

EMOTION RECOGNITION USING SPEECH PROCESSING

Submitted in partial fulfillment of the
requirements for the award of
Bachelor of Engineering degree in Computer Science and Engineering

By

HIMA SRI KORIPALLI (39110381)
A VASAVI SREYA (39110099)



DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING
SCHOOL OF COMPUTING

SATHYABAMA

INSTITUTE OF SCIENCE AND TECHNOLOGY
(DEEMED TO BE UNIVERSITY)

Accredited with Grade "A" by NAAC | 12B Status by UGC | Approved by AICTE
JEPPIAAR NAGAR, RAJIV GANDHISALAI,
CHENNAI - 600119

APRIL- 2023



SATHYABAMA

INSTITUTE OF SCIENCE AND TECHNOLOGY
(DEEMED TO BE UNIVERSITY)

Accredited with Grade "A" by NAAC | 12B Status by UGC | Approved by AICTE

www.sathyabama.ac.in

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

BONAFIDE CERTIFICATE

This is to certify that this Project Report is the bonafide work **HIMA SRI.KORIPALLI (Reg.No-39110381)** and **A.Vasavi Sreya (Reg.No-39110099)** who carried out the Project entitled "**EMOTION RECOGNITION USING SPEECH PROCESSING**" under my supervision from January 2023 to April 2023.

Internal Guide

Dr. L. SUJIHELEN, M.E., Ph.D.

Head of the Department

Dr. L. LAKSHMANAN, M.E., Ph.D.,



Submitted for Viva voce Examination held on 20.4.2023

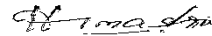
Internal Examiner

External Examiner

DECLARATION

I, HIMA SRI KORIPALLI (Reg. No - 39110381) and A VASAVI SREYA (Reg. No- 39110099) hereby declare that the Project Report entitled “**EMOTION RECOGNITION USING SPEECH PROCESSING**” done by me under the guidance of **Dr. L. SUJHELEN, M.E., Ph.D.**, is submitted in partial fulfillment of the requirements for the award of Bachelor of Engineering degree in **Computer Science and Engineering**.

DATE: 20.4.2023



PLACE: Chennai

SIGNATURE OF THE CANDIDATE

ACKNOWLEDGEMENT

I am pleased to acknowledge my sincere thanks to **Board of Management of SATHYABAMA** for their kind encouragement in doing this project and for completing it successfully. I am grateful to them.

I convey my thanks to **Dr. T.Sasikala M.E., Ph. D, Dean**, School of Computing, **Dr. L. Lakshmanan M.E., Ph.D.**, Head of the Department of Computer Science and Engineering for providing me necessary support and details at the right time during the progressive reviews.

I would like to express my sincere and deep sense of gratitude to my Project Guide **Dr. L. SUJHELEN M.E., Ph.D.**, for her valuable guidance, suggestions and constant encouragement paved way for the successful completion of my phase-2 project work.

I wish to express my thanks to all Teaching and Non-teaching staff members of the **Department of Computer Science and Engineering** who were helpful in many ways for the completion of the project.

ABSTRACT

Speech is the main and direct means of transmitting information. It contains a wide variety of information, and it can express rich emotional information through the emotions it contains and visualize it in response to object, scenes or events. In human machine interface application, emotion recognition from the speech signal has been research topic since many years. Emotions play an extremely important role in human mental life. It is a medium of expression of one's perspective or one's mental state to others. Speech Emotion Recognition (SER) can be defined as extraction of the emotional state of the speaker from his or her speech signal. Deep Neural Networks (DNNs) are based on feed-forward structures comprised of one or more underlying hidden layers between inputs and outputs. The feed-forward architectures such as Deep Neural Networks (DNNs) and Convolutional Neural Networks (CNNs) provides efficient results for image and video processing. On the other hand, recurrent architectures such as Recurrent Neural Networks (RNNs) and Long Short-Term Memory (LSTM) are much effective in speech-based classification such as natural language processing (NLP) and SER. Apart from their effective way of classification these models do have some limitations. For instance, the positive aspect of CNNs is to learn features from high-dimensional input data, but on the other hand, it also learns features from small variations and distortion occurrence and hence, requires large storage capability. Similarly, LSTM-based RNNs are able to handle variable input data and model long-range sequential text data. There are few universal emotions- including Neutral, Anger, Happiness, Sadness etc. in which any intelligent system with finite computational resources can be trained to identify or synthesize as required. In this work, we are extracting Mel-frequency cepstral coefficients (MFCC), Chromogram, Mel scaled spectrogram in conjunction with Spectral contrast and Tonal Centroid features.

TABLE OF CONTENTS

CHAPTERNO	TITLE	Page No.
	ABSTRACT	v
	LIST OF FIGURES	viii
1	INTRODUCTION	1
2	LITERATURE SURVEY	3
	2.1 Summaries for literature survey	
	2.2 Open problems in Existing System	4
3	REQUIREMENTS ANALYSIS	
	3.1 Feasibility Studies	6
	3.2 Software and Hardware Requirements	7
	3.3 System Use case	8
4	DESCRIPTION OF PROPOSED SYSTEM	
	4.1 Selected Methodology or process model	17
	4.2 Architecture / Overall Design of Proposed System	18
	4.3 Description Of Software For Implementation And Testing Plan Of The Proposed System	19
5	IMPLEMENTATION DETAILS	
	5.1 Development and Deployment Setup	21
	5.2 Algorithms	26
	5.3 Testing	34
6	RESULTS AND DISCUSSION	37
7	CONCLUSION	
	7.1 Conclusion	40
	7.2 Future work	40
8	REFERENCES	41
	APPENDIX	
	A. SOURCE CODE	43
	B. SCREENSHOTS	52

LIST OF FIGURES

FIGURE NO	FIGURE NAME	Page No.
3.1	USE CASE DIAGRAM	9
3.2	CLASS DIAGRAM	10
3.3	SEQUENCE DIAGRAM	11
3.4	COLLABORATION DIAGRAM	12
3.5	DEPLOYMENT DIAGRAM	12
3.6	ACTIVITY DIAGRAM	13
3.7	COMPONENT DIAGRAM	14
3.8	ER DIAGRAM	15
3.9	DFD DIAGRAM	16
4.1	ARCHITECTURE OF PROPOSED METHOD	18
5.1	PYTHON INSTALLATION	23
5.2	PYCHARM INSTALLATION	24
5.3	PYCHARM LOGIN	25
6.1	MODEL ACCURACY 1	37
6.2	MODEL ACCURACY 2	38
6.3	MODEL ACCURACY 3	39

CHAPTER 1

INTRODUCTION

Emotion recognition from speech has evolved from being a niche to an important component for Human-Computer Interaction (HCI). These systems aim to facilitate the natural interaction with machines by direct voice interaction instead of using traditional devices as input to understand verbal content and make it easy for human listeners to react. Some applications include dialogue systems for spoken languages such as call center conversations, onboard vehicle driving system and utilization of emotion patterns from the speech in medical applications. Nonetheless, there are many problems in HCI systems that still need to be properly addressed, particularly as these systems move from lab testing to real-world application. Hence, efforts are required to effectively solve such problems and achieve better emotion recognition by machines. There are many ways of communication but the speech signal is one of the fastest and most natural methods of communications between humans. Therefore the speech can be the fast and efficient method of interaction between human and the machine as well. Humans have the natural ability to use all their available senses for maximum awareness of the received message. Through all the available senses people actually sense the emotional state of their communication partner. The emotional detection is natural for humans but it is very difficult task for machine. Therefore the purpose of emotion recognition system is to use emotion related knowledge in such a way that human machine communication will be improved. In this system, the quality of feature extraction directly affected the accuracy of speech emotion recognition. In the process of feature extraction, it usually took the whole emotion sentence as units for feature extracting, and extraction contents were four aspects of emotion speech, which were several acoustic characteristics of time construction, amplitude construction, fundamental frequency construction, and formant construction. Then contrast emotion speech with no emotion sentence from these four aspects, acquiring the law of emotional signal distribution, then classify emotion speech according to the law. Deep neural network (DNN) has unprecedented success in the field of speech recognition and image recognition; however, so far no research on deep neural network has been applied to speech emotion processing. We found that the DNN in speech emotion processing has a huge advantage. Therefore, this paper proposed a method to realize the emotional

features automatically extracted from the audio using the librosa package in python. We used DNN to train a 5-layer-deep network to extract speech emotion features. It incorporates the speech emotion features of more consecutive frames, to build a high latitude characteristic, and uses softmax classifier layer to classify the emotional speech. To classify the emotions, The important issues in speech emotion recognition system are the signal processing unit in which appropriate features are extracted from available speech signal and another is a classifier which recognizes emotions from the speech signal. The average accuracy of the most of the classifiers for speaker independent system is less than that for the speaker dependent. Automatic emotion recognitions from the human speech are increasing now a day because it results in the better interactions between human and machine. Emotion recognition systems based on digitized speech is comprised of three fundamental components: signal preprocessing, feature extraction, and classification. Acoustic preprocessing such as denoising, as well as segmentation, is carried out to determine meaningful units of the signal. Feature extraction is utilized to identify the relevant features available in the signal. Lastly, the mapping of extracted feature vectors to relevant emotions is carried out by classifiers. In this section, a detailed discussion of speech signal processing, feature extraction, and classification is provided. Also, the differences between spontaneous and acted speech are discussed due to their relevance Figure 1 depicts a simplified system utilized for speech-based emotion recognition. In the first stage of speech-based signal processing, speech enhancement is carried out where the noisy components are removed. The second stage involves two parts, feature extraction, and feature selection. The required features are extracted from the preprocessed speech signal and the selection is made from the extracted features. Such feature extraction and selection is usually based on the analysis of speech signals in the time and frequency domains. During the third stage, various classifiers such as GMM and HMM, etc. are utilized for classification of these features. Lastly, based on feature classification different emotions are recognized.

CHAPTER 2

LITERATURE SURVEY

2.1 INFERENCES FROM LITERATURE SURVEY:

Szegedy, Christian & Toshev, Alexander & Erhan, Dumitru. (2013). Deep Neural Networks (DNNs) have recently shown outstanding performance on image classification tasks. In this paper we go one step further and address the problem of object detection using DNNs, that is not only classifying but also precisely localizing objects of various classes. We present a simple and yet powerful formulation of object detection as a regression problem to object bounding box masks. We define a multi-scale inference procedure which is able to produce high-resolution object detections at a low cost by a few network applications. State-of-the-art performance of the approach is shown on Pascal VOC.

Benk, Sal & Elmir, Youssef & Dennai, Abdeslem. (2019). Speech is an easy and usable technique of communication between humans, but nowadays humans are not limited to connecting to each other but even to the different machines in our lives. The most important is the computer. So, this communication technique can be used between computers and humans. This interaction is done through interfaces, this area called Human Computer Interaction (HCI). This paper gives an overview of the main definitions of Automatic Speech Recognition (ASR) which is an important domain of artificial intelligence and which should be taken into account during any related research (Type of speech, vocabulary size... etc.). It also gives a summary of important research relevant to speech processing in the few last years, with a general idea of our proposal that could be considered as a contribution in this area of research and by giving a conclusion referring to certain enhancements that could be in the future works.

Ashish B. Ingale & D. S. Chaudhari (2012). In human machine interface application, emotion recognition from the speech signal has been research topic since many years. To identify the emotions from the speech signal, many systems have been developed. In this paper speech emotion recognition based on the previous technologies which uses different classifiers for the emotion recognition is reviewed. The classifiers are used to differentiate emotions such as anger, happiness, sadness, surprise, neutral state, etc. The database for the speech emotion recognition system

is the emotional speech samples and the features extracted from these speech samples are the energy, pitch, linear prediction cepstrum coefficient (LPCC), Mel frequency cepstrum coefficient (MFCC). The classification performance is based on extracted features. Inference about the performance and limitation of speech emotion recognition system based on the different classifiers are also discussed.

Chenchen Huang, Wei Gong, Wenlong Fu, Dongyu Feng (2014). Feature extraction is a very important part in speech emotion recognition, and in allusion to feature extraction in speech emotion recognition problems, this paper proposed a new method of feature extraction, using DBNs in DNN to extract emotional features in speech signal automatically. By training 5 layers depth DBNs, to extract speech emotion feature and incorporate multiple consecutive frames to form a high dimensional feature. The features after training in DBNs were the input of nonlinear SVM classifier, and finally speech emotion recognition multiple classifier system was achieved. The speech emotion recognition rate of the system reached 86.5%, which was 7% higher than the original method.

L. Kerkeni, Y. Serrestou, M. Mbarki, K. Raoof, M. Ali Mahjoub and C. Cleder. (2019). Emotion plays a significant role in daily interpersonal human interactions. This is essential to our rational as well as intelligent decisions. It helps us to match and understand the feelings of others by conveying our feelings and giving feedback to others. Research has revealed the powerful role that emotion plays in shaping human social interaction. Emotional displays convey considerable information about the mental state of an individual. This has opened up a new research field called automatic emotion recognition, having basic goals to understand and retrieve desired emotions.

2.2 OPEN PROBLEMS IN EXISTING SYSTEM:

In most of the currently available systems, the model used for emotion recognition uses traditional Machine learning algorithms like Support Vector Machines (SVM), K-Nearest neighbors (KNN) etc. The accuracies of these models are low. However, there are other Deep learning models as well but they are generally trained using large datasets which takes a lot of time and hence are very complex models.

Disadvantages:

- Lower accuracy.
- High computational complexity.
- Requires high performance hardware to use the application.

2.3 PROPOSED SYSTEM:

We propose a modified speech emotion recognition method which uses deep neural networks for training. The method uses Mel-frequency cepstral coefficients (MFCC), Chromogram, Mel scaled spectrogram in conjunction with Spectral contrast and Tonal Centroid features to extract details about an audio file. The features are used to train DNN model in a 5 layer deep neural network. The dataset used here is the Ryerson Audio-Visual Database of Emotional Speech and Song (RAVDESS). We have only chosen the speech part which consists of 24 actors (gender balanced) with 1440 audio files. The model classifies the speech audio in 8 different emotions namely neutral, calm, happy, sad, angry, fearful, disgust, surprised.

Advantages:

- Higher Accuracy.
- Low computational complexities.
- Does not require very high-performance hardware(s) to run.

CHAPTER 3

REQUIREMENTS ANALYSIS

3.1 FEASIBILITY STUDY:

The feasibility of the project is analyzed in this phase and business proposal is put forth with a very general plan for the project and some cost estimates. During system analysis the feasibility study of the proposed system is to be carried out. This is to ensure that the proposed system is not a burden to the company. For feasibility analysis, some understanding of the major requirements for the system is essential.

Three key considerations involved in the feasibility analysis are

- **ECONOMICAL FEASIBILITY**
- **TECHNICAL FEASIBILITY**

Economical Feasibility:

This study is carried out to check the economic impact that the system will have on the organization. The amount of fund that the company can pour into the research and development of the system is limited. The expenditures must be justified. Thus the developed system as well within the budget and this was achieved because most of the technologies used are freely available. Only the customized products had to be purchased.

Technical Feasibility:

This study is carried out to check the technical feasibility, that is, the technical requirements of the system. Any system developed must not have a high demand on the available technical resources. This will lead to high demands on the available technical resources. This will lead to high demands being placed on the client. The developed system must have a modest requirement, as only minimal or null changes are required for implementing this system. The aspect of study is to check the level of acceptance of the system by the user. This includes the process of training the user to use the system efficiently. The user must not feel threatened by the system, instead must accept it as a necessity. The level of acceptance by the users solely depends on the methods that are employed to educate the user about the system

and to make him familiar with it. His level of confidence must be raised so that he is also able to make some constructive criticism, which is welcomed, as he is the final user of the system.

3.2 SOFTWARE AND HARDWARE REQUIREMENTS

Hardware Requirements:

- Processor - I3/Intel Processor
- RAM - 4GB (min)
- Hard Disk - 128 GB
- Key Board - Standard Windows Keyboard
- Mouse - Two or Three Button Mouse
- Monitor - Any

Software Requirements:

- Operating System : Windows 7+
- Server-side Script : Python 3.6+
- IDE : PyCharm
- Libraries Used : Pandas, Numpy, Keras, Tensorflow, Librosa, OpenCV, Flask, Pickle.
- Dataset : RAVDESS speech dataset

3.3 SYSTEM USE CASE:

Uml Diagrams:

UML stands for Unified Modeling Language. UML is a standardized general-purpose modeling language in the field of object-oriented software engineering. The standard is managed, and was created by, the Object Management Group.

The goal is for UML to become a common language for creating models of object oriented computer software. In its current form UML is comprised of two major components: a Meta-model and a notation. In the future, some form of method or process may also be added to; or associated with, UML.

The Unified Modeling Language is a standard language for specifying, Visualization, Constructing and documenting the artifacts of software system, as well as for business modeling and other non-software systems.

The UML represents a collection of best engineering practices that have proven successful in the modeling of large and complex systems.

The UML is a very important part of developing objects oriented software and the software development process. The UML uses mostly graphical notations to express the design of software projects.

GOALS:

The Primary goals in the design of the UML are as follows:

1. Provide users a ready-to-use, expressive visual modeling Language so that they can develop and exchange meaningful models.
2. Provide extendibility and specialization mechanisms to extend the core concepts.
3. Be independent of particular programming languages and development process.
4. Provide a formal basis for understanding the modeling language.
5. Encourage the growth of OO tools market.

6. Support higher level development concepts such as collaborations, frameworks, patterns and components.
7. Integrate best practices.

USE CASE DIAGRAM:

A use case diagram in the Unified Modeling Language (UML) is a type of behavioral diagram defined by and created from a Use-case analysis. Its purpose is to present a graphical overview of the functionality provided by a system in terms of actors, their goals (represented as use cases), and any dependencies between those use cases. The main purpose of a use case diagram is to show what system functions are performed for which actor. Roles of the actors in the system can be depicted.

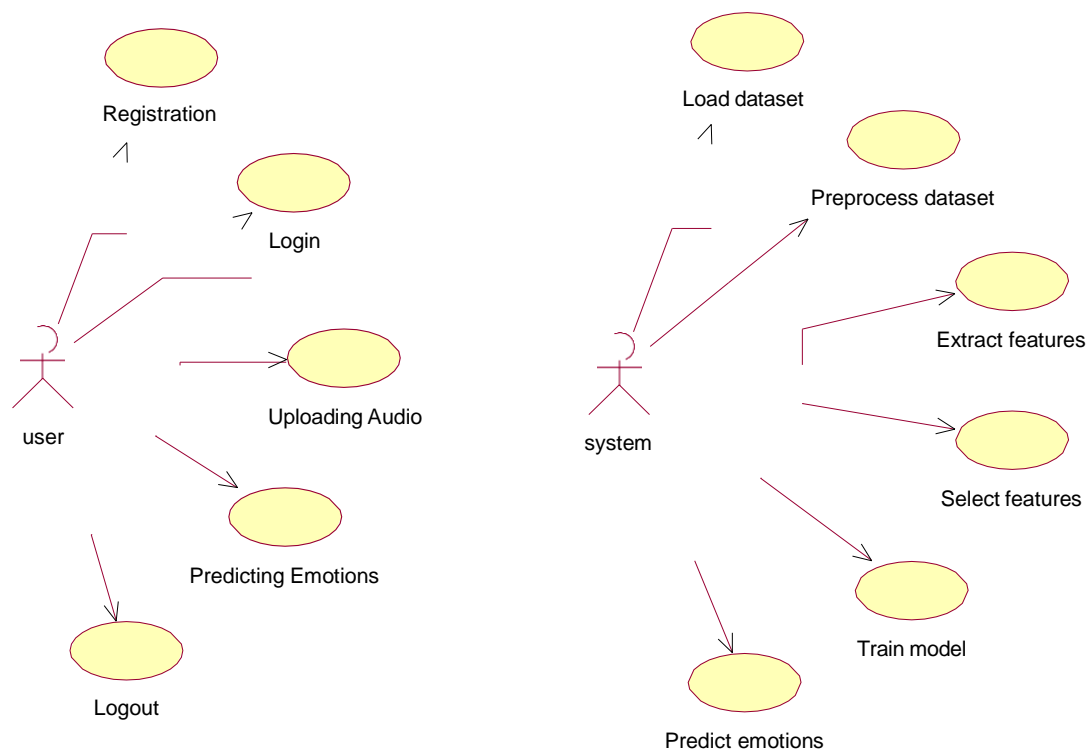


Fig.No 3.1 Usecase Diagram

CLASS DIAGRAM:

In software engineering, a class diagram in the Unified Modeling Language (UML) is a type of static structure diagram that describes the structure of a system by showing the system's classes, their attributes, operations (or methods), and the relationships among the classes. It explains which class contains information.



Fig.No 3.2 Class Diagram

SEQUENCE DIAGRAM:

A sequence diagram in Unified Modeling Language (UML) is a kind of interaction diagram that shows how processes operate with one another and in what order. It is a construct of a Message Sequence Chart. Sequence diagrams are sometimes called event diagrams, event scenarios, and timing diagrams.

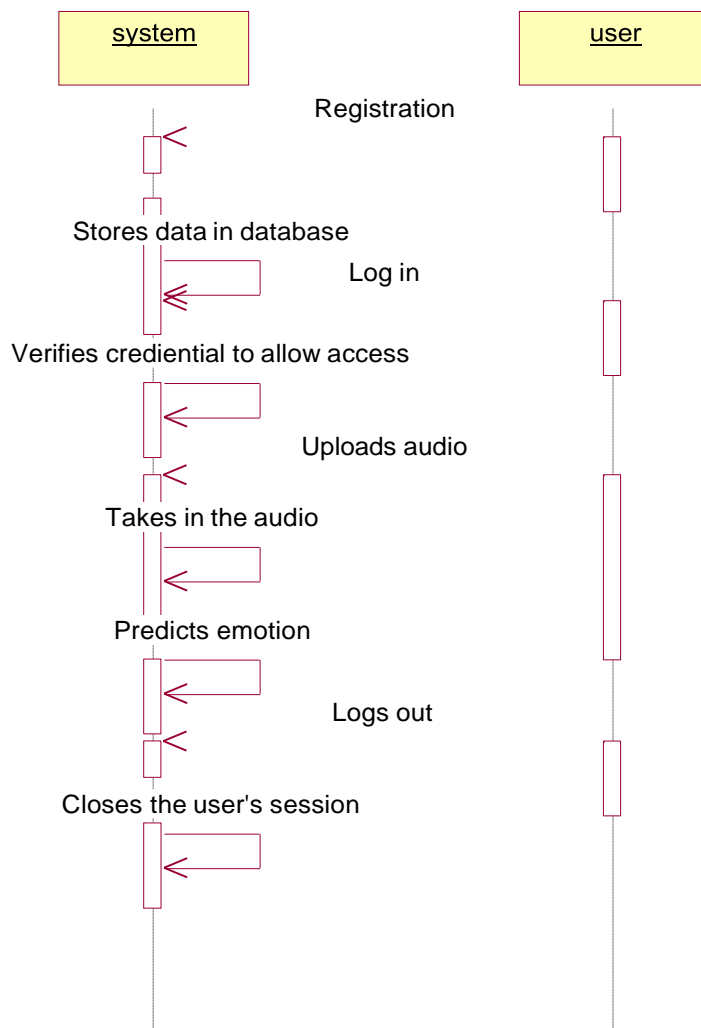


Fig.No 3.3 Sequence Diagram

COLLABORATION DIAGRAM:

In collaboration diagram the method call sequence is indicated by some numbering technique as shown below. The number indicates how the methods are called one after another. We have taken the same order management system to describe the collaboration diagram. The method calls are similar to that of a sequence diagram. But the difference is that the sequence diagram does not describe the object organization where as the collaboration diagram shows the object organization.

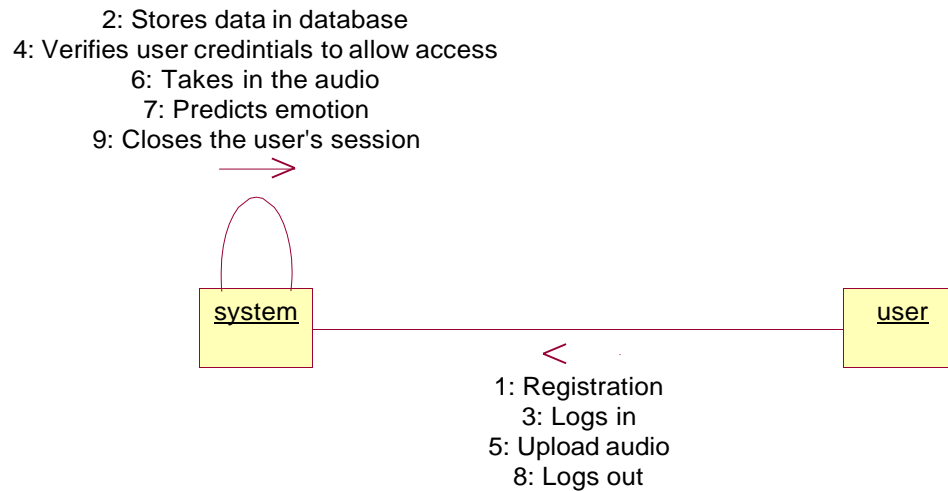


Fig.No 3.4 Collaboration Diagram

DEPLOYMENT DIAGRAM:

Deployment diagram represents the deployment view of a system. It is related to the component diagram. Because the components are deployed using the deployment diagrams. A deployment diagram consists of nodes. Nodes are nothing but physical hardware used to deploy the application.



Fig.No 3.5 Deployment Diagram

ACTIVITY DIAGRAM:

Activity diagrams are graphical representations of workflows of stepwise activities and actions with support for choice, iteration and concurrency. In the Unified Modeling Language, activity diagrams can be used to describe the business and operational step-by-step workflows of components in a system. An activity diagram shows the overall flow of control.

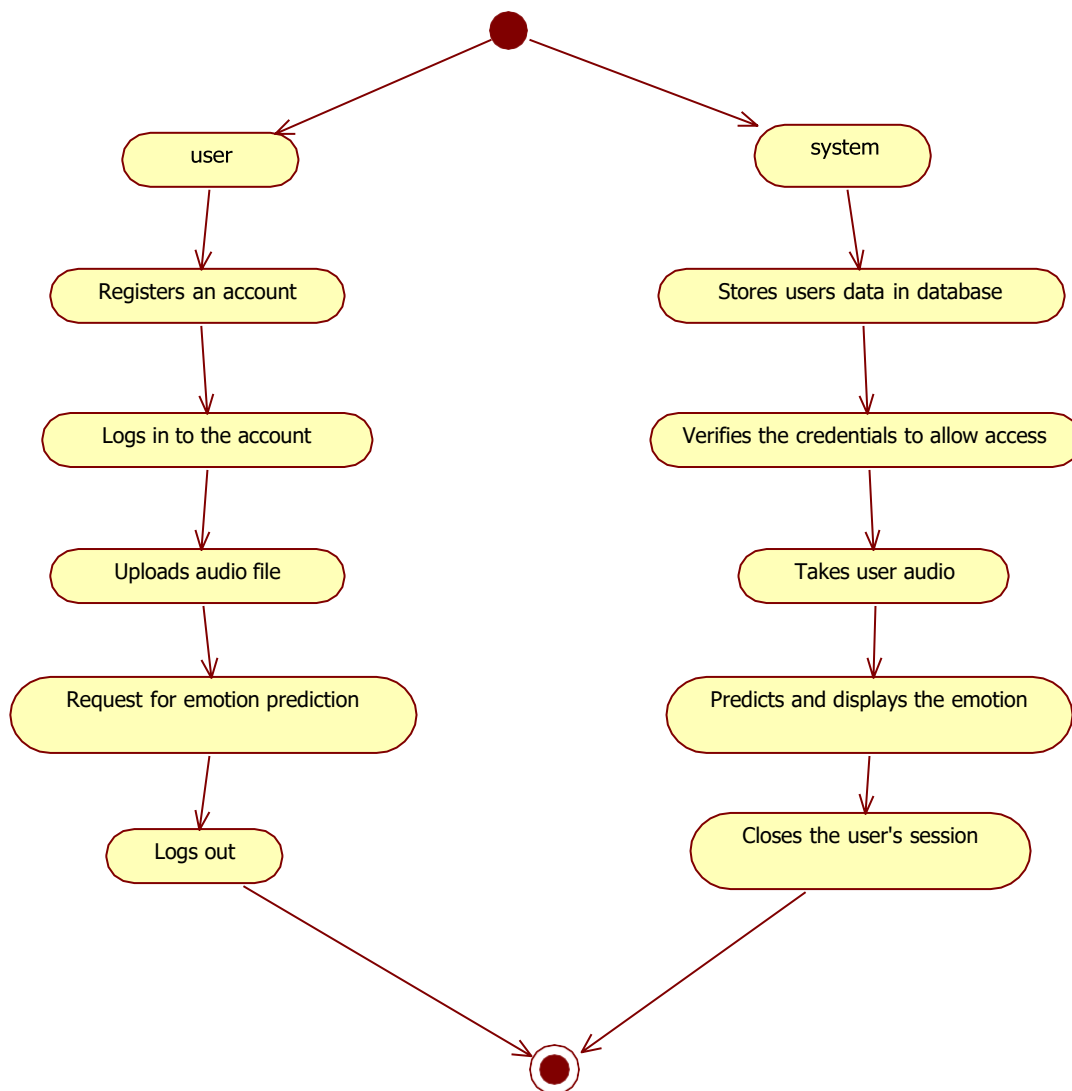


Fig.No 3.6 Activity Diagram

COMPONENT DIAGRAM:

A component diagram, also known as a UML component diagram, describes the organization and wiring of the physical components in a system. Component diagrams are often drawn to help model implementation details and double-check that every aspect of the system's required functions is covered by planned development.



Fig.No 3.7 Component Diagram

ER DIAGRAM:

An Entity–relationship model (ER model) describes the structure of a database with the help of a diagram, which is known as Entity Relationship Diagram (ER Diagram). An ER model is a design or blueprint of a database that can later be implemented as a database. The main components of E-R model are: entity set and relationship set.

An ER diagram shows the relationship among entity sets. An entity set is a group of similar entities and these entities can have attributes. In terms of DBMS, an entity is a table or attribute of a table in database, so by showing relationship among tables and their attributes, ER diagram shows the complete logical structure of a database. Let's have a look at a simple ER diagram to understand this concept.

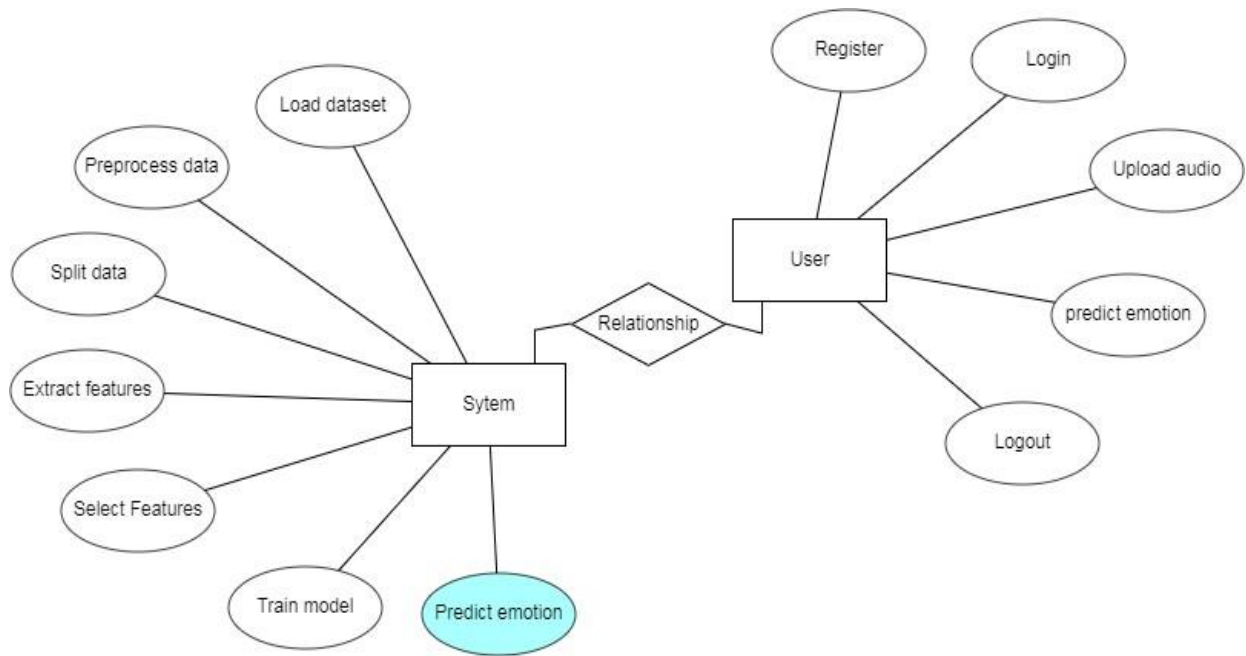


Fig.No 3.8 ER Diagram

DFD DIAGRAM:

A Data Flow Diagram (DFD) is a traditional way to visualize the information flows within a system. A neat and clear DFD can depict a good amount of the system requirements graphically. It can be manual, automated, or a combination of both. It shows how information enters and leaves the system, what changes the information and where information is stored. The purpose of a DFD is to show the scope and boundaries of a system as a whole. It may be used as a communications tool between a systems analyst and any person who plays a part in the system that acts as the starting point for redesigning a system.

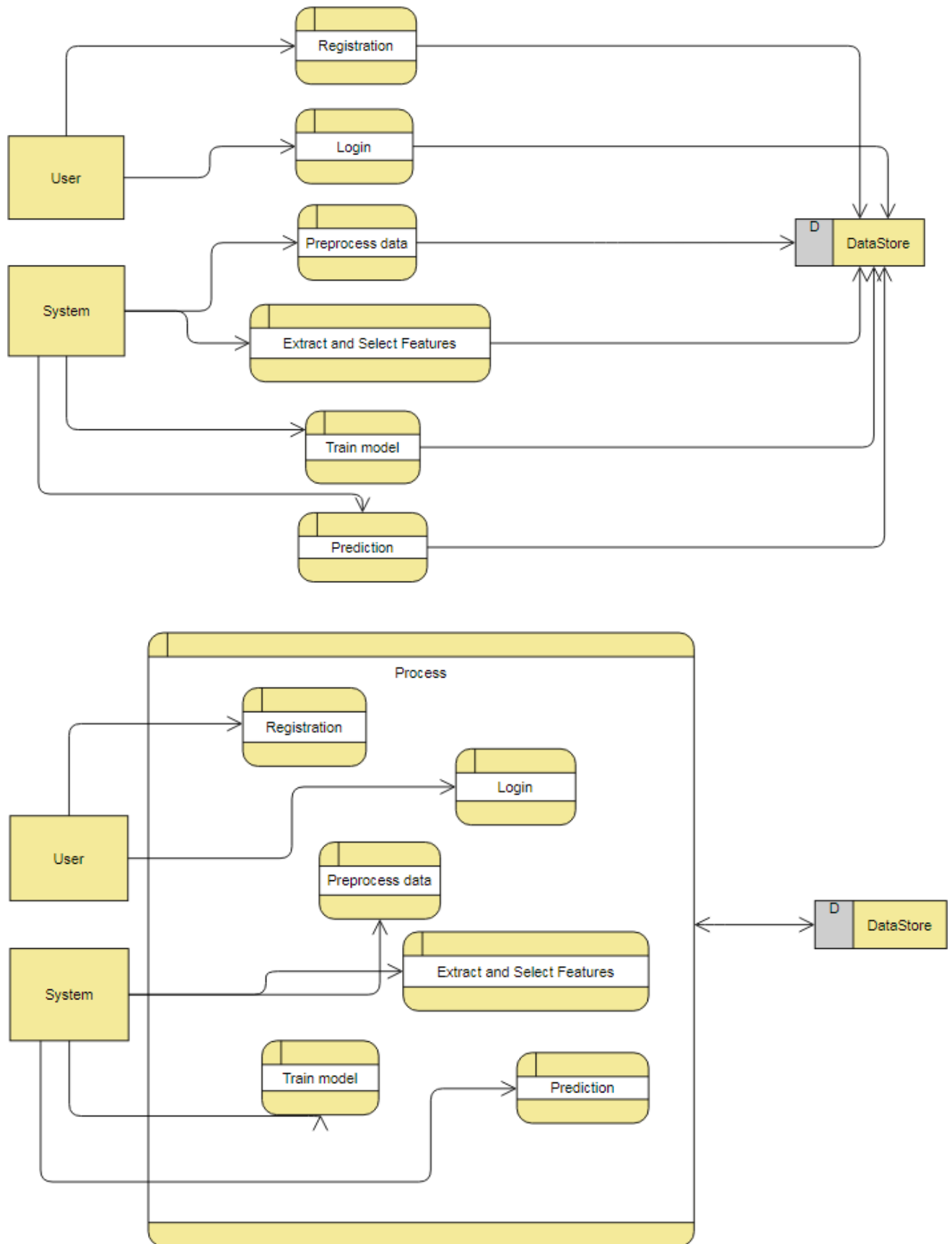


Fig.No 3.9 DFD Diagram

CHAPTER 4

DESCRIPTION OF PROPOSED SYSTEM

4.1 METHODOLOGY

Emotion recognition can have interesting applications in human robot interaction, thus paving the way for a scenario where human-machine interaction will normally take place in the real world. When a speaker expresses an emotion while adhering to an inconspicuous intonation pattern, human listeners can nevertheless perceive the emotional information through the pitch and intensity of speech. On the other hand, our aim is to capture the diverse acoustic cues that are in the speech signal and to analyze their mutual relationship to the speaker's emotion we propose a technique to recognize several basic emotions namely sadness(SAD),anger(ANG),surprise(SUR),fear(FEA), happiness(HAP) and disgust (DIS), based on the analysis of phonetic and acoustic properties.

An experimental methodology was set up, consisting of three different databases built from speakers of different areas of the world. The first database includes five European adults in the age group from 25 to 35 years (2 women and 3 men; mean age 29) from different countries of the European Union (Spain, Italy, Belgium, Romania, and France), the second group contains five Asian (Middle East) adult speakers in the age group from 19 to 45 years (2women and 3 men; mean age 31) and the third database contains recordings from American English speakers in the age group from 21 to 38 years (3 women and 2 men; mean age 28).

Six simple sentences of everyday life were chosen, namely: "What are you doing here?"- "Are you ready?"- "Come in"- "Thank you"- "You are welcome"- "Where are you?" The participants to the experiment had to repeat these six sentences for three times with a neutral (NEU) intonation, with 1 second of interval between each sentence, in order to distinguish rising-falling intonations and pitch movements.

4.2 ARCHITECTURE OF PROPOSED SYSTEM:

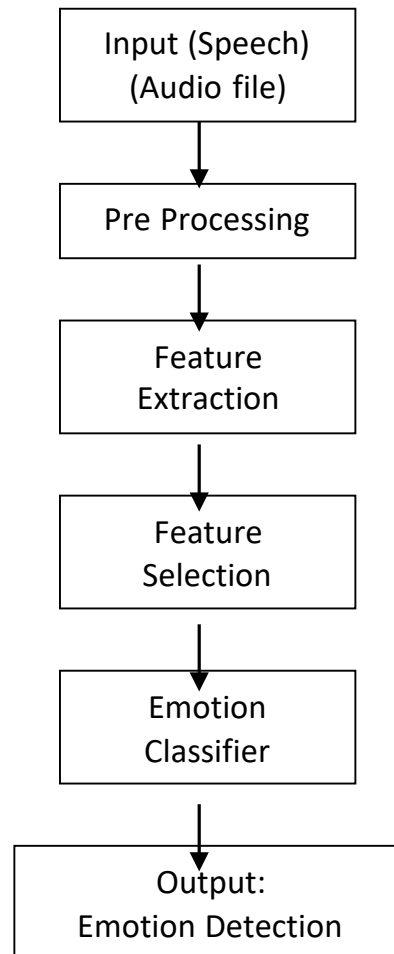


Fig.No 4.1 Architecture Diagram

4.3 DESCRIPTION OF SOFTWARE FOR IMPLEMENTATION AND TESTING PLAN OF THE PROPOSED SYSTEM:

Upload the dataset of audio (.wav files) to be read using librosa library. Data Preprocessing is a technique that is used to convert the raw data into a clean data set. Cleaning the data refers to removing the null values, filling the null values with meaningful value, removing duplicate values, removing outliers, removing unwanted attributes. If dataset contains any categorical records means convert those categorical variables to numerical values. In total our model in 5 layers deep.

The extracted features are Mel-frequency cepstral coefficients (MFCC), Chromogram, Mel scaled spectrogram in conjunction with Spectral contrast and Tonal Centroid features. We split our dataset of 1440 audio files in 2 parts, training data with 1008 audio files and testing data with 432 audio files. Here 70% of the data is taken for the training dataset. Deep learning can provide increased accuracy and decrease in computational power. Deep Neural Network (DNN) is widely used in deep learning to train models for tasks which traditional machine learning algorithms cannot do or is hard to do.

An audio is uploaded by the user (which includes speech of a person), and the model is used to predict the emotion of the speaker in the audio. A Flask architecture based web application is developed to use the model. It has 2 parts, the system and the user. There is a user registration and login management system in the UI. A new user first needs to register their details which includes name, email and the password. This user information is stored in MySQL database.

An already registered user, whose data is stored in the system's MySQL database can login to the web app using their valid credentials. Once they successfully log in, only then they are provided access to the application to predict the emotions. A Sequential () model with 5 layers is created. Train dataset is used for training the dataset using 700 epochs. The best one with respect to test accuracy is the model which we will use for prediction.

In the output layer, we use softmax for classification of the emotions. Softmax takes in a vector of numbers and converts them to probabilities which are then used for image generating results. Soft max converts logits into probabilities by taking the exponents from every output and then normalize each of these numbers by the sum of such exponents, such that the entire output vector adds up to one.

CHAPTER 5

IMPLEMENTATION DETAILS

5.1 DEVELOPMENT AND DEPLOYMENT SETUP

Upload:

Upload the dataset of audio (.wav files) to be read using librosa library.

View:

Uploaded dataset can be viewed.

Preprocessing:

Data Preprocessing is a technique that is used to convert the raw data into a clean data set. Cleaning the data refers to removing the null values, filling the null values with meaningful value, removing duplicate values, removing outliers, removing unwanted attributes. If dataset contains any categorical records means convert those categorical variables to numerical values.

Identifying Features:

The extracted features are Mel-frequency cepstral coefficients (MFCC), Chromogram, Mel scaled spectrogram in conjunction with Spectral contrast and Tonal Centroid features.

Train and Test Split:

We split our dataset of 1440 audio files in 2 parts, training data with 1008 audio files and testing data with 432 audio files. Here 70% of the data is taken for the training dataset.

Building the model:

- To understand the audio and predict emotions, we are proposing a Deep learning based method.
- Deep learning can provide increased accuracy and decrease in computational power.
- We will use Deep Neural Networks (DNN) to create the model.
- Deep Neural Network (DNN) is widely used in deep learning to train models for tasks which traditional machine learning algorithms cannot do or is hard to do.
- The model is created using 5 layers of neural networks.
- We have used dropouts to minimize the problem of overfitting.

- In order to classify the audio to the emotions, we are using softmax in the outermost layer of our DNN model.
- Softmax takes in a vector of numbers and converts them to probabilities which are then used for image generating results.
- Softmax converts logits into probabilities by taking the exponents from every output and then normalize each of these numbers by the sum of such exponents, such that the entire output vector adds up to one.

Prediction:

An audio is uploaded by the user (which includes speech of a person), and the model is used to predict the emotion of the speaker in the audio.

User Interface:

A Flask architecture based web application is developed to use the model. It has 2 parts, the system and the user. There is a user registration and login management system in the UI.

Registration:

A new user first needs to register their details which includes name, email and the password. This user information is stored in MySQL database.

User Login:

An already registered user, whose data is stored in the system's MySQL database can login to the web app using their valid credentials. Once they successfully log in, only then they are provided access to the application to predict the emotions.

Installing Python:

1. To download and install Python visit the official website of Python <https://www.python.org/downloads/> and choose your version.

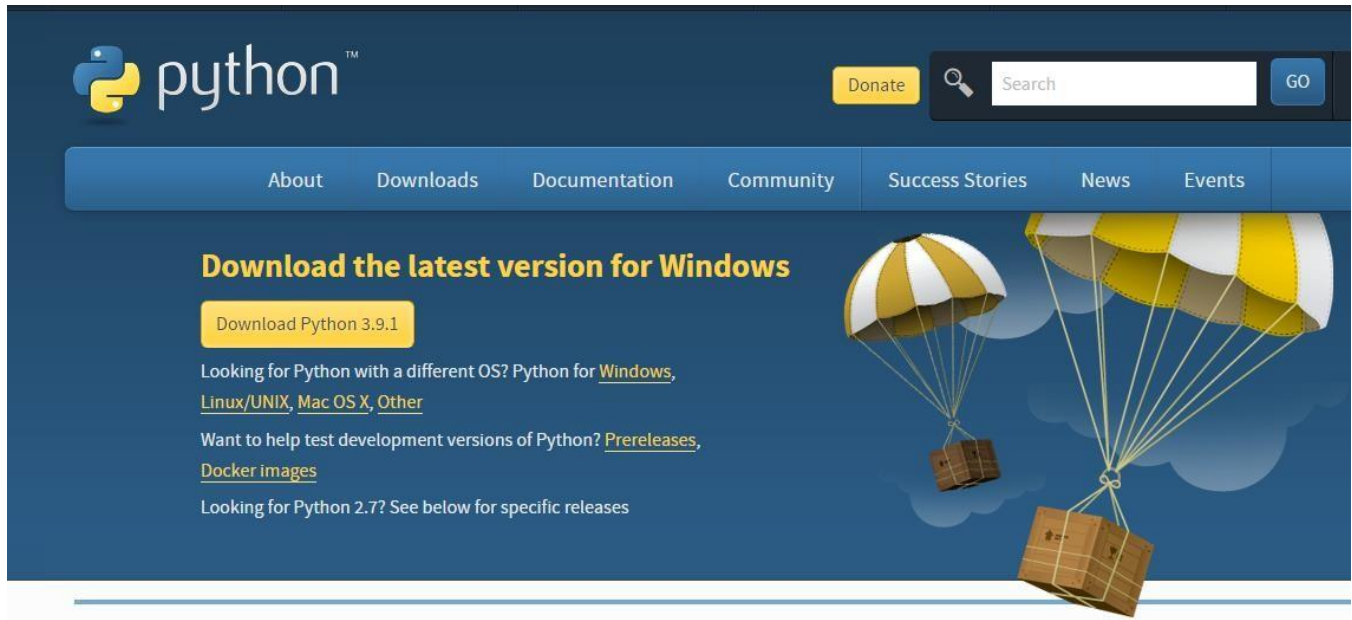


Fig.No 5.1 Python Installation

2. Once the download is complete, run the exe for install Python. Now click on Install Now.
3. You can see Python installing at this point.
4. When it finishes, you can see a screen that says the Setup was successful. Now click on "Close".

Installing PyCharm:

1. To download PyCharm visit the website <https://www.jetbrains.com/pycharm/download/> and Click the "DOWNLOAD" link under the Community Section.

2. Once the download is complete, run the exe for install PyCharm.

Download PyCharm

[Windows](#)

[Mac](#)

[Linux](#)

Professional

For both Scientific and Web Python development. With HTML, JS, and SQL support.

Download

Free trial

Community

For pure Python development

Download

Free, open-source

Fig,No 5.2 PyCharm Installation

3. The setup wizard should have started. Click “Next”.
4. On the next screen, Change the installation path if required. Click “Next”.
5. On the next screen, you can create a desktop shortcut if you want and click on “Next”.
6. Choose the start menu folder. Keep selected JetBrains and click on “Install”.
7. Wait for the installation to finish.
8. Once installation finished, you should receive a message screen that PyCharm is installed. If you want to go ahead and run it, click the “Run PyCharm Community Edition” box first and click “Finish”.
9. After you click on "Finish," the Following screen will appear.

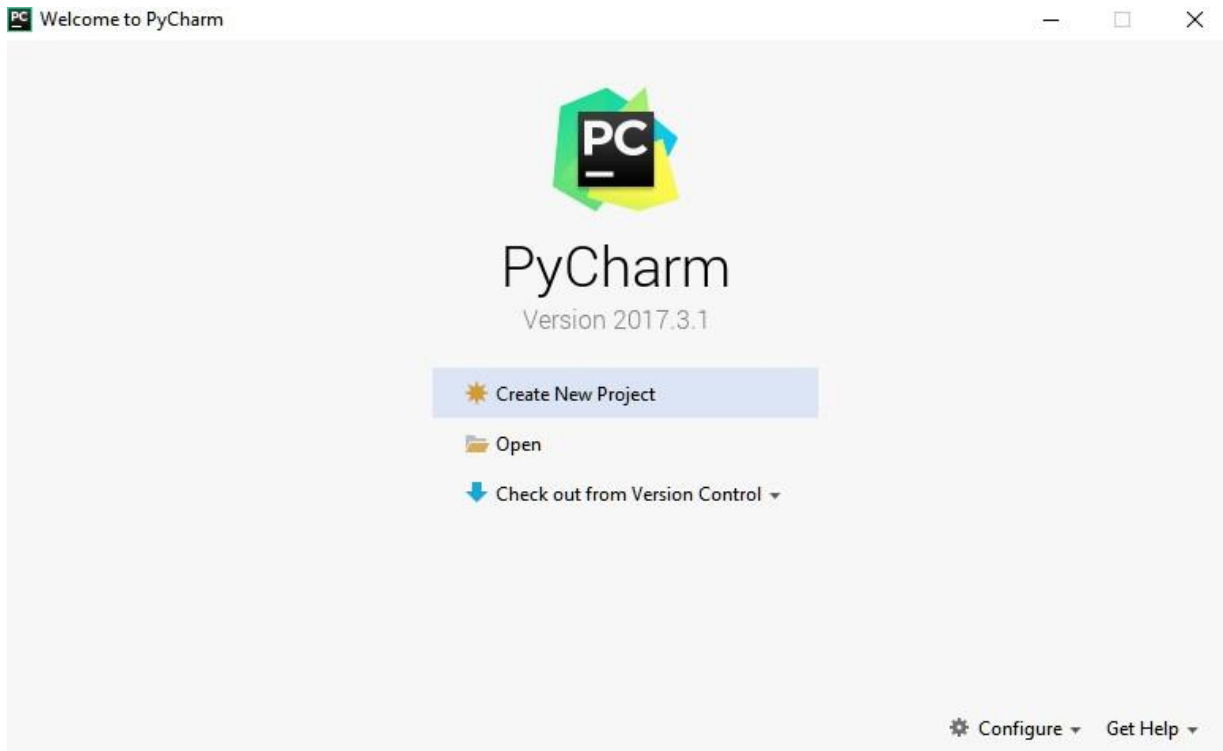


Fig.No 5.3 PyCharm Login

9. You need to install some packages to execute your project in a proper way.
10. Open the command prompt/ anaconda prompt or terminal as administrator.
11. The prompt will get open, with specified path, type “pip install package name” which you want to install (like numpy, pandas, seaborn, scikit-learn, matplotlib.pyplot)

Ex: pip install numpy

```
C:\WINDOWS\system32>pip install numpy==1.18.5
Collecting numpy==1.18.5
  Downloading numpy-1.18.5-cp36-cp36m-win_amd64.whl (12.7 MB)
    |████████████████████████████████████████| 12.7 MB 939 kB/s
ERROR: tensorboard 2.0.2 has requirement setuptools>=41.0.0, b
Installing collected packages: numpy
Successfully installed numpy-1.18.5
```

5.2 ALGORITHMS

Artificial Neural Network:

An artificial neural network is a system of hardware or software that is patterned after the working of neurons in the human brain and nervous system. Artificial neural networks are a variety of deep learning technology which comes under the broad domain of Artificial Intelligence.

Deep learning is a branch of Machine Learning which uses different types of neural networks. These algorithms are inspired by the way our brain functions and therefore many experts believe they are our best shot to moving towards real AI (Artificial Intelligence).

Some types of Neural Networks:

1. Feed forward Neural Network – Artificial Neuron

This is one of the simplest types of artificial neural networks. In a feed forward neural network, the data passes through the different input nodes until it reaches the output node.

In other words, data moves in only one direction from the first tier onwards until it reaches the output node. This is also known as a front propagated wave which is usually achieved by using a classifying activation function.

Unlike in more complex types of neural networks, there is no back propagation and data moves in one direction only. A feed forward neural network may have a single layer or it may have hidden layers.

In a feed forward neural network, the sum of the products of the inputs and their weights are calculated. This is then fed to the output. Here is an example of a single layer feedforward neural network.

2. Radial Basis Function Neural Network

A radial basis function considers the distance of any point relative to the centre. Such neural networks have two layers. In the inner layer, the features are combined with

the radial basis function.

Then the output of these features is taken into account when calculating the same output in the next time-step.

The radial basis function neural network is applied extensively in power restoration systems. In recent decades, power systems have become bigger and more complex. This increases the risk of a blackout. This neural network is used in the power restoration systems in order to restore power in the shortest possible time.

3. Multilayer Perceptron (MLP)

A multilayer perceptron has three or more layers. It is used to classify data that cannot be separated linearly. It is a type of artificial neural network that is fully connected. This is because every single node in a layer is connected to each node in the following layer.

A multilayer perceptron uses a nonlinear activation function (mainly hyperbolic tangent or logistic function).

This type of neural network is applied extensively in speech recognition and machine translation technologies.

4. Convolutional Neural Network

A convolutional neural network(CNN) uses a variation of the multilayer perceptrons. A CNN contains one or more than one convolutional layers. These layers can either be completely interconnected or pooled.

Before passing the result to the next layer, the convolutional layer uses a convolutional operation on the input. Due to this convolutional operation, the network can be much deeper but with much fewer parameters.

Due to this ability, convolutional neural networks show very effective results in image and video recognition, natural language processing, and recommender systems.

Convolutional neural networks also show great results in semantic parsing and paraphrase detection. They are also applied in signal processing and image

classification.

5. Recurrent Neural Network(RNN) – Long Short Term Memory

A Recurrent Neural Network is a type of artificial neural network in which the output of a particular layer is saved and fed back to the input. This helps predict the outcome of the layer.

The first layer is formed in the same way as it is in the feed forward network. That is, with the product of the sum of the weights and features. However, in subsequent layers, the recurrent neural network process begins.

From each time-step to the next, each node will remember some information that it had in the previous time-step. In other words, each node acts as a memory cell while computing and carrying out operations. The neural network begins with the front propagation as usual but remembers the information it may need to use later.

If the prediction is wrong, the system self-learns and works towards making the right prediction during the back propagation. This type of neural network is very effective in text-to-speech conversion technology.

6. Modular Neural Network

A modular neural network has a number of different networks that function independently and perform sub-tasks. The different networks do not really interact with or signal each other during the computation process. They work independently towards achieving the output.

As a result, a large and complex computational process can be done significantly faster by breaking it down into independent components. The computation speed increases because the networks are not interacting with or even connected to each other.

7. Sequence-To-Sequence Models

A sequence to sequence model consists of two recurrent neural networks. There's an encoder that processes the input and a decoder that processes the output. The encoder and decoder can either use the same or different parameters. This model is

particularly applicable in those cases where the length of the input data is not the same as the length of the output data.

Sequence-to-sequence models are applied mainly in chatbots, machine translation, and question answering systems.

Let us consider the basic part of any Neural Network:

- 1. Input Layer:** It is the layer where we provide the input for the model. The number of features our input has is equal to the number of neuron in the input layer.
- 2. Hidden Layer:** The input features are transferred to the hidden layer(s) where different processes/activities takes place. There can be multiple hidden layers. The layers undergoes mathematical operations like matrix multiplication, convolutions, pooling etc. along with an activation function.
- 3. Output Layer:** They layer which is used to generate probability scores using sigmoid or softmax functions which is then converted to the output of our model.

Padding:

To handle the edge pixels there are several approaches:

- Losing the edge pixels
- Padding with zero value pixels
- Reflection padding

Reflection padding is by far the best approach, where the number of pixels needed for the convolutional kernel to process the edge pixels are added onto the outside copying the pixels from the edge of the image. For a 3x3 kernel, one pixel needs to be added around the outside, for a 7x7 kernel then three pixels would be reflected around the outside. The pixels added around each side is the dimension, halved and rounded down.

Traditionally in many research papers, the edge pixels are just ignored, which loses a small proportion of the data and this gets increasing worse if there are many deep convolutional layers. For this reason, I could not find existing diagrams to easily convey some of the points here without being misleading and confusing stride 1

convolutions with stride 2 convolutions.

With padding, the output from an input of width w and height h would be width w and height h (the same as the input with a single input channel), assuming the kernel takes a stride of one pixel at a time.

Strides:

It is common to use a stride two convolution rather than a stride one convolution, where the convolutional kernel strides over 2 pixels at a time, for example our 3×3 kernel would start at position (1, 1), then stride to (1, 3), then to (1, 5) and so on, halving the size of the output channel/feature map, compared to the convolutional kernel taking strides of one. With padding, the output from an input of width w , height h and depth 3 would be the ceiling of width $w/2$, height $h/2$ and depth 1, as the kernel outputs a single summed output from each stride.

For example, with an input of $3 \times 64 \times 64$ (say a 64×64 RGB three channel image), one kernel taking strides of two with padding the edge pixels, would produce a channel/feature map of 32×32 .

The first step of creating and training a new convolutional neural network (Convnet) is to define the network architecture. This topic explains the details of Convnet layers, and the order they appear in a ConvNet. For a complete list of deep learning layers and how to create them, see [List of Deep Learning Layers](#). To learn about LSTM networks for sequence classification and regression, see [Long Short-Term Memory Networks](#). To learn how to create your own custom layers, see [Define Custom Deep Learning Layers](#). The network architecture can vary depending on the types and numbers of layers included.

Image Input Layer:

Create an image input layer using `image input layer`. An image input layer inputs images to a network and applies data normalization. Specify the image size using the `input Size` argument. The size of an image corresponds to the height, width, and the number of color channels of that image. For example, for a grayscale image, the number of channels is 1, and for a color image it is 3.

Convolution Layer:

Convolutional layers are the major building blocks used in convolutional neural networks.

A convolution is the simple application of a filter to an input that results in an activation. Repeated application of the same filter to an input results in a map of activations called a feature map, indicating the locations and strength of a detected feature in an input, such as an image.

The innovation of convolutional neural networks is the ability to automatically learn a large number of filters in parallel specific to a training dataset under the constraints of a specific predictive modeling problem, such as image classification. The result is highly specific features that can be detected anywhere on input images.

Pooling Layer:

It is common to periodically insert a Pooling layer in-between successive Convolution layers in a CNN architecture. Its function is to progressively reduce the spatial size of the representation to reduce the amount of parameters and computation in the network, and hence to also control overfitting. The Pooling Layer operates independently on every depth slice of the input and resizes it spatially, using the MAX operation. The most common form is a pooling layer with filters of size 2x2 applied with a stride of 2 downsamples every depth slice in the input by 2 along both width and height, discarding 75% of the activations. Every MAX operation would in this case be taking a max over 4 numbers (little 2x2 region in some depth slice). The depth dimension remains unchanged.

Input Sequence Layer:

A sequence input layer inputs sequence data to a network. In the general case, input sequences and output sequences have different lengths (e.g. machine translation) and the entire input sequence is required in order to start predicting the target. This requires a more advanced setup, which is what people commonly refer to when mentioning "sequence to sequence models" with no further context. Here's how it works:

In the general case, input sequences and output sequences have different lengths (e.g. machine translation) and the entire input sequence is required in order to start predicting the target. This requires a more advanced setup, which is what people commonly refer to when mentioning "sequence to sequence models". Here's how it works:

- A RNN layer (or stack thereof) acts as "encoder": it processes the input sequence and returns its own internal state. Note that we discard the outputs of the encoder RNN, only recovering the state.
- Another RNN layer (or stack thereof) acts as "decoder": it is trained to predict the next characters of the target sequence, given previous characters of the target sequence.

LSTM Layer:

Recurrent Neural Networks suffer from short-term memory. If a sequence is long enough, they'll have a hard time carrying information from earlier time steps to later ones. So if you are trying to process a paragraph of text to do predictions, RNN's may leave out important information from the beginning.

During back propagation, recurrent neural networks suffer from the vanishing gradient problem. Gradients are values used to update a neural networks weights. The vanishing gradient problem is when the gradient shrinks as it back propagates through time. If a gradient value becomes extremely small, it doesn't contribute too much learning.

With the recent breakthroughs that have been happening in data science, it is found that for almost all of these sequence prediction problems, Long short Term Memory networks, a.k.a LSTMs have been observed as the most effective solution.

LSTMs have an edge over conventional feed-forward neural networks and RNN in many ways. This is because of their property of selectively remembering patterns for long durations of time.

Fully Connected Layer:

Fully connected layers connect every neuron in one layer to every neuron in

another layer. The flattened matrix goes through a fully connected layer to classify the images.

Convolutional layers act to detect features that help classification, by picking up edges and curves and then from there detecting shapes and from there picking out ears for example, but as they can only filter images a dense layer is required to look at the output of the final convolutional neurons/filters and output a number (or numbers if one hot encoding is being used) as the classification. The outputs of all the neurons/filters in the last convolutional layer are joined together and then are “flattened” into 1D data ie it all becomes one row of data instead of the rows and columns of the data. The 1D data then acts as the input to the neuron(s) of the fully connected layer which performs a dot product of this input data and the neuron’s weights to produce a single number as output (a single number per neuron).

Output Layers:

Softmax and Classification Layers:

Softmax converts logits into probabilities by taking the exponents from every output and then norms each of these numbers by the sum of such exponents, such that the entire output vector adds up to one – every probability should be one. Generally, cross-entropy loss is the loss of such a problem in several classes. In the last layer of an image classification network such as CNN (e.g. VGG16) used in ImageNet competitions, softmax is also applied

A softmax layer applies a softmax function to the input. A classification layer computes the cross-entropy loss for multi-class classification problems with mutually exclusive classes. Create a classification layer using classification Layer. For classification problems, a softmax layer and then a classification layer must follow the final fully connected layer. The softmax function is also known as the normalized exponential and can be considered the multi-class generalization of the logistic sigmoid function. For typical classification networks, the classification layer must follow the softmax layer. In the classification layer, train Network takes the values from the Softmax function.

5.3 TESTING

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/or 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 test. Each test type addresses a specific testing requirement.

TYPES OF TESTS:

Unit testing:

Unit testing involves the design of test cases that validate that the internal program logic is functioning properly, and that program inputs produce valid outputs. All decision branches and internal code flow should be validated. It is the testing of individual software units of the application. It is done after the completion of an individual unit before integration. This is a structural testing, that relies on knowledge of its construction and is invasive. Unit tests perform basic tests at component level and test a specific business process, application, and/or system configuration. Unit tests ensure that each unique path of a business process performs accurately to the documented specifications and contains clearly defined inputs and expected results.

Integration testing:

Integration tests are designed to test integrated software components to determine if they actually run as one program. Testing is event driven and is more concerned with the basic outcome of screens or fields. Integration tests demonstrate that although the components were individually satisfactory, as shown by successfully unit testing, the combination of components is correct and consistent. Integration testing is specifically aimed at exposing the problems that arise from the combination of components.

Functional test:

Functional tests provide systematic demonstrations that functions tested are available as specified by the business and technical requirements, system documentation, and user manuals.

Functional testing is centered on the following items:

Valid Input : identified classes of valid input must be accepted.

Invalid Input : identified classes of invalid input must be rejected.

Functions : identified functions must be exercised.

Output : identified classes of application outputs must be exercised.

Systems/Procedures: interfacing systems or procedures must be invoked.

Organization and preparation of functional tests is focused on requirements, key functions, or special test cases. In addition, systematic coverage pertaining to identify Business process flows; data fields, predefined processes, and successive processes must be considered for testing. Before functional testing is complete, additional tests are identified and the effective value of current tests is determined.

SYSTEMTEST:

System testing ensures that the entire integrated software system meets requirements. It tests a configuration to ensure known and predictable results. An example of system testing is the configuration oriented system integration test. System testing is based on process descriptions and flows, emphasizing pre-driven process links and integration points.

White Box Testing

White Box Testing is a testing in which the software tester has knowledge of the inner workings, structure and language of the software, or at least its purpose. It is used to test areas that cannot be reached from a black box level.

Black Box Testing

Black Box Testing is testing the software without any knowledge of the inner workings, structure or language of the module being tested. Black box tests, as most other kinds of tests, must be written from a definitive source document, such as specification or requirements document. It is a testing in which the software under test is treated, as a black box .you cannot “see” into it. The test provides inputs and responds to outputs without

considering how the software works.

Unit Testing:

Unit testing is usually conducted as part of a combined code and unit test phase of the software lifecycle, although it is not uncommon for coding and unit testing to be conducted as two distinct phases.

Test strategy and approach

Field testing will be performed manually and functional tests will be written in detail.

Test objectives

- All field entries must work properly.
- Pages must be activated from the identified link.
- The entry screen, messages and responses must not be delayed.

Features to be tested

- Verify that the entries are of the correct format
- No duplicate entries should be allowed
- All links should take the user to the correct page.

Integration Testing

Software integration testing is the incremental integration testing of two or more integrated software components on a single platform to produce failures caused by interface defects.

The task of the integration test is to check that components or software applications, e.g. components in a software system or – one step up – software applications at the company level – interact without error.

Test Results: All the test cases mentioned above passed successfully. No defects encountered.

Acceptance Testing

User Acceptance Testing is a critical phase of any project and requires significant participation by the end user. It also ensures that the system meets the functional requirements.

Test Results: All the test cases mentioned above passed successfully. No defects encountered.

CHAPTER 6

RESULTS AND DISCUSSION:

- The proposed scheme presented an approach to recognize the emotion from the human speech.
- This approach has been implemented by the using the neural networks.
- We have successfully developed a deep learning model using the deep neural network architecture to predict the emotions of the speaker in an audio.
- We have famed our project in a web based application using the Flask architecture. The UI also includes user registration system.
- We were able to get a test accuracy of 73.4% using the trained model.

**The train vs test accuracies of our model over 700 epochs are shown below:
The model gets the best test accuracy of 73.4%**

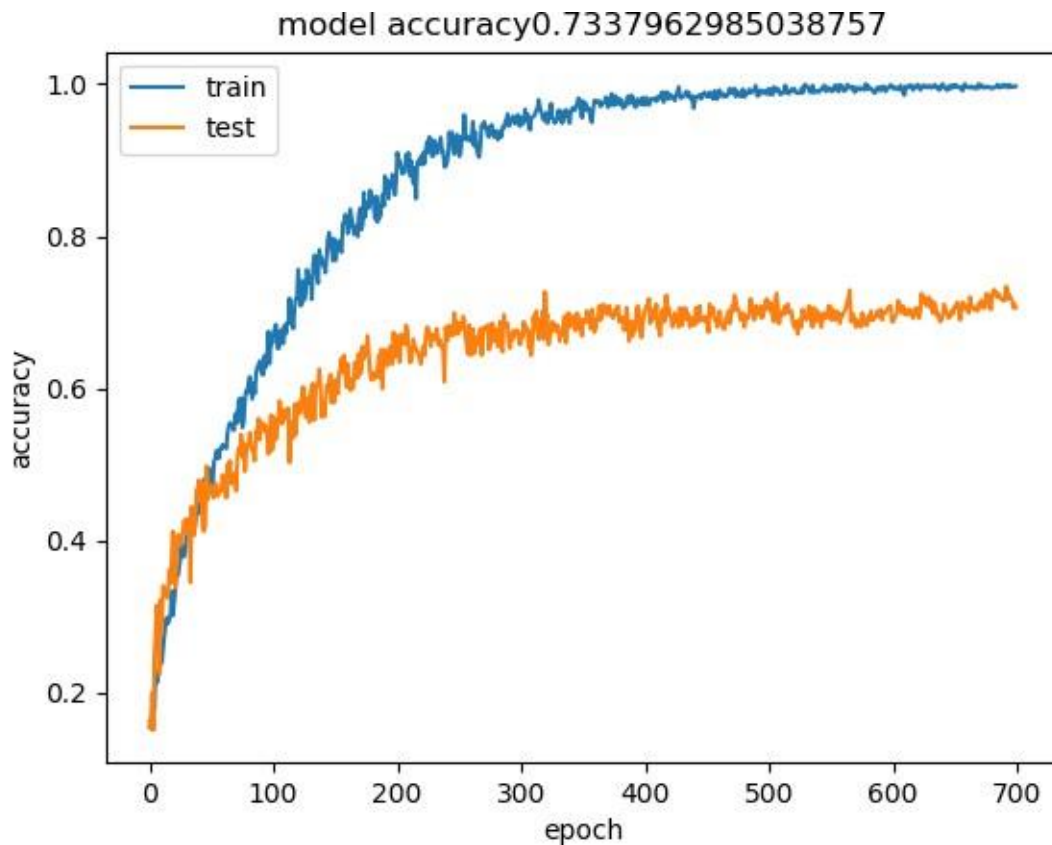


Fig.No 6.1 model accuracy 1

We previously tried other models too, with different accuracies.

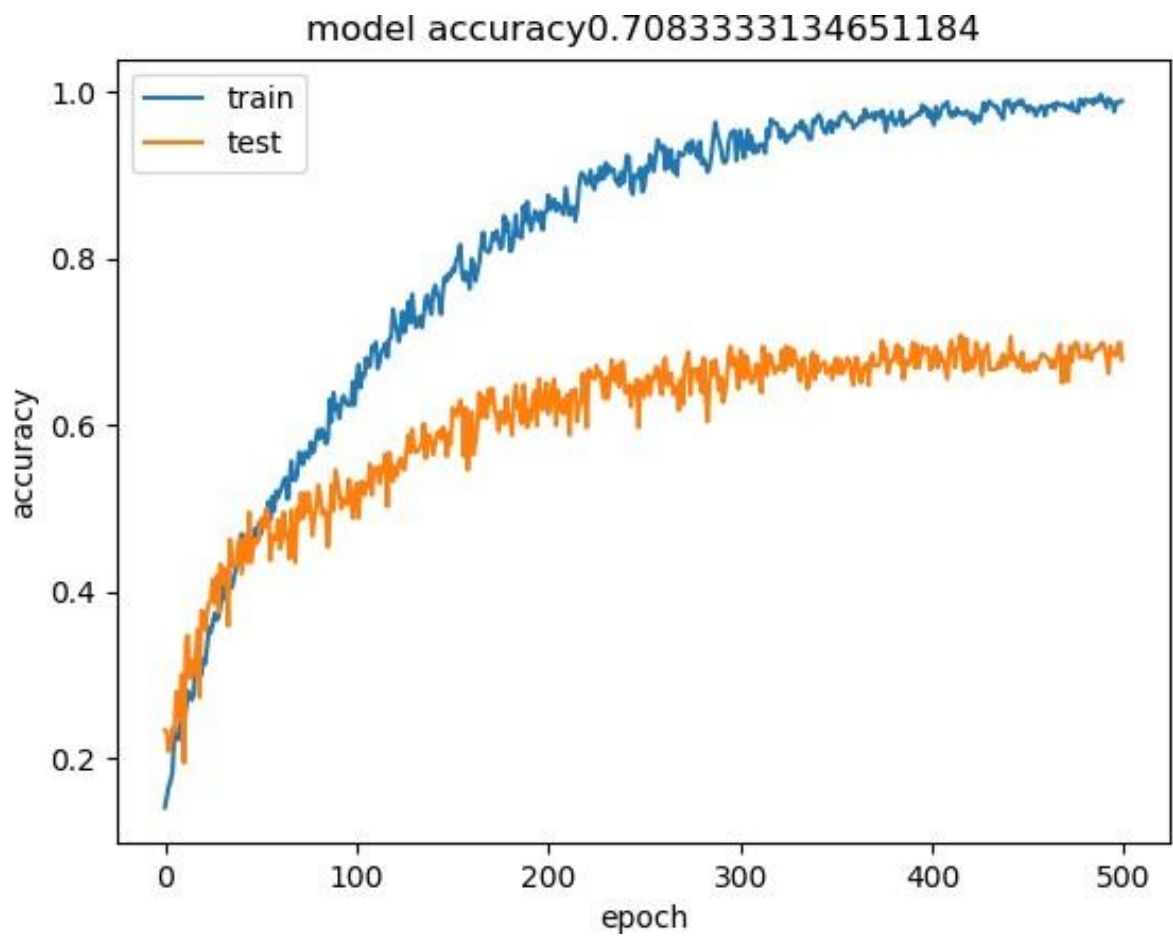


Fig.No 6.2 model accuracy 2

And this is any other version that we developed with DNN.

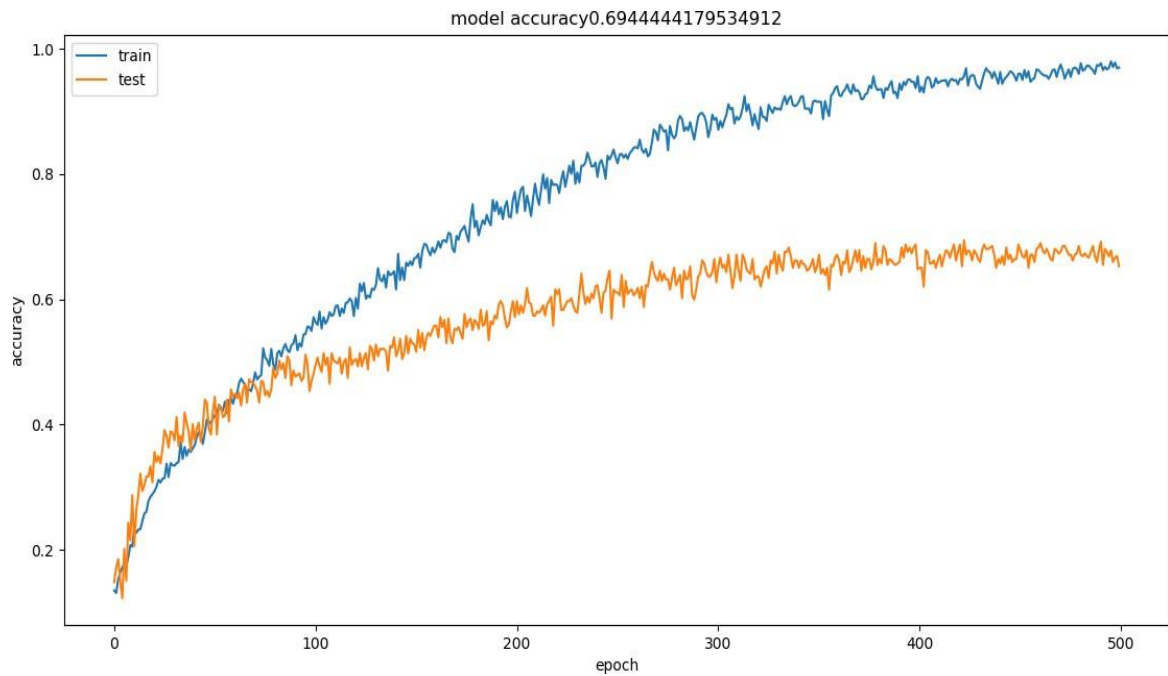


Fig.No 6.3 model accuracy 3

Chapter 7

CONCLUSION

7.1 CONCLUSION:

The proposed scheme presented an approach to recognize the emotion from the human speech. This approach has been implemented by the using the neural networks. We have successfully developed a deep learning model using the deep neural network architecture to predict the emotions of the speaker in an audio. We have famed our project in a web based application using the Flask architecture. The UI also includes user registration system. We were able to get a test accuracy of 73.4% using the trained model.

Please note that emotion prediction is subjective and the emotions rated by a person for the same audio can differ from person to person. This is also the reason why the algorithm which is trained on human rated emotions can generate erratic results sometimes. The model was trained of RAVDESS dataset, so the accent of the speaker can also lead to erratic results as the model is only trained on North American accent database.

7.2 FUTURE SCOPE:

The ability to record a voice live and to predict the emotions in real time as the speaker is speaking. It also requires the knowledge of signal processing as the voice needs to be cleaned of all the unwanted noises in them before prediction.

CHAPTER 8

REFERENCES:

- [1] Ashish B. Ingale & D. S. Chaudhari (2012). Speech Emotion Recognition. International Journal of Soft Computing and Engineering (IJSCE) ISSN: 2231-2307, Volume-2 Issue-1, March 2012.
- [2] Benk, Sal & Elmir, Youssef & Dennai, Abdeslem. (2019). A Study on Automatic Speech Recognition. 10. 77-85. 10.6025/jitr/2019/10/3/77-85.
- [3] B. Nakisa, M. N. Rastgoo, A. Rakotonirainy, F. Maire, and V. Chandran, "Automatic Emotion Recognition Using Temporal Multimodal Deep Learning," IEEE Access, 2020.
- [4] Chenchen Huang, Wei Gong, Wenlong Fu, Dongyu Feng, "A Research of Speech Emotion Recognition Based on Deep Belief Network and SVM", *Mathematical Problems in Engineering*, vol. 2014, Article ID 749604, 7 pages, 2014. <https://doi.org/10.1155/2014/749604>
- [5] C.-J. Yang, N. Fahier, W.-C. Li, and W.-C. Fang, "A Convolution Neural Network Based Emotion Recognition System using Multimodal Physiological Signals," in 2020 IEEE International Conference on Consumer Electronics-Taiwan (ICCE-Taiwan), 2020.
- [6] I. Chiriacescu, "Automatic Emotion Analysis Based On Speech", M.Sc. THESIS Delft University of Technology, 2009.
- [7] J. Han, Z. Zhang, F. Ringeval and B. Schuller, "Reconstruction-error-based learning for continuous emotion recognition in speech", *ICASSP. IEEE*, pp. 2367-2371, 2017.
- [8] Livingstone SR, Russo FA (2018) The Ryerson Audio-Visual Database of Emotional Speech and Song (RAVDESS): A dynamic, multimodal set of facial and

vocal expressions in North American English. PLoS ONE 13(5): e0196391.

[9] M. A. Asghar, M. J. Khan, M. Rizwan, R. M. Mehmood, and S.-H. Kim, "An Innovative Multi-Model Neural Network Approach for Feature Selection in Emotion Recognition Using Deep Feature Clustering," *Sensors*, vol. 20, p. 3765, 2020.

[10] M. E. Ayadi, M. S. Kamel, F. Karray, "Survey on Speech Emotion Recognition: Features, Classification Schemes, and Databases", *Pattern Recognition* 44, PP.572-587, 2011.

[11] Mustaqeem, M.; Sajjad, M.; Kwon, S. Clustering based speech emotion recognition by incorporating learned features and deep BiLSTM. *IEEE Access* **2020**.

[12] P.Shen, Z. Changjun, X. Chen, "Automatic Speech Emotion Recognition Using Support Vector Machine", *International Conference On Electronic And Mechanical Engineering And Information Technology*, 2011.

[13] S. Emerich, E. Lupu, A. Apatean, "Emotions Recognitions by Speech and Facial Expressions Analysis", *17th European Signal Processing Conference*, 2009.

[14] Szegedy, Christian & Toshev, Alexander & Erhan, Dumitru. (2013). Deep Neural Networks for Object Detection. 1-9.

[15] T. Vogt, E. Andre and J. Wagner, "Automatic Recognition of Emotions from Speech: A review of the literature and recommendations for practical realization", *LNCS 4868*, PP.75-91, 2008.

[16] W. Nie, Y. Yan, D. Song, and K. Wang, "Multi-modal feature fusion based on multi-layers LSTM for video emotion recognition," *Multimedia Tools and Applications*, pp. 1-10, 2020.

APPENDIX

A. SOURCE CODE:

Audio _wave.py:s

```
#Import libraries
import matplotlib.pyplot as plt
import wave
import pyaudio
from keras.models import load_model
import os
import glob
import librosa
import numpy as np

CHUNK = 1024*4
FORMAT = pyaudio.paInt16
CHANNELS = 1
RATE = 48000

model=load_model("model/model_final.h5")

#Extract features
def extract_features(file_name):
    X, sample_rate = librosa.load(file_name)
    #Short time fourier transformation
    stft = np.abs(librosa.stft(X))
    #Mel Frequency Cepstra coeff (40 vectors)
    mfccs=np.mean(librosa.feature.mfcc(y=X, sr=sample_rate, n_mfcc=40).T, axis=0)
    #Chromogram or power spectrum (12 vectors)
    chroma=np.mean(librosa.feature.chroma_stft(S=stft, sr=sample_rate).T, axis=0)
    #mel scaled spectrogram (128 vectors)
    mel=np.mean(librosa.feature.melspectrogram(y=X, sr=sample_rate).T, axis=0)
    # Spectral contrast (7 vectors)
    contrast=np.mean(librosa.feature.spectral_contrast(S=stft, sr=sample_rate).T,
axis=0)
    #tonal centroid features (6 vectors)

tonnetz=np.mean(librosa.feature.tonnetz(y=librosa.effects.harmonic(X),sr=sample_rate).T,
axis=0)
    return mfccs, chroma, mel, contrast, tonnetz

#generating predictions
def speech_to_emotion(filename):
    mfccs, chroma, mel, contrast, tonnetz= extract_features(filename)

    features=np.empty((0,193))
    f=np.hstack([mfccs, chroma, mel, contrast, tonnetz])
    features=np.vstack([features, f])

    his=model.predict(features)

    emotions=['neutral','calm','happy','sad','angry','fearful','disgusted','surprised']
    y_pred=np.argmax(his, axis=1)
    pred_prob=np.max(his,axis=1)
    pred_emo=(emotions[y_pred[0]],pred_prob[0])

    return pred_emo

def record_audio(record=True, file_loc=None):
```

```

if record:
    p=pyaudio.PyAudio()
    stream=p.open(format=FORMAT, channels=CHANNELS, rate=RATE, input=True,
output=True, frames_per_buffer=CHUNK)

    # create matplotlib figure and axes
    fig, ax = plt.subplots(1, figsize=(7, 4))
    # variable for plotting
    x = np.arange(0, 2 * CHUNK, 2)
    # create a line object with random data
    line = ax.plot(x, np.random.rand(CHUNK), '-', lw=2)[0]
    # basic formatting for the axes
    ax.set_title('AUDIO WAVEFORM')
    ax.set_xlabel('Samples')
    ax.set_ylabel('Volume')
    ax.set_xlim(0, 2 * CHUNK)
    plt.setp(ax, xticks=[0, CHUNK, 2 * CHUNK], yticks=[-1000, 1000])
    ax.set_xticklabels([])
    ax.set_yticklabels([])
    # show the plot
    plt.show(block=False)
    wm = plt.get_current_fig_manager()
    wm.window.attributes('-topmost', 1)
    #wm.window.attributes('-topmost', 0)

    frames = []
    for i in range(0, int(RATE / CHUNK * 5)):
        data = stream.read(CHUNK)
        frames.append(data)
        result = np.frombuffer(data, dtype=np.int16)
        line.set_ydata(result)
        fig.canvas.draw()
        fig.canvas.flush_events()
        prog=round((i*100)/(int(RATE/CHUNK*5)))
        plt.suptitle('Progress: '+str(prog)+"%")

    filename = "output.wav"

    # Save the recorded data as a WAV file
    wf = wave.open(filename, 'wb')
    wf.setnchannels(CHANNELS)
    wf.setsampwidth(p.get_sample_size(FORMAT))
    wf.setframerate(RATE)
    wf.writeframes(b''.join(frames))
    wf.close()

    main_dir = r'C:\Users\YMTS0297\PycharmProjects\Emotion Recognition using
speech\output.wav'
    pred = speech_to_emotion(main_dir)
    #fig.suptitle(pred[0])
    fig.texts=[]
    plt.title('AUDIO WAVEFORM\nPredicted Emotion: '+str(pred[0].capitalize()))

    rec=wave.open(filename, 'r')
    return rec
else:
    if file_loc:
        emo=speech_to_emotion(file_loc)[0]
        #print("The predicted emotion is: "+emo.capitalize())
        return emo.capitalize()

```

Forms.py:

```
from flask_wtf import FlaskForm
from wtforms import StringField, PasswordField, SubmitField, BooleanField
from wtforms.validators import DataRequired, Length, Email, EqualTo

class RegistrationForm(FlaskForm):
    username=StringField('Username', validators=[DataRequired(), Length(min=3, max=20)])
    email=StringField('Email', validators=[DataRequired(), Email()])
    password=PasswordField('Password', validators=[DataRequired()])
    confirm_password=PasswordField('Confirm Password', validators=[DataRequired(),
    EqualTo('password')])

    submit=SubmitField('Sign Up')

class LoginForm(FlaskForm):
    email=StringField('Email', validators=[DataRequired(), Email()])
    password=PasswordField('Password', validators=[DataRequired()])
    remember=BooleanField('Remember Me')
    submit=SubmitField('Log In')
```

SPEECH_TRAIN.PY:

```
#Import libraries
import glob
import os
import librosa
import numpy as np
from keras.models import Sequential
from keras.layers import Dense, Activation
from keras.layers import Dropout
from sklearn.model_selection import train_test_split
from sklearn.metrics import confusion_matrix, accuracy_score
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
from tensorflow.keras.callbacks import ReduceLROnPlateau, ModelCheckpoint,
EarlyStopping
import keras
from keras import regularizers

#Loading features and labels
X=np.load('X.npy') #features
y=np.load('y.npy') #labels

#Splitting the dataset
train_X, test_X, train_y, test_y = train_test_split(X, y, test_size=
0.3,random_state=42)

def get_network():
    input_shape = (193,)
    num_classes = 8
    keras.backend.clear_session()

    model = keras.models.Sequential()
    model.add(keras.layers.Dense(1024, activation="relu", input_shape=input_shape))
    model.add(Dropout(0.5))
    model.add(keras.layers.Dense(512, activation="relu", input_shape=input_shape))
```

```

model.add(keras.layers.Dense(256, activation="relu", input_shape=input_shape))
model.add(keras.layers.Dense(128, activation="relu", input_shape=input_shape))
model.add(keras.layers.Dense(num_classes, activation = "softmax"))
model.compile(optimizer='adam',
              loss='categorical_crossentropy',
              metrics=["accuracy"])
return model

model = get_network()

# Model Training
lr_reduce = ReduceLROnPlateau(monitor='val_accuracy', factor=0.9, patience=20,
min_lr=0.000001)
# Please change the model name accordingly.
mcp_save = ModelCheckpoint('model/hello.h5', save_best_only=True,
monitor='val_accuracy', mode='max')

#callbacks = [EarlyStopping(monitor='val_loss', mode='min', patience=20), mcp_save,
lr_reduce]

history=model.fit(train_X, train_y, epochs = 700, batch_size = 24,
validation_data=(test_X, test_y), callbacks=[mcp_save, lr_reduce])

#l, a = model.evaluate(x_test, y_test, verbose = 0)

#Plots
# Plotting the Train Valid Loss Graph

plt.plot(history.history['accuracy'])
plt.plot(history.history['val_accuracy'])
plt.title('model accuracy: '+str(max(history.history['val_accuracy'])))
plt.ylabel('accuracy')
plt.xlabel('epoch')
plt.legend(['train', 'test'], loc='upper left')
plt.show()

```

SPEECH_MAIN.PY:

```

#Import libraries
import glob
import os
import librosa
import numpy as np
from keras.models import Sequential
from keras.layers import Dense, Activation
from keras.layers import Dropout
from sklearn.model_selection import train_test_split
from sklearn.metrics import confusion_matrix, accuracy_score
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
from tensorflow.keras.callbacks import ReduceLROnPlateau, ModelCheckpoint

#Extract features
def extract_features(file_name):
    X, sample_rate = librosa.load(file_name)
    #Short time fourier transformation
    stft = np.abs(librosa.stft(X))
    #Mel Frequency Cepstra coeff (40 vectors)
    mfccs=np.mean(librosa.feature.mfcc(y=X, sr=sample_rate, n_mfcc=40).T, axis=0)
    #Chromogram or power spectrum (12 vectors)

```

```

        chroma=np.mean(librosa.feature.chroma_stft(S=stft, sr=sample_rate).T, axis=0)
        #mel scaled spectrogram (128 vectors)
        mel=np.mean(librosa.feature.melspectrogram(y=X, sr=sample_rate).T, axis=0)
        # Spectral contrast (7 vectors)
        contrast=np.mean(librosa.feature.spectral_contrast(S=stft, sr=sample_rate).T,
axis=0)
        #tonal centroid features (6 vectors)

tonnetz=np.mean(librosa.feature.tonnetz(y=librosa.effects.harmonic(X),sr=sample_rate).T,
axis=0)
        return mfccs, chroma, mel, contrast, tonnetz

#parsing audio files
def parse_audio_files(parent_dir, sub_dirs, file_ext="*.wav"):
    features, labels = np.empty((0,193)), np.empty(0)
    for label, sub_dir in enumerate(sub_dirs):
        for fn in glob.glob(os.path.join(parent_dir, sub_dir, file_ext)):
            try:
                mfccs, chroma, mel, contrast, tonnetz= extract_features(fn)
            except Exception as e:
                print("Error encountered while parsing file: ", fn)
                continue
            ext_features = np.hstack([mfccs, chroma, mel, contrast, tonnetz])
            features = np.vstack([features, ext_features])
            labels = np.append(labels, fn.split("\\")[8].split("-")[2])
    return np.array(features), np.array(labels, dtype = np.int)

#fn=glob.glob(os.path.join(main_dir, sub_dir[0], "*.wav"))[0]

#One-Hot Encoding the multi class labels
def one_hot_encode(labels):
    n_labels = len(labels)
    n_unique_labels = len(np.unique(labels))
    one_hot_encode = np.zeros((n_labels, n_unique_labels + 1))
    one_hot_encode[np.arange(n_labels), labels] = 1
    one_hot_encode=np.delete(one_hot_encode, 0, axis=1)
    return one_hot_encode

#Extracting features in X
#Storing labels in y
main_dir = r'C:\Users\YMTS0297\PycharmProjects\Emotion Recognition using
speech\Datasets\Audio_Speech_Actors_01-24'
sub_dir = os.listdir(main_dir)
print("\nCollecting features and labels.")
print("\nThis will take some time.")
features, labels = parse_audio_files(main_dir, sub_dir)
print("\nCompleted")

#save features
np.save('X', features)
#one hot encode labels
labels = one_hot_encode(labels)
np.save('y', labels)

emotions=['neutral','calm','happy','sad','angry','fearful','disgusted','surprised']
#labels2=np.array([emotions[labels[i]-1] for i in range(len(labels))])
#np.save('y2', labels2)

#Loading features and labels
#X=np.load('X.npy') #features
#y=np.load('y.npy') #labels

#Splitting the dataset
#train_X, test_X, train_y, test_y = train_test_split(X, y, test_size=

```

```

0.3,random_state=42)

#Parameters
#n_dim = train_X.shape[1]
#n_classes = train_y.shape[1]

#n_hidden_units_1 = n_dim
#n_hidden_units_2 = 400
#n_hidden_units_3 = 200
#n_hidden_units_4 = 100

#Define model
#def create_model(activation_function='relu', init_type='normal', optimizer='adam',
#dropout_rate=0.2):
#    model = Sequential()
#    #Layer 1
#    model.add(Dense(n_hidden_units_1, input_dim = n_dim, kernel_initializer=init_type,
#activation=activation_function))
#    #Layer 2
#    model.add(Dense(n_hidden_units_2,kernel_initializer=init_type,
#activation=activation_function))
#    model.add(Dropout(0.2))
#    #Layer 3
#    model.add(Dense(n_hidden_units_3, kernel_initializer=init_type,
#activation=activation_function))
#    model.add(Dropout(0.2))
#    #Layer 4
#    model.add(Dense(n_hidden_units_4, kernel_initializer=init_type,
#activation=activation_function))
#    model.add(Dropout(0.2))
#    #Output layer
#    model.add(Dense(n_classes, kernel_initializer=init_type, activation='softmax'))

#    #Model compilation
#    model.compile(loss="categorical_crossentropy", optimizer=optimizer,
#metrics=['accuracy'])
#    return model

#Create the model
#model=create_model()

# Model Training
#lr_reduce = ReduceLROnPlateau(monitor='val_accuracy', factor=0.9, patience=20,
#min_lr=0.000001)
# Please change the model name accordingly.
#mcp_save = ModelCheckpoint('model/hello.h5', save_best_only=True,
#monitor='val_accuracy', mode='max')

#Train the model
#import time
#start=time.time()
#history = model.fit(train_X, train_y, epochs=200, batch_size=4,
#validation_data=(test_X, test_y), callbacks=[mcp_save, lr_reduce])
#end=time.time()
#print(end-start)

#Prediction
#predict=model.predict(test_X, batch_size=4)

#convert PREDICTED probabilities to emotions
#emotions=['neutral','calm','happy','sad','angry','fearful','disgusted','surprised']
#y_pred=np.argmax(predict, axis=1)
#predicted emotions of test dataset
#predicted_emo=[]

```



```

#for i in range(test_y.shape[0]):
#    emo=emotions[y_pred[i]]
#    predicted_emo.append(emo)

#actual emotions of test dataset
#actual_emo=[]
#y_true=np.argmax(test_y, axis=1)
#for i in range(test_y.shape[0]):
#    emo=emotions[y_true[i]]
#    actual_emo.append(emo)

#Creating a confusion matrix
#cm=confusion_matrix(actual_emo, predicted_emo)
#index = ['angry', 'calm', 'disgust', 'fearful', 'happy', 'neutral', 'sad', 'surprised']
#columns = ['angry', 'calm', 'disgust', 'fearful', 'happy', 'neutral', 'sad',
'surprised']
#cm_df = pd.DataFrame(cm,index,columns)
#plt.figure(figsize=(10,10))
#sns.heatmap(cm_df, annot=True)

#Training accuracy
accuracy=accuracy_score(actual_emo, predicted_emo)

#Plots
# Plotting the Train Valid Loss Graph

#plt.plot(history.history['accuracy'])
#plt.plot(history.history['val_accuracy'])
#plt.title('model accuracy'+str(max(history.history['val_accuracy'])))
#plt.ylabel('accuracy')
#plt.xlabel('epoch')
#plt.legend(['train', 'test'], loc='upper left')
#plt.show()

```

MAIN.PY:

```

from flask import Flask, render_template, request, url_for, redirect, flash,
send_from_directory, session
from forms import RegistrationForm, LoginForm
import pymysql
import pymysql.cursors
import pandas as pd
import os
from audio_wave import *

APP_ROOT=os.path.dirname(os.path.abspath(__file__))
app=Flask(__name__)
app.config['UPLOAD_FOLDER']=os.path.join(APP_ROOT, 'static/image/')
app.config['SECRET_KEY']='b0b4fbefdc48be27a6123605f02b6b86'

@app.before_first_request
def initialize():
    session['loggedin']=False

@app.route('/')
@app.route('/home')
def home():
    return render_template('index.html')

@app.route('/about')
def about():
    return render_template('about.html')

@app.route('/application')

```

```

def application():
    return render_template('application.html')

@app.route('/contact')
def contact():
    return render_template('contact.html')

@app.route('/library')
def library():
    return render_template('library.html')

@app.route('/login', methods=['GET', 'POST'])
def login():
    form = LoginForm()

    #Database connection
    db=pymysql.connect(host="localhost", port=3306, user="root", password="",
db="emotion_recognition", cursorclass=pymysql.cursors.DictCursor)
    #Table as a data frame
    register=pd.read_sql_query('select * from {}'.format('register'), db)

    if form.validate_on_submit():
        all_emails=register['email']
        if form.email.data in list(all_emails):
            row=all_emails[all_emails==form.email.data]
            index=row.index[0] ### Id is (1 + index)
            if form.password.data == register['password'][index]:
                session['loggedin']=True
                flash(f'Welcome {register['name'][index]} ! You have been logged
in.', 'success')
                return redirect(url_for('application'))
            else:
                flash("You password was incorrect. Please try again with correct
password.", 'warning')
                return redirect(url_for('login'))
        else:
            flash(f"No account with the email id {form.email.data} exists. Please
register now.", 'info')
            return redirect(url_for('register'))
    return render_template('login.html', form=form)

@app.route("/register", methods=['GET', 'POST'])
def register():
    form = RegistrationForm()
    #Database connection
    db=pymysql.connect(host="localhost", port=3306, user="root", password="",
db="emotion_recognition", cursorclass=pymysql.cursors.DictCursor)
    #Table as a data frame
    register=pd.read_sql_query('select * from {}'.format('register'), db)

    if form.validate_on_submit():
        email=form.email.data
        all_emails=register['email']
        if email in list(all_emails):
            flash('Account already exists with this Email Id! Please Log In.', 'warning')
            db.close()
            return redirect(url_for('login'))
        else:
            #Insert new record
            try:
                with db.cursor() as cur:
                    sql = "INSERT INTO `register` (`name`,`email`,`password`) VALUES
(%s, %s, %s)"
                    cur.execute(sql, ({form.username.data}, {form.email.data},

```

```

{form.password.data}))
        db.commit()
    finally:
        db.close()
    flash(f'Account Created for {form.username.data} Sucessfully!', 'success')

    return redirect(url_for('login'))
    return render_template('register.html', form=form)

@app.route("/upload", methods=["POST"])
def upload():
    target=os.path.join(APP_ROOT, 'static/audio/')

    if not os.path.isdir(target):
        os.mkdir(target)
    file=request.files["myaudio"]
    filename=file.filename
    if filename=="":
        flash('No File Selected', 'danger')
        return redirect(url_for('application'))

    destination="/".join([target, filename])
    #Extension check
    ext = os.path.splitext(destination)[1]
    if (ext==".wav"):
        pass
    else:
        flash("Invalid Extenstions! Please select a .wav audio file only.",
category="danger")
        return redirect(url_for('application'))

    if not os.path.isfile(destination):
        file.save(destination)

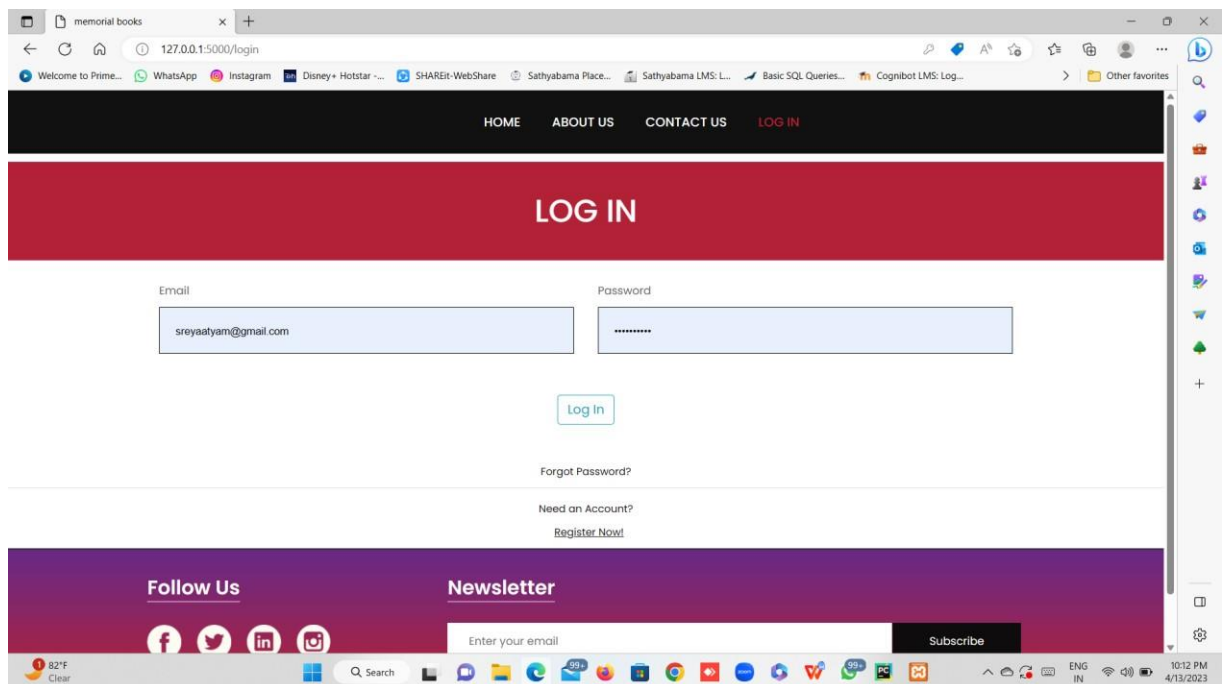
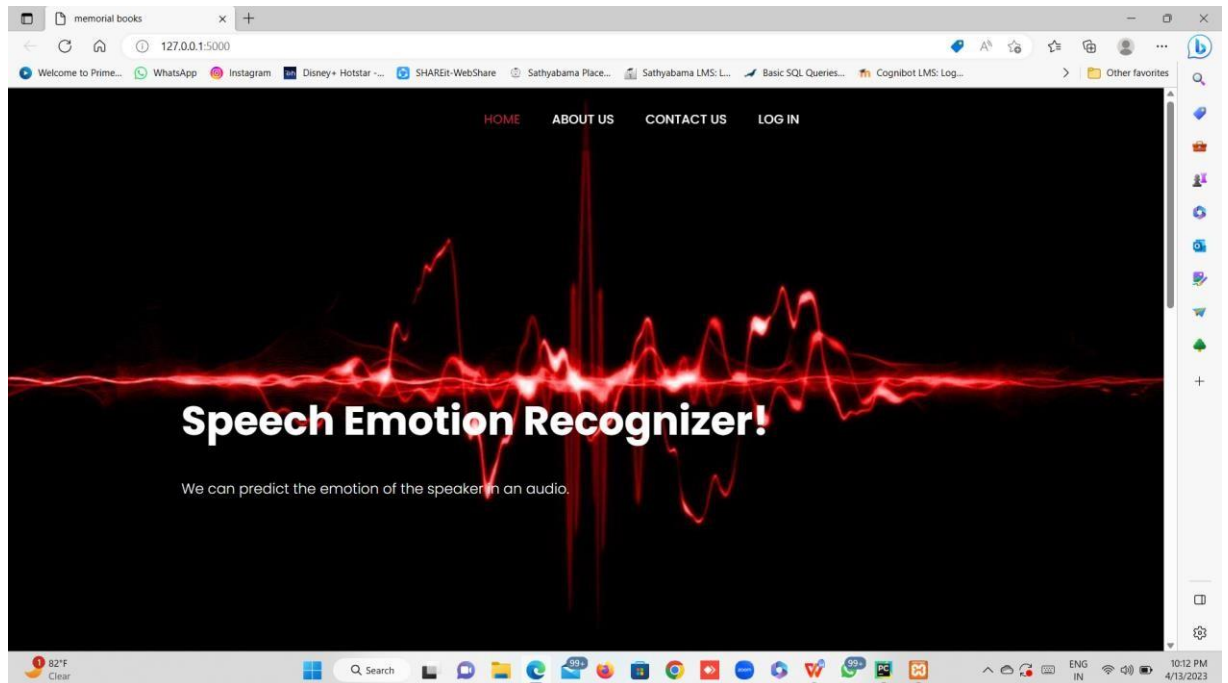
    result=record_audio(record=False, file_loc=destination)
    new_dest=('static/audio/'+str(filename))
    return render_template("upload.html", img_name=filename, emo=result,
destination=new_dest)

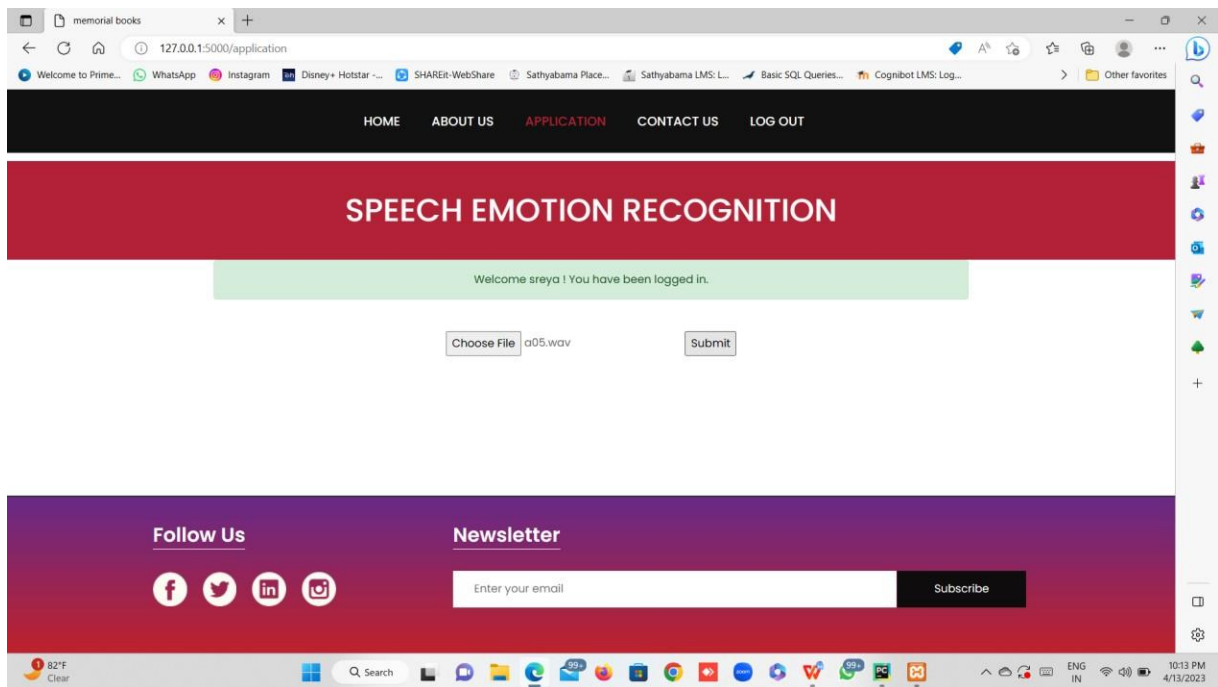
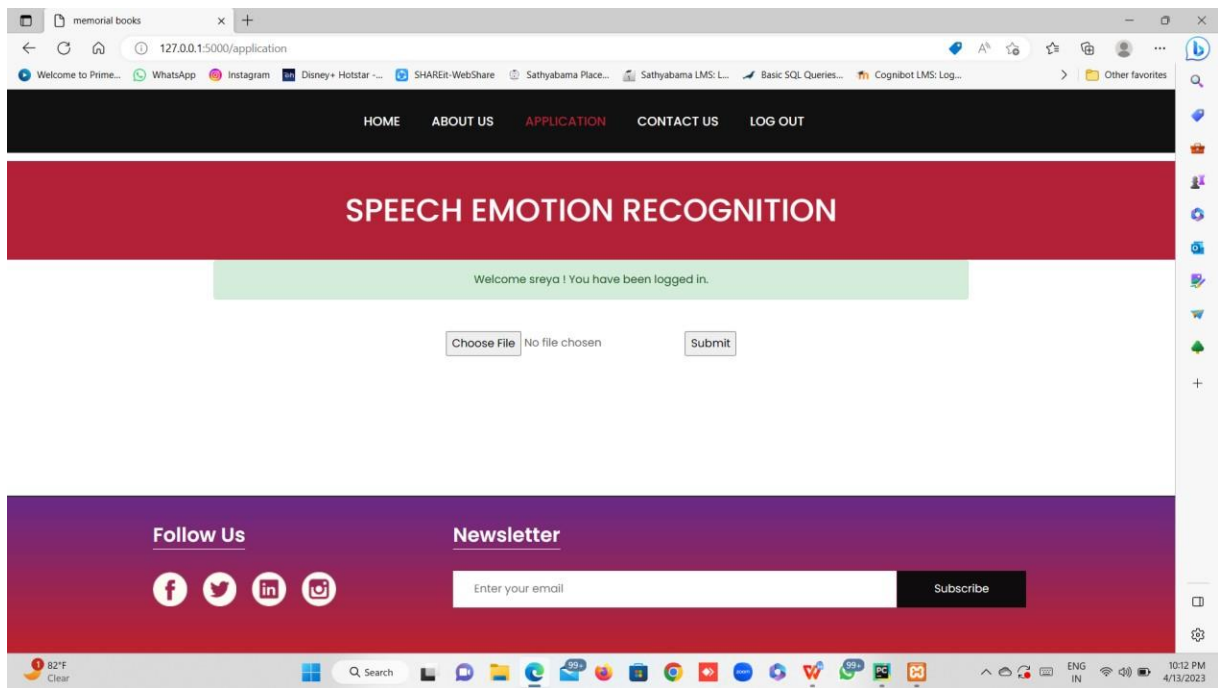
@app.route('/logout')
def logout():
    session.pop('loggedin', None)
    #flash('Logged Out Sucessfully!', 'success')
    return redirect(url_for('home'))

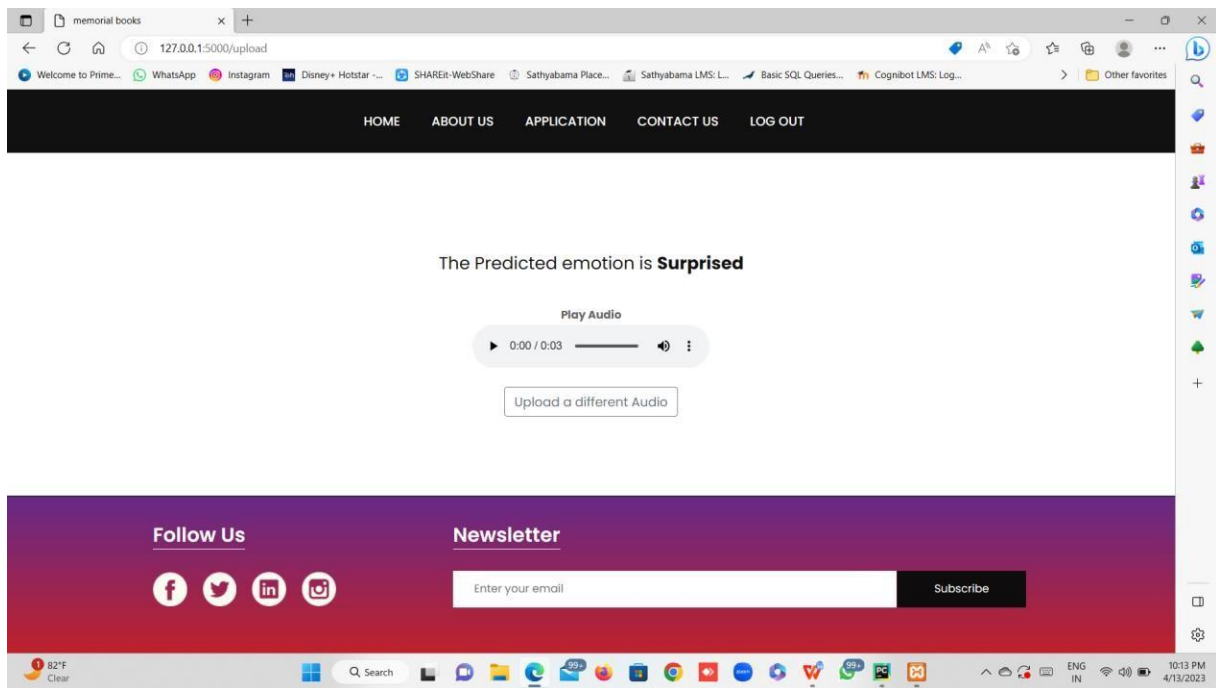
if __name__ == '__main__':
    app.run(debug=True)

```

B. SCREENSHOTS:







C.RESEARCH PAPER:

ABSTRACT:

In human device applications, speech popularity has been a subject for many years. Emotions play a totally critical position in a person's intellectual lifestyles. This is a manner of expressing one aspect or country of thoughts to others. Emotional speech (SER) may be defined as extracting the speaker's emotional country from his or her speech signal. There are several general feelings, which includes Neutrality, Anger, Happiness, Sadness, etc., in which any smart device with restricted computing sources may be programmed to understand or adjust as needed. In this work, we extract Mel frequency cepstral coefficients (MFCC), chromogram, scale Mel spectrogram with spectral evaluation and tonal capabilities. A deep neural network is used on this work to expect feelings.

Keywords: Emotion, MFCC, LDA, NSL, Deep Neural Network

PROBLEM STATEMENT:

Human speech is the maximum herbal form of self-expression. We use it everywhere from calls, emails, meetings, interviews, and many others. Since feelings play a important role in communication, their detection and evaluation are important in today's

world of remote communication. This software can be defined as a hard and fast of methodologies that seek advice from the tactics and styles of speech so that you can hit upon motion in them. We will attempt to locate the motion

person or speaker. We version a deep neural network (DNN) to create an application.

INTRODUCTION

There are many methods to speak, but signal language is one of the quickest and maximum natural approaches to talk among humans. Therefore, speech may be a totally rapid and efficient way of human interplay. Humans have a herbal ability to use all of the senses to be had to them to growth their focus of receiving a message. With all the sensitive senses, someone clearly feels the emotional state of his communication accomplice. Detecting motion is herbal for humans, however for machines it's far a totally tough mission. Therefore, the purpose of the motion of information is to use the feelings associated with knowledge in any such way as to improve the interaction among guy and machine.

In this device, the pleasant of feature extraction is immediately associated

with the suitable recognition of speech emotions. In the characteristic extraction method, the whole sentence motion became used as the gadgets for extracting the capabilities, and the content of the extraction was four components of speech movement, which were extra acoustic characteristics of time constructing, amplitude constructing, and fundamental frequency building and forming constructing. Then insert the emotional speech with an unemotional sentence from these 4, grasped by way of the law of the distribution of the emotional sign, then in line with this law suggest the emotional speech.

Deep Neural Network (DNN) has had extraordinary achievement in speech recognition and photo reputation; however, till now no research on deep neural networks has been implemented to emotional speech processing. We determined that DNN has wonderful application in processing speech motions. Therefore, this text describes a technique for routinely extracting motion capabilities from audio the use of a library file in Python. We used DNN to teach a 5-layer network to extract speech movement functions. Motion speech carries the functions of numerous consecutive frames to construct a high bandwidth function,

and a softmax classifier layer to perceive motion speech. The accuracy of the speech movement recognition take a look at reached 73.38%, which is excessive as compared to different fashions of this length.

The system getting to know methods are ok-Nearest Neighbor (KNN), hidden Markov model (HMM) and vector device (SVM), synthetic neural machine (ANN), Gaussian combination version (GMM), etc. For the category of motion.

The fundamental problems in spotting emotion in speech are the signal processing unit, wherein applicable features are extracted from the available speech signal, and any other classifier, which recognizes emotion from the speech sign. The average accuracy of most classifiers for the speaker-impartial gadget is decrease than for the speaker-structured device. Automatic popularity of emotions from human speech is growing each day because it leads to better interaction among humans and machines.

LITERATURE REVIEW

[1] Szegedy, Christian & Toshev, Alexander & Erhan, Dumitru. (2013). Deep Neural Networks for Object Detection. 1-9.

Deep Neural Networks (DNNs) have currently proven remarkable overall

performance in photograph class responsibilities [14]. In this article, we move in addition and remedy the trouble of item detection by way of DNN, that is, no longer simplest the classification, however additionally the correct localization of objects of numerous sorts. We gift a simple however effective method of item detection as a regression hassle for proscribing item covering. We define a multi-sequence system that may produce high-resolution item detection at low fee for multiple community programs. Today's overall performance approach is proven within the Easter VOC exhibit.

Summary: This paper discusses deep neural community concept and object detection using DNN.

[2] Benk, Sal & Elmir, Youssef & Dennai, Abdeslem. (2019). A Study on Automatic Speech Recognition. 10. 77-85. 10.6025/jitr/2019/10/3/77-85.

Speech is a smooth and convenient way for humans to speak, but now human beings not handiest communicate with every other, but also with various devices in our lives. The maximum important factor is to enhance. Thus, this method of communique can be used among

computer systems and human beings. This interplay is finished through interfaces, this region is called human laptop interplay (HCI). This article gives a top level view of the principle definitions of Automatic Speech Recognition (ASR), that's an important region of synthetic intelligence and should be taken into consideration in any associated studies (kind, phrase length, and many others.). It also affords a summary of important studies at some point of those years related to speech approaches, with a fashionable concept of our challenge, which can be taken into consideration as a contribution to this location of studies, and with a end associated with specifics. Upgrades have to be made. Future works

Summary: This article facilitates us to recognize and use system speech popularity, which improves human laptop interaction, and is also useful in our layout.

[3] Ashish B. Ingale & D. S. Chaudhari (2012). Speech Emotion Recognition. International Journal of Soft Computing and Engineering (IJSCE) ISSN: 2231-2307, Volume-2 Issue-1, March 2012

In human gadget applications, speech reputation has been a subject for many years. Many techniques had been

developed to come across the movement of the speech signal. This article discusses the speech recognition of emotions that have been used in preceding technologies using various classifications for emotion recognition. Classifiers are used to differentiate emotions including anger, happiness, unhappiness, surprise, neutral popularity, and so forth. In the emotional database, the popularity of the speech device is the motion patterns of the speech, and the traits of the speech to be extracted from those patterns are power, pitch, and linearity. Prediction ceps coefficient (LPCC), Mel ceps coefficient (MFCC). The type performance is primarily based on extracting capabilities. Conclusions approximately the outcomes and limitations of the speech movement reputation device are discussed in various classifications.

Summary: In this newsletter we can study the significance and necessity of various capabilities in any sound or speech, which include mfcc, honey, and different capabilities used in our sound based emotion prediction app.

[4] Chenchen Huang, Wei Gong, Wenlong Fu, Dongyu Feng, "A Research of Speech Emotion Recognition Based on Deep Belief Network and SVM", *Mathematical*

Problems

in

Engineering, vol. 2014, Article

ID 749604, 7 pages, 2014. <https://doi.org/10.1155/2014/749604>

Feature extraction is the maximum vital a part of speech motion reputation, and regarding characteristic extraction in speech movement popularity duties, this text proposes a brand new DBN extraction method using DNN to mechanically extract movement inside the speech sign. By imposing a DBN with five layers deep to extract the speech emotion characteristic and several non-stop layers to shape a better dimensional characteristic. After the features had been skilled within the DBN by using a non-linear SVM classifier, and ultimately the speech emotion reputation multiple classifier device turned into carried out. The speech movement reputation charge reached via the gadget is 86.5%, that's 7% better than the original technique.

Summary: In this newsletter we have found out about the importance of DNN version and its implementation which we will also use in our application.

SCOPE:

- Automatic emotion recognition through speech can help businesses higher recognize their customers as they communicate.
- Call facilities can expand separate

techniques for interacting with one-of-a-kind humans.

- In the case of learning, faculties can music the attitudes in their students so that the schooling gadget can higher put together college students.
- Robotics makes vast use of movement detection due to the fact the robot is disconnected from human desires to understand human movement.
- Motion detection is the important thing to Human Computer Interaction (HCI).

EXISTING METHOD

In the most recent systems available, the model used for motion recognition uses traditional machine learning algorithms such as support vector machines (SVMs), K-neighbors (KNNs), etc. The accuracy of these models is low. However, there are other deep models in the literature, but they tend to be trained on large datasets, which are time-consuming and therefore the models are very complex.

DISADVANTAGES:

- Less accurate.
- High computational complexity.
- The application requires high performance hate.

PROPOSED SYSTEM

We propose a modified speech emotion recognition method that uses neural networks for deep learning. The Mel method uses cepstral frequency coefficients (MFCC), a chromogram, an ascending Mel spectrogram in combination with spectral contrast and tonal functions to extract details from an audio file. These data are used to train a DNN model in a 5-layer deep neural network. The dataset used here is the Ryerson Audiovisual Motion Speech and Song Database (RAVDESS). We selected only part of the speech, consisting of four actors (group-balanced) with 1440 audio files. The speech model classifies tone into 8 different emotions, namely neutral, calm, happy, sad, angry, monster, disgust, and surprised.

ADVANTAGES

- More precisely.
- Low computational complexity.
- Does not require very high performance equipment to operate.

APPLICATIONS

- Scientific organizations.
- Business processes from exciting companies to call centers.
- Organizations working in the field of robotics.
- Human Computer Interaction (HCI)

Organizations.

HARDWARE & SOFTWARE REQUIREMENTS

H/W

Configuration

:

- Processor
 - I3/Intel Processor
- RAM - 4GB (min)
- Hard Disk - 128 GB
- Keyboard-Standard Windows Keyboard
- Mouse - Two or Three Button Mouse
- Monitor - Any

S/W Configuration:

Operating System : Windows 7+

Server side Script : Python 3.6+

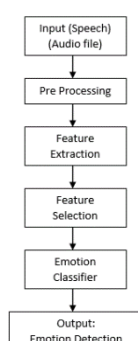
IDE : PyCharm

Libraries Used :Pandas, Numpy, Kera, Tensorflow, Librosa, OpenCV, Flask, Pickle.

Dataset : RAVDESS speech dataset.

ARCHITECTURE

The flow for the project is given below:



MODULES

Upload:

Download a hard and fast of audio statistics (.Wav files) using the studying library.

View:

The downloaded dataset can be regarded.

Preprocessing:

Data preprocessing is a technique that uses uncooked statistics to convert it into pure records. Data cleansing refers to doing away with null values, filling nulls with a meaningful value, getting rid of duplicate values, getting rid of outliers, and doing away with needless attributes. If the table contains any categorical tables, the ones tools convert the categorical variables to numeric values.

Identifying Features:

Honey extracted functions are frequency cepstral coefficients (MFCC), chromogram, honey scale spectrogram with spectrum contrast and tonal function.

Train and Test Split:

We cut up our dataset of 1440 audio files into 2 elements: schooling statistics with 1008 audio documents and take a look at facts with 432 audio files. Here, 70% of the records is taken for training dataset.

Building the model:

- We advise a deep getting to know method for sound and predictive movement sensing.
- Deep getting to know can offer stepped forward accuracy and reduced processing strength.
- We will use Deep Neural Networks (DNN) to create the model.
- Deep Neural Network (DNN) is extensively used in deep gaining knowledge of to teach models in responsibilities that conventional system studying algorithms have issue fixing or can't.
- The model is created the usage of five layers of neural networks.
- We used dropout to reduce the overfitting hassle.
- To consult with noise by means of emotion, we use a softmax version in the bottom layer of our DNN.
- Softmax takes a vector of numbers and converts them into possibilities, which might be then used to generate a generational photograph.
- Softmax converts logits to probabilities by way of taking the exponents from each output, after which normalizing every of these numbers in the sum of the exponents, so that the vector sums to one.

Prediction:

Audio is loaded by the person (along

with human speech) and a version of the speaker's feelings is used in the audio.

User Interface:

To use the version, a web application turned into evolved based on the Flask architecture. It includes 2 elements, motive and practice. The consumer interface has a user registration and account opening device.

Registration:

A new user first wishes to sign up their info, along with name, electronic mail address and password. This website statistics is stored in a MySQL database.

User Login:

You can already be a person whose information is saved inside the MySQL database machine within the internet software using their legitimate credentials. Only after efficiently logging in are you able to get right of entry to the emotion prediction app.

ALGORITHM:**Artificial Neural Network:**

An artificial neural community is a machine of hardware or software based on neurons within the human mind and anxious system. Artificial neural networks are a form of deep mastering technology that belongs to the large subject of synthetic

intelligence.

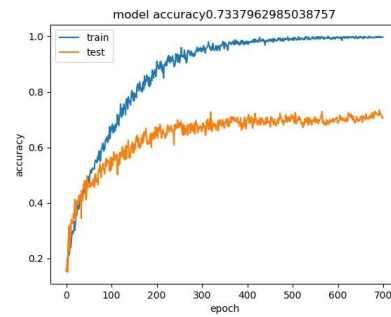
Deep learning is a branch of device learning that makes use of different kinds of neural networks. These algorithms are inspired via how the brain works, and consequently many professionals consider that they're the satisfactory flow towards actual AI (artificial intelligence).

Results

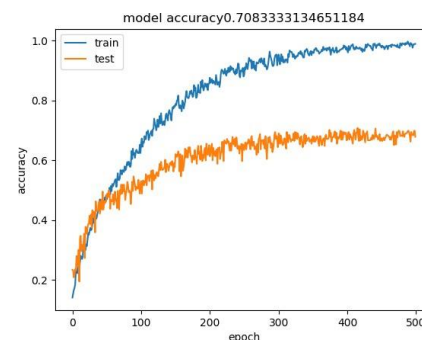
- The proposed technique provides an method to figuring out feelings from human speech.
- A neural network technique was used.
- We have successfully advanced a deep mastering version with a deep neural community structure to expect speaker actions.
- We carried out our mission the use of an internet utility structure field. The user interface also includes a user registration machine.
- Using the trained version, we have been capable of get a test accuracy of 73.4%.

The educate and trying out of our version is shown to be accurate over 700 epochs;

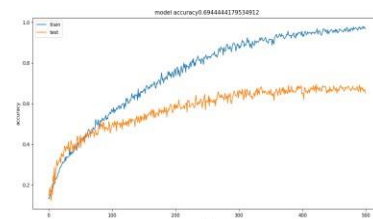
The version receives the best test accuracy of 73.4%



We have previously tried different fashions with one of a kind accuracy.



And this is any other version that we developed with DNN.



CONCLUSION:

The proposed scheme is an method to popularity by means of human speech belief. This approach uses neural networks. We have efficiently advanced a deep mastering model the use of a deep neural community structure for speaker emotions in audio. We made our design very clear in the internet software the usage of field architecture. The user interface also includes a consumer registration

gadget. We have been in a position to test the accuracy of 73.4% through the use of the model layout.

Please observe that the prediction is subjective emotion and human-judged emotion, because the same audio may additionally range from individual to man or woman. This is likewise the reason why a set of rules educated on human feelings can from time to time give misguided consequences. The model become skilled at the RAVDESS dataset, so the speaker's accessory may also lead to misguided results whilst the model was educated simplest at the North American accessory database.

FUTURE SCOPE:

Ability to report live voice and are expecting emotions in real time during a speaker's speech. It also requires information of signal processing, as the voice ought to be cleaned of all undesirable noise in it earlier than prediction.

REFERENCES:

[1] Szegedy, Christian & Toshev, Alexander & Erhan, Dumitru. (2013). Deep Neural Networks for Object Detection. 1-9.
[2] Benk, Sal & Elmir, Youssef & Dennai, Abdeslem. (2019). A Study on Automatic Speech Recognition. 10. 77-

85. 10.6025/jitr/2019/10/3/77-85.

[3] Ashish B. Ingale & D. S. Chaudhari (2012). Speech Emotion Recognition. International Journal of Soft Computing and Engineering (IJSCE) ISSN: 2231-2307, Volume-2 Issue-1, March 2012
[4] Chenchen Huang, Wei Gong, Wenlong Fu, Dongyu Feng, "A Research of Speech Emotion Recognition Based on Deep Belief Network and SVM", *Mathematical Problems in Engineering*, vol. 2014, Article ID 749604, 7 pages, 2014. <https://doi.org/10.1155/2014/749604>
[5] Livingstone SR, Russo FA (2018) The Ryerson Audio-Visual Database of Emotional Speech and Song (RAVDESS): A dynamic, multimodal set of facial and vocal expressions in North American English. PLoS ONE 13(5): e0196391.
[6] M. E. Ayadi, M. S. Kamel, F. Karray, "Survey on Speech Emotion Recognition: Features, Classification Schemes, and Databases", Pattern Recognition 44, PP.572-587, 2011.
[7] I. Chiriacescu, "Automatic Emotion Analysis Based On Speech", M.Sc. THESIS Delft University of Technology, 2009.
[8] T. Vogt, E. Andre and J. Wagner, "Automatic Recognition of Emotions from Speech: A review of the literature

and recommendations for practical realization", LNCS 4868, PP.75-91, 2008.

[9] S. Emerich, E. Lupu, A. Apatean, "Emotions Recognitions by Speech and Facial Expressions Analysis", 17th European Signal Processing

Conference, 2009.

[10] P.Shen, Z. Changjun, X. Chen, "Automatic Speech Emotion Recognition Using Support Vector Machine", International Conference On Electronic And Mechanical Engineering And Information Technology, 2011.