


Fig 6.5 Displaying emoji for speech input

Here it displays the emoji according to the input provided by the user. It Selects the emoji based on the keywords such as sad, happy etc.

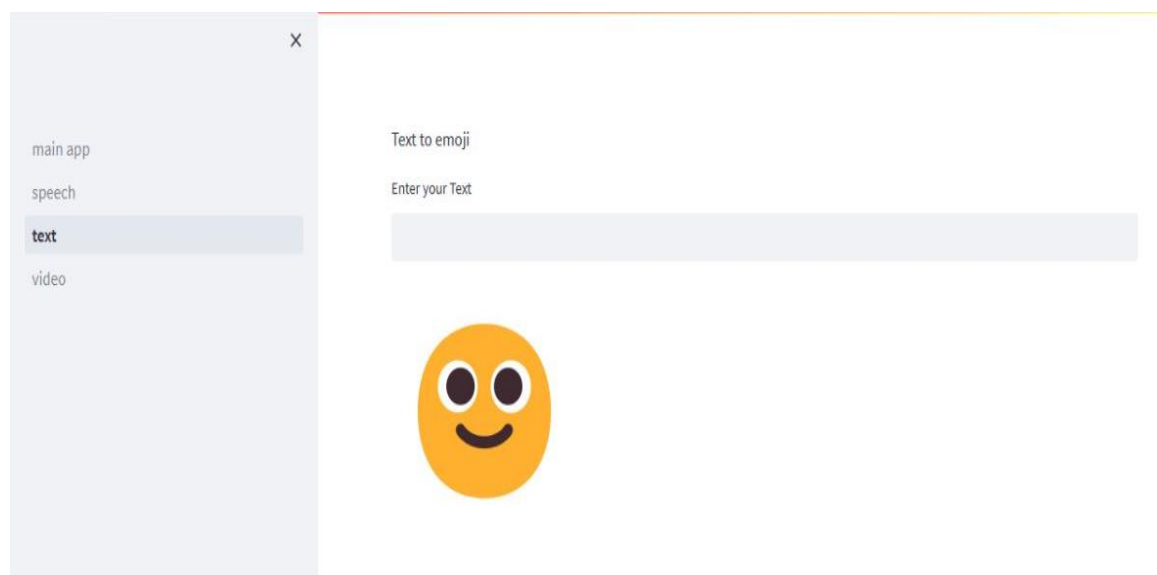


Fig 6.6 Acquiring the input given through text

Here it is acquiring the input provided by the user through text. It asks for the input through the message “Enter your Text”. Note that the below smiling emoji is the default emoji.

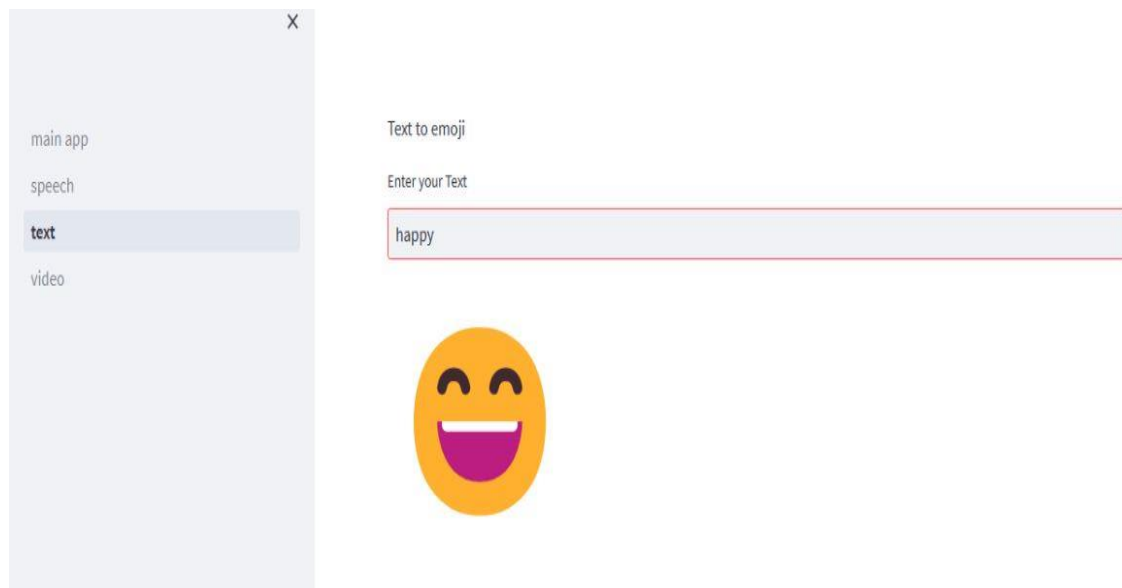


Fig 6.7 Displaying emoji for text input

Here it displays the emoji according to the input provided by the user. It replaces the default emoji with the fetched emoji according to the text input.



Fig 6.8 Acquiring the input given through real time image

Here it is recognizing the input provided by the user through real time image. Here the image is recognized through the laptop camara.

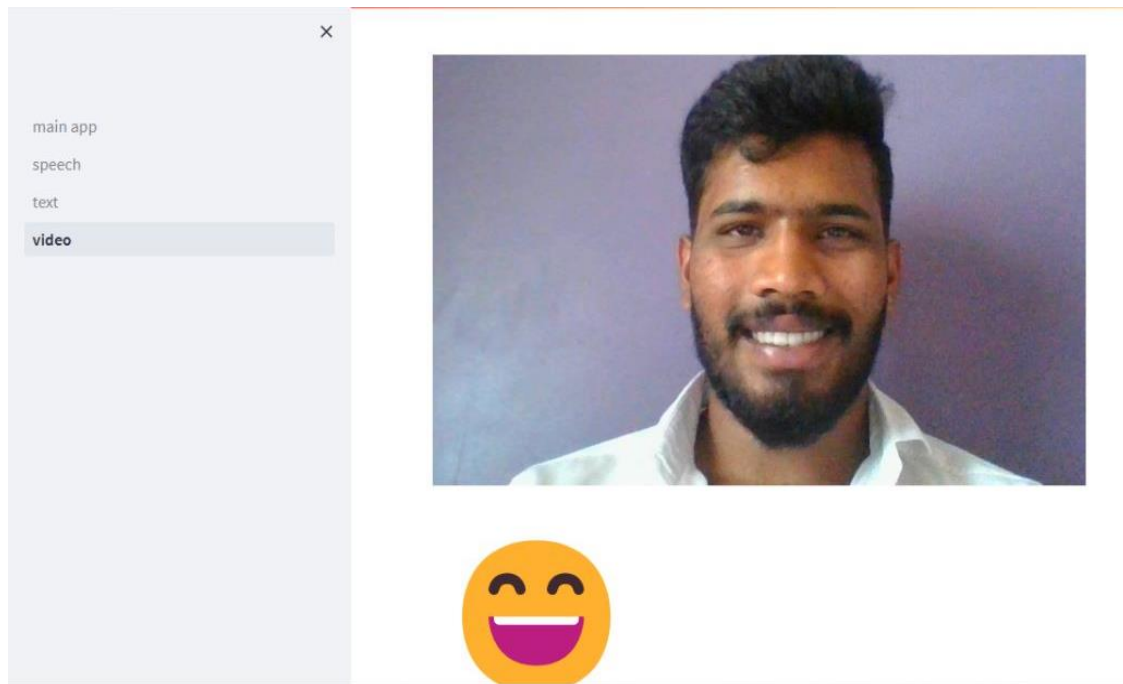


Fig 6.9 Displaying emoji for image input

Here it displays the emoji according to the input provided by the user. Since it provides real time output the emoji very simultaneously according to the real time image provided.

CHAPTER 7

SYSTEM TESTING

7.1 Test Cases

In this phase the model is being tested with various user input. For a particular input the model produces an output.

Test case 1

Test Case Name	Test Description	Input	Expected output	Actual Results/Drawbacks
User Interface	All the dataset values must be free from null data and redundancy.	Data sets input sheet (.xlsx)	Clean dataset with no redundancy	Successfully shows the cleaned dataset

Table 7.1: Checking for Redundancy

Test case 2

Test Case Name	Test Description	Input	Expected output	Actual Results/Drawbacks
User Interface	Ensure that the user interface functions as intended and is intuitive for users.	Facial image, text, or speech input	Corresponding emoji, or avatar mapping displayed to the user	User interface successfully displays the corresponding emoji or avatar mapping based on the input

Table 7.2: User Interface

7.2 Tests Conducted

The purpose of testing is to discover errors. Testing is the process of trying to discover every conceivable fault or weakness in a work product. It provides a way to check the functionality of components, sub-assemblies, assemblies and a finished product. It is the process of exercising software with the intent of ensuring that the Software system meets its requirements and user expectations and does not fail in an unacceptable manner. There are various types of tests. Each test type addresses a specific testing requirement.

CHAPTER 8

SUMMARY

This work aims to provide a worldwide perspective and clues for emoji researchers by conducting a comprehensive review of related studies. It provides a concise overview of emoji's history, current state of development, usage patterns, functional qualities, and research areas. Originating as emoticons, emoji now serve both emotional and semantic purposes. Different people, places, and technologies all have an impact on and contribute to the wide range of emoji usage that exists today. It's possible for there to be some ambiguity and miscommunication in many contexts and cultures. This also examines the state of emoji research from a number of different angles, including communication, social media platforms and education proposes. some new directions for the study of emoji in the future, including explorations of their emotional associations, user preferences, and potential social and technological impacts

CONCLUSION AND FUTURE WORK

Conclusion

The Emojifier project is a powerful software application that uses advanced computer vision and machine learning techniques to classify human facial expressions and map them to corresponding emojis or avatars. The project aims to enhance communication and expression in online environments by providing users with an intuitive and easy-to-use interface to convey emotions and sentiments accurately and effectively. To ensure that the Emojifier project is functioning as intended, various test cases have been developed to validate the different components of the project. These test cases ensure that the user interface is functioning correctly, the collected data is preprocessed accurately, the features are extracted accurately, the machine learning models are trained accurately, and the deployed model maps input to corresponding emoji or avatar accurately. Overall, the Emojifier project has many potential applications, including chatbots, customer service, virtual reality environments, and marketing and brand emotion analysis. It is a powerful tool for enhancing communication and expression in online environments, improving the user experience and facilitating the conveyance of emotions and sentiments.

Future Work

In the future we can Improve the accuracy of emoji prediction, extend your emojifier to support multiple languages, Instead of predicting emojis from text, explore the possibility of generating custom emojis based on the content of the text etc.

REFERENCES

- [1] "Emotion Recognition Based Emoji Retrieval Using Deep Learning", S.Srivastava, P. Gupta and P. Kumar, 2021 5th International Conference on Trends in Electronics and Informatics (ICOEI), 2021, pg. 1182-1186.
- [2] "Anon-Emoji: An Optical See-Through Augmented Reality System for Children with Autism Spectrum Disorders to promote Understanding of Facial Expressions and Emotions," R. Sun, H. Haraldsson, Y. Zhao and S. Belongie, 2019 IEEE International Symposium on Mixed and Augmented Reality Adjunct (ISMAR-Adjunct), 2019, pg. 448-450.
- [3] "Facial Emoji Recognition", N.Swapna Goud, K. Revanth Reddy, G.Alekha and G.S Suchitha, International JTheirnal of Trend in Scientific Research and Development(IJTSRD), Volume:3 Issue:3, March-April 2019, Page No. 1330-1333.
- [4] "Real-Time Facial Expression Recognition Based on CNN," K.C. Liu, C.C. Hsu, W.Y. Wang and H.H. Chiang, 2019 International Conference on System Science and Engineering (ICSSE), 2019, pg. 120-123.
- [5] "Emojify Create yTheir own emoji with Deep Learning", Payas Doshi, Priyanshi Sethi, Pulkit Sharma, Mahaveer Jain, International Research JTheirnal of Engineering and Technology (IRJET), Volume: 08 Issue: 11 , Nov 2021. Page No. 1014-1024.
- [6] "Exploiting Deep Neural Networks for Tweet-based Emoji Prediction", Andrei Catalin Coman1 , Giacomo Zara , Yaroslav Nechaev , Gianni Barlacchi , and Alessandro Moschit, University of Trento, Trento, Italy.
- [7] "Transformation of Facial Expression into corresponding Emoticons", Ankur Ankit, Dhananjay Narayan, Alok Kumar International JTheirnal of Engineering and Advanced Technology (IJEAT). Volume-8 Issue-5, June 2019
- [8] "Mapping of human facial expressions to emojis using Deep learning", Parth Vishe , Pune Institute of Computer Technology, Pune, Maharashtra, India. May 2022, Volume 9, Issue 5 .
- [9] "Emojify: Emoji Prediction from Sentence", Chen Huang, Xueying (Shirley) Xie, Boyu (Bill) Zhang, Stanford University.
- [10] "Emojify-Create Ytheir Own Emojis With Deep Learning" Sagar Chilivery, Sandeep Pukale, Yashraj Sonawane Vasantdada Patil Pratishthan's College Of Engineering.

APPENDIX A

MACHINE LEARNING

A.1 Supervised Machine Learning

A supervised learning algorithm learns from labeled training data, helps to predict out- comes for unforeseen data.

A.1.1 Logistic Regression

Logistic regression is a computational methodology which often represents the associations between the independent variables $x_1 x_2 \dots x_n$ and y that is the two alternative categories of discrete dependent variable coded in 0 or 1.

A.1.2 Naive Bayes

A basic learning algorithm that uses Bayes and strongly assumes that the attributes in the groups are conditionally separate.

A.1.3 Random Forest

Random forests were a hybrid of trees forecasting flaws in forest general statement since each tree operates on the theory of the random variable within each tree.

A.1.4 Support Vector Machine

Support Vector Machine has been a collection of similar supervised learning methods for use in categorization and regression.

A.1.5 K-Nearest Neighbours

K-Nearest Neighbors is the simplest and most direct method for classification where the distribution of data has had no firsthand knowledge. This rule essentially preserves the whole training set in the process of learning and assigns a class representing the majority mark of its closest neighbors in the training set to each submission.

A.1.6 Decision Trees

Decision Trees is a data gathering categorization and analysis technique. Decision trees constitute

the fundamental recursive basis for the sequence process of classification, in that one of the disjoint class decision-making structures contains nodes and leaves, is allocated a case identified with a collection of attributes.

A.1.7 Code for training the CNN algorithm

```
# Import the required libraries
import numpy as np
import cv2
# from keras import Sequential
# from keras.layers import Dense
from keras.models import Sequential
from keras.layers import Dense, Dropout, LSTM, BatchNormalization
from keras.layers import Dropout
from keras.layers import Flatten
from keras.layers import Conv2D
from keras.optimizers import adam_v2
# from tensorflow.keras.optimizers import Adam
from keras.layers import MaxPooling2D
from keras.preprocessing.image import ImageDataGenerator
from tensorflow.keras.utils import plot_model

# Initialize the training and validation generators:
train_dir = 'data/train'
val_dir = 'data/test'
train_datagen = ImageDataGenerator(rescale=1./255)
val_datagen = ImageDataGenerator(rescale=1./255)
#training generator for CNN
train_generator = train_datagen.flow_from_directory(
    train_dir,
    target_size=(48,48),
    batch_size=64,
    color_mode="grayscale",
    class_mode='categorical')
```

#validation generator for CNN

```
validation_generator = val_datagen.flow_from_directory(  
    val_dir,  
    target_size=(48,48),  
    batch_size=64,  
    color_mode="grayscale",  
    class_mode='categorical')
```

To display the train data

```
# for i in os.listdir("train/"):
```

```
# print(str(len(os.listdir("train/"+i))) + " " + i + " images")
```

Build the convolution network architecture:

```
emotion_model = Sequential()
```

```
emotion_model.add(Conv2D(32, kernel_size=(3, 3), activation='relu',  
input_shape=(48,48,1)))#output=(48-3+0)/1+1=46
```

```
emotion_model.add(Conv2D(64, kernel_size=(3, 3), activation='relu'))#output=(46-  
3+0)/1+1=44
```

```
emotion_model.add(MaxPooling2D(pool_size=(2, 2)))#output=devided input by 2 it means  
22,22,64
```

```
emotion_model.add(Dropout(0.25))#reduce 25% module at a time of output
```

```
emotion_model.add(Conv2D(128, kernel_size=(3, 3),  
activation='relu',input_shape=(48,48,1)))#(22-3+0)/1+1=20
```

```
emotion_model.add(MaxPooling2D(pool_size=(2, 2)))#10
```

```
emotion_model.add(Conv2D(128, kernel_size=(3, 3), activation='relu'))#(10-3+0)/1+1=8
```

```
emotion_model.add(MaxPooling2D(pool_size=(2, 2)))#output=4
```

```
emotion_model.add(Dropout(0.25))#nothing change
```

```
emotion_model.add(Flatten())#here we get multidimension output and pass as linear to the  
dense so that 4*4*128=2048
```

```
emotion_model.add(Dense(1024, activation='relu'))#hddien of 1024 neurons of input
```

```
emotion_model.add(Dropout(0.5))
```

```
emotion_model.add(Dense(7, activation='softmax'))#hddien of 7 neurons of input
```

```
# plot_model(emotion_model, to_file='model_plot.png', show_shapes=True,
```

```
show_layer_names=True)#save model leyer as model_plot.png
```

```
emotion_model.summary()
```



```
# Compile and train the model
emotion_model.compile(loss='categorical_crossentropy',optimizer=adam_v2.Adam(learning_rate=0.0001, decay=1e-6),metrics=['accuracy'])
# emotion_model.compile(loss='categorical_crossentropy',optimizer=Adam)
emotion_model_info = emotion_model.fit( #to fetch the model info from validation generator
    train_generator,
    steps_per_epoch=28709 // 64,
    epochs=5,
    validation_data=validation_generator,
    validation_steps=7178 // 64)

# Save the model weights:
emotion_model.save_weights('model.h5')
```

A.1.7 Code for Converting video to emoji

```
# Import the Libraries
import tkinter as tk
from tkinter import *
import cv2
from PIL import Image
from PIL import ImageTk
import os
import numpy as np
import cv2
from keras.models import Sequential
from keras.layers import Dense
from keras.layers import Dropout
from keras.layers import Flatten
from keras.layers import Conv2D
# from keras.optimizers import adam
from tensorflow.keras.optimizers import Adam
from keras.layers import MaxPooling2D
from keras.preprocessing.image import ImageDataGenerator
import threading
```

Model Creation

```

emotion_model = Sequential()#to extract the features in model
emotion_model.add(Conv2D(32, kernel_size=(3, 3), activation='relu', input_shape=(48,48,1)))
emotion_model.add(Conv2D(64, kernel_size=(3, 3), activation='relu'))
emotion_model.add(MaxPooling2D(pool_size=(2, 2)))
emotion_model.add(Dropout(0.25))
emotion_model.add(Conv2D(128, kernel_size=(3, 3), activation='relu'))
emotion_model.add(MaxPooling2D(pool_size=(2, 2)))
emotion_model.add(Conv2D(128, kernel_size=(3, 3), activation='relu'))
emotion_model.add(MaxPooling2D(pool_size=(2, 2)))
emotion_model.add(Dropout(0.25))
emotion_model.add(Flatten())
emotion_model.add(Dense(1024, activation='relu'))
emotion_model.add(Dropout(0.5))
emotion_model.add(Dense(7, activation='softmax'))
emotion_model.load_weights('model.h5')
cv2ocl.setUseOpenCL(False)

```

Mapping of facial emotion with Avtar

#emotion dictionary contains the emotions present in the dataset

```

em_dict = {0: " Angry ", 1: "Disgusted", 2: " Fearful ", 3: " Happy ", 4: " Neutral ", 5: "
Sad ", 6: "Surprised"}
cur_path = os.path.dirname(os.path.abspath(__file__))

```

```

emoji_dist={0:cur_path+"/emojis/angry.png",1:cur_path+"/emojis/disgusted.png",
2:cur_path+"/emojis/fearful.png",3:cur_path+"/emojis/happy.png",4:cur_path+"/emojis/neutral.
png",5:cur_path+"/emojis/sad.png",6:cur_path+"/emojis/surprised.png"}

```

```

#emoji_dist={0:"./emojis/angry.png",2:"./emojis/disgust.png",2:"./emojis/fear.png",3:"./emojis/
happy.png",4:"./emojis/neutral.png",5:"./emojis/sad.png",6:"./emojis/surprise.png"}

```

global last_frame1 #emoji dictionary is created with images for every emotion present ion dataset

```
last_frame1 = np.zeros((480, 640, 3), dtype=np.uint8)
```

global cap1

global frame_number

```
show_text=[0]
```

```

def show_subject(): #to open the camera and to record video
    # cap1 = cv2.VideoCapture(r'C:\Users\Aryaman\Pictures\Camera Roll\surprised.mp4') #it
starts capturing
    cap1 = cv2.VideoCapture(0) #it starts capturing
    if not cap1.isOpened(): #if camera is not open
        print("Can't open the camera")
    global frame_number
    length = int(cap1.get(cv2.CAP_PROP_FRAME_COUNT))
    frame_number +=1
    # if frame_number >= length:
    # exit()
    # cap1.set(1, frame_number)
    flag1, frame1 = cap1.read()
    frame1 = cv2.resize(frame1,(600,500))#to resize the image frame
    bound_box = cv2.CascadeClassifier('haar_cascade_frontal_face.xml')
    gray_frame = cv2.cvtColor(frame1, cv2.COLOR_BGR2GRAY)#to color the frame
    num_faces = bound_box.detectMultiScale(gray_frame,scaleFactor=1.3, minNeighbors=5)
    for (x, y, w, h) in num_faces: #for n different faces of a video
        cv2.rectangle(frame1, (x, y-50), (x+w, y+h+10), (255, 0, 0), 2)
        roi_frame = gray_frame[y:y + h, x:x + w]
        crop_img = np.expand_dims(np.expand_dims(cv2.resize(roi_frame, (48, 48)), -1),
0)#crop the image and save only emotion contating face
        prediction = emotion_model.predict(crop_img)#predict the emotion from the cropped
image
        maxindex = int(np.argmax(prediction))
        cv2.putText(frame1, em_dict[maxindex], (x+20, y-60),
cv2.FONT_HERSHEY_SIMPLEX, 1, (255, 255, 255), 2, cv2.LINE_AA)

        show_text[0]=maxindex #store the emotion found in image from emotion dictionary
    if flag1 is None:
        print ("Major error!")
    elif flag1:
        global last_frame1
        last_frame1 = frame1.copy()
        pic = cv2.cvtColor(last_frame1, cv2.COLOR_BGR2RGB) #to store the image

```

```
img = Image.fromarray(pic)
    imgtk = ImageTk.PhotoImage(image=img)
    lmain.imgtk = imgtk
    lmain.configure(image=imgtk)
    root.update()
    lmain.after(10, show_subject)
if cv2.waitKey(1) & 0xFF == ord('q'):
    exit()

def show_avatar():
    frame2=cv2.imread(emoji_dist[show_text[0]])#to store the emoji with respect to the emotion
    pic2=cv2.cvtColor(frame2,cv2.COLOR_BGR2RGB)
    img2=Image.fromarray(frame2)
    imgtk2=ImageTk.PhotoImage(image=img2)
    lmain2.imgtk2=imgtk2
    lmain3.configure(text=em_dict[show_text[0]],font=('arial',45,'bold'))#to configure image and
text
    lmain2.configure(image=imgtk2)
    root.update()
    lmain2.after(10, show_avatar)

if __name__ == '__main__':
    frame_number = 0
    root=tk.Tk()
    lmain = tk.Label(master=root,padx=50,bd=10)
    lmain2 = tk.Label(master=root,bd=10)
    lmain3=tk.Label(master=root,bd=10,fg="#CDCDCD",bg='black')
    lmain.pack(side=LEFT)
    lmain.place(x=50,y=250)
    lmain3.pack()
    lmain3.place(x=960,y=250)
    lmain2.pack(side=RIGHT)
    lmain2.place(x=900,y=350)
    root.title("Photo To Emoji")
```

```
root.geometry("1400x900+100+10")
root['bg']='black'
exitbutton = Button(root,
text='Quit',fg="red",command=root.destroy,font=('arial',25,'bold')).pack(side = BOTTOM)
threading.Thread(target=show_subject).start()
threading.Thread(target=show_avatar).start()
root.mainloop()
```

APPENDIX B

EVALUATION PARAMATERS

B.1 Precision and Recall

Pattern detection is a fraction of the appropriate instances within the retrieved instances, with knowledge retrieval and classification (machine study), accuracy (also called optimistic predictive value), while recall (also known as sensitivity) is the fraction of the total number of the specific instances that were currently retrieved. Accuracy and alert are both dependent on awareness and significance calculation.

B.2 Pattern Detection

Pattern detection is a fraction of the appropriate instances within the retrieved instances, with knowledge retrieval and classification (machine study), accuracy (also called optimistic predictive value), while recall (also known as sensitivity) is the fraction of the total number of the specific instances that were currently retrieved. Accuracy and alert are both dependent on awareness and significance calculation.

B.3 Confusion Matrix

A confusion matrix is a table that is often used to describe the performance of a classification model (or “classifier”) on a set of test data for which the true values are known. It allows the visualization of the performance of an algorithm.

B.4 F1 Score

The indicator of precision of a check is the F1 (also F-point or F-mount). It takes into consideration both the precision p and the warning r of the check in order to measure the score: p is the proportion of correct positive results determined by the number of possible results recorded by the classifier and r is the number of correct positive results segregated by all samples involved.