

CHAT BOT SONG RECOMMENDED SYSTEM USING MACHINE LEARNING

A Project Report Phase-II

Submitted in partial fulfilment of requirements for the award of
Bachelor of Engineering

in

Electronics and Communication Engineering

by

SITARAM PONNAGANTI (39130351)

CHETAN PRASAD CHALASANI (39130084)



DEPARTMENT OF ELECTRONICS AND COMMUNICATION ENGINEERING

SCHOOL OF ELECTRICAL AND ELECTRONICS

SATHYABAMA

**INSTITUTE OF SCIENCE AND TECHNOLOGY
(DEEMED TO BE UNIVERSITY)**

Accredited with Grade "A" by NAAC

JEPPIAAR NAGAR, RAJIV GANDHI SALAI, CHENNAI – 600119

APRIL – 2022



SATHYABAMA

INSTITUTE OF SCIENCE AND TECHNOLOGY

(DEEMED TO BE UNIVERSITY)

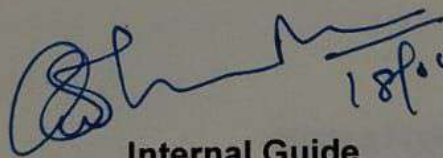
Accredited "A" Grade by NAAC | 12B Status by UGC | Approved by AICTE

www.sathyabama.ac.in

DEPARTMENT OF ELECTRONICS AND COMMUNICATION ENGINEERING

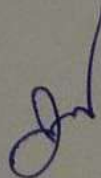
BONAFIDE CERTIFICATE

This is to certify that this Project Report is the bonafide work of **Sitaram Ponnaganti (39130351)** and **Chetan Prasad Chalasani (39130084)** who carried out the project entitled "**CHAT BOT SONG RECOMMENDED SYSTEM USING MACHINE LEARNING**" under our supervision from Aug 2022 to Oct 2022.


18/04/2023

Internal Guide

Dr.M.S. GODWIN PREMI, M.E., Ph.D.

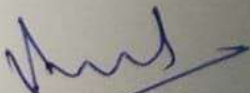


Head of the Department

Dr.T. RAVI M.E., Ph.D.

Submitted for Viva voce Examination held on

19/04/23



19/04/23

DECLARATION

We, SITARAM PONNAGANTI (39130351) and CHETAN PRASAD CHALASANI (39130084) hereby declare that the Project Report entitled "CHAT BOT SONG RECOMMENDED SYSTEM USING MACHINE LEARNING" done by us under the guidance of Dr. M.S. GODWIN PREMI, M.E, Ph.D., Professor, Department of Electronics and Communication Engineering is submitted in partial fulfillment of the requirements for the award of Bachelor of Engineering degree in Electronics and Communication Engineering.

SIGNATURE OF THE CANDIDATES

DATE: 19/4/23

PLACE: Chennai

1. P.S. to Dr.

2. Ch. Chetan Prasad

ACKNOWLEDGEMENT

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

We convey our thanks to **Dr. NANDHITHA, M.E., Ph.D.**, Dean, School of Electrical and Electronics and **Dr.T. RAVI M.E., Ph.D.** Head of the Department, Department of Electronics and Communication Engineering for providing me necessary support and details at the right time during the progressive reviews.

We would like to express our sincere and deep sense of gratitude to our Project Guide **Dr.M.S. GODWIN PREMI, M.E, Ph.D.**, Professor, Department of Electronics and Communication Engineering for her valuable guidance, suggestions and constant encouragement paved way for the successful completion of our project work.

We wish to express our thanks to all Teaching and Non-teaching staff members of the Department of Electronics and Communication Engineering who were helpful in many ways for the completion of the project.

ABSTRACT

Technology has a great impact on every part of lives, which also includes our day-to-day habits. present scenario, fields like artificial intelligence and machine learning are on great boom. With the help of advancement in these fields, large number of people are interacting through the system via chat bots and voice assistants. Considering above factors, this project is aimed to implement the Machine Learning based Chat Bot Song Recommender System that includes chat bot to assist user and recommend songs using the Natural Language Processing. In this paper we will discuss methodology, algorithms and the flow of the system.

TABLE OF CONTENTS

CHAPTER No.	TITLE	PAGE No.
	ABSTRACT	V
	LIST OF FIGURES	VII
1	INTRODUCTION	
	1.1 Introduction to Chat Bot Systems	1
	1.2 Chat Bot	2
	1.3 Machine Learning Overview	3
	1.4 Deep Learning	4
	1.5 Machine Learning System Classification	6
	1.6 Machine Learning Applications	7
	1.7 Learning Associations	9
	1.8 Classification	10
2	LITERATURE REVIEW	
	2.1 Learning Audio Embeddings	13
	2.2 Optimizing Recommendation Algorithms	13
	2.3 Emotion Based Music Recommendation System	14
	2.4 Music Genre Classification Using Acoustic Features	14
	2.5 Designing Music Recommendation System	14
	2.6 Unified Transformer Multitask Learning	15
	2.7 Building an Arabic Flight Booking	15
	2.8 Enhancing Harvested Energy Utilization for Energy	16
	2.9 Entertainment Chat Bot for the Digital Inclusion	17
	2.10 Problem Statement	18
	2.11 Aim	18

3	EXISTING AND PROPOSED WORK	
	3.1 Existing System	19
	3.2 Methodology	19
	3.3 Architecture Of Existing System	20
4	EXPERIMENTAL OR MATERIALS AND METHODS	
	4.1 Installation Process of Visual Studio Code	23
	4.2 Deep Neural Network Method	25
	4.3 Development of DNN Based Prediction	26
	4.4 Method Followed	27
	4.5 Importing Libraries	29
	4.6 Model Making and Training	32
	4.7 Flask	40
5	RESULTS AND DISCUSSION, PERFORMANCE	
	5.1 Result	42
	5.2 Discussions	43
	REFERENCE	56

S .No	LIST OF FIGURES	Page No
1	3.2.1 Architecture of Existing System	20
2	3.5.1 Architecture of Proposed System	22
3	4.1.1 Downloading the Visual Studio Code	23
4	4.1.2 Installing the Visual Studio Code	24
5	4.1.3 Installing the Python in Visual Studio Code	24

6	4.1.4 Installation the Jupyter in Visual Studio Code	24
7	4.3.1 DNN Model	27
8	4.6.1 Training Model	38
9	5.1.1 The Emotion Sadness detected	42
10	5.1.2 The Emotion Shame detected	43
11	5.2.1 A Typical workflow of Model Development	49
12	5.3.1 Operation of LSTM	50
13	5.3.2 Detection of Shame	52
14	5.3.3 Detection of Fear	52
15	5.3.4 Detection of Joy	53
16	5.3.5 Detection of Anger	53
17	5.3.6 Detection of Sadness	54

S.NO	LIST OF TABLES	PAGE NO
1	Confusion Matrix	44

S.NO	LIST OF GRAPHS	PAGE NO
1	Confusion Matrix	45
2	Confusion Matrix Analysis	45
3	Model Accuracy	50

CHAPTER-1

INTRODUCTION

CHAT BOT SONG RECOMMENDED SYSTEM USING MACHINE LEARNING

1.1 Introduction to Chat Bot Systems

Now-a-days, we all are living in the time where we know that nothing is certain. Same goes with our mind, at regular instances of time our mood, our choices and our priorities changes. Considering the constant changing behavior of human being we have developed our system. We have made our system considering that the humans experience frequently changes in their mood and somehow, at particular moment of time, frequently changing of mood would also result in change in mood of music of their choice. Hence, with the help of our system you can listen to music according to your mood. In addition, we have also provided the facility to user to chat with the chat bot after all texting makes conversation between chat bot and user more interactive and it will efficiently help in analysing the current mood of the user and based on that chat bot will recommend songs. Besides these, our system comprises of four modules. Initially for a new user we have provided a sign-up page. After completing sign-up, user can login to our system. In addition to our system, we've also provided the password recovery facility to user, in case user forgot the password. After chatting with the chat bot, current mood of the user is analysed and list of songs is suggested to user, using the concept of NLP (Natural Language Processing). Based on the list of songs user can choose the song to be played based on his or her choice. We have used Python as our prime language because it supports an extensive set of open-source libraries which can be used by our system. Every day, each and every person undergoes lot of troubles and the reliever of all the stress that are encountered is Music. If it is so, the vital part of hearing the song has to be in a facilitated way, that is player able to play the song in accordance with the person's mood. The paper proposes such a player and hence named Emotion based music player. Recognizing the human emotions is considered to be a global consistency but depict variability among the humans on the basis of their abilities. The different approaches being the classification of emotions that are existing are depicted as knowledge-based, statistical and hybrid techniques. However there relies on several difficulties while

retrieving the music information such as querying by singing, genre classification, etc. The most possible implementation is by producing music suggestion that is based on the content. The multiple efforts to detect and describe the feeling, to classify based on features are all wanted to obtain outstanding music recommendation system. For describing a music taxonomy, a feeling descriptor is found to be useful. In this project, we would be building an extensive Chatbot service, to which you can talk to. And talking to a chatbot wouldn't be business-driven. It would just be casual conversations. Further, on top of it, the chatbot would also be recommending songs to the user based on the tone of the user. This song recommendation feature employs the use of Last.fm API, very much similar to the popular Spotify API. Also, for tone/emotion analysis of the conversation we will be using the IBM Tone Analyzer API. Collaborating with these types of APIs is very much critical as in today's world the popular chatbots do much more than simply having a data-driven conversation; to supplement additional user-oriented features. Also, the reason to choose python to build the chatbot is because python boasts a wide array of opensource libraries for chatbots, including scikit-learn and TensorFlow. It is great for small data sets and more simple analyses; also, Python's libraries are much more practical.

1.2 Chat Bot

At the most basic level, a chatbot is a computer program that simulates and processes human conversation (either written or spoken), allowing humans to interact with digital devices as if they were communicating with a real person. Chatbots can be as simple as rudimentary programs that answer a simple query with a single-line response, or as sophisticated as digital assistants that learn and evolve to deliver increasing levels of personalization as they gather and process information.

1.2.1 Methodology

The proposed system work develops a personalized system, where the user's current emotion is analyzed with the help of the chat bot. The chat bot identifies the user's sentiment by chat conversation with the user. Based on the input provided by the user, current emotion or mood is analyzed by the chat bot and it will suggest song to the user. The objective of our application is to identify the mood expressed by the user and once the mood is identified, songs are played by the application

1.3 Machine Learning – Overview

Machine learning is a very hot topic for many key reasons, and because it provides the ability to automatically obtain deep insights, recognize unknown patterns, and create high performing predictive models from data, all without requiring explicit programming instructions.

This high-level understanding is critical if ever involved in a decision-making process surrounding the usage of machine learning, how it can help achieve business and project goals, which machine learning techniques to use, potential pitfalls, and how to interpret the results.

The most common machine learning tasks that one may come across while trying to solve a machine learning problem. Under each task are also listed a set of machine learning methods that could be used to resolve these tasks. Please feel free to comment/suggest if I missed mentioning one or more important points.

Following are the key machine learning tasks briefed later in this article:

Feature Selection, Regression, Classification, Clustering, Multivariate querying, Density estimation, Dimension reduction, Testing and matching

Following are top 8 most common machine learning tasks that one could come across most frequently while solving an advanced analytics problem:

Feature Selection: Feature selection is one of the critical tasks which would be used when building machine learning models. Feature selection is important because selecting right features would not only help build models of higher accuracy but also help achieve objectives related to building simpler models, reduce overfitting etc. The following are some of the techniques which could be used for feature selection:

Filter methods which help in selecting features based on the outcomes of statistical tests. The following are some of the statistical tests which are used:

Pearson's correlation, Linear discriminant analysis (LDA), Analysis of Variance (ANOVA), Chi-square tests

Wrapper methods which help in feature selection by using a subset of features and determining the model accuracy. The following are some of the algorithms used:

Forward selection, backward elimination

1.3.1 Applications of Chat Bot

It is used in Retail and e-commerce, Travel and hospitality, Banking, finance, and fintech, Healthcare, Media and entertainment, Education.

1.3.2 Classification

Classification tasks is simply related with predicting a category of a data (discrete variables). One of the most common examples is predicting whether or not an email is spam or ham. Some of the common use cases could be found in the area of healthcare such as whether a person is suffering from a particular disease or not. It also has its application in financial use cases such as determining whether a transaction is fraud or not. The ML methods such as following could be applied to solve classification tasks:

Kernel discriminant analysis (*Higher accuracy*), K-Nearest Neighbours (*Higher accuracy*), Artificial neural networks (ANN) (*Higher accuracy*), Support vector machine (SVM) (*Higher accuracy*), Random forests (*Higher accuracy*), Decision trees, Boosted trees, Logistic regression, naive Bayes

1.4 Deep Learning

1.4.1 Clustering

Clustering tasks are all about finding natural groupings of data and a label associated with each of these groupings (clusters). Some of the common examples includes customer segmentation, product features identification for product roadmap. Some of the followings are common ML methods:

Mean-shift (*Higher accuracy*), Hierarchical clustering, K-means, Topic models

1.4.2 Multivariate Querying

Multivariate querying is about querying or finding similar objects. Some of the following ML methods could be used for such problems:

Nearest neighbours, Range search, Farthest neighbours

1.4.3 Density Estimation

Density estimation problems are related with finding likelihood or frequency of objects. In probability and statistics, density estimation is the construction of an estimate, based on observed data, of an unobservable underlying probability density function. Some of the following ML methods could be used for solving density estimation tasks:

Kernel density estimation (*Higher accuracy*), Mixture of Gaussians, Density estimation tree

1.4.5 Dimension Reduction

Dimension reduction is the process of reducing the number of random variables under consideration and can be divided into feature selection and feature extraction. Following is some of ML methods that could be used for dimension reduction:

Manifold learning/KPCA (*Higher accuracy*), Gaussian graphical models, Non-negative matrix factorization, Compressed sensing.

1.4.6 Testing and Matching

Testing and matching tasks relates to comparing data sets. Following is some of the methods that could be used for such kind of problems:

Minimum spanning tree, Bipartite cross-matching, N-point correlation

1.5 Machine Learning System Classification

Although supervised and unsupervised learning are two of the most widely accepted machine learning methods by businesses today, there are various other machine learning techniques. Following is an overview of some of the most accepted ML methods

1.5.1 Supervised Learning

These algorithms are trained using labelled examples, in different scenarios, as an input where the desired outcome is already known. An equipment, for instance, could have data points such as "F" and "R" where "F" represents "failed" and "R" represents "runs".

A learning algorithm will receive a set of input instructions along with the corresponding accurate outcomes. The learning algorithm will then compare the actual outcome with the accurate outcome and flag an error, if there is any discrepancy. Using different methods, such as regression, classification, gradient boosting, and prediction, supervised learning uses different patterns to proactively predict the values of a label on extra unlabelled data. This method is commonly used in areas where historical data is used to predict events that are likely to occur in the future.

1.5.2 Unsupervised Learning

This method of ML finds its application in areas where data has no historical labels. Here, the system will not be provided with the "right answer" and the algorithm should identify what is being shown. The main aim here is to analyse the data and identify a pattern and structure within the available data set. Transactional data serves as a good source of data set for unsupervised learning.

For instance, this type of learning identifies customer segments with similar attributes and then lets the business to treat them similarly in marketing campaigns. Similarly, it can also identify attributes that differentiate customer segments from one another.

Either way, it is about identifying a similar structure in the available data set. Besides, these algorithms can also identify outliers in the available data sets.

Some of the widely used techniques of unsupervised learning are
k-means clustering, self-organizing maps, value decomposition

1.5.3 Semi-Supervised Learning

This kind of learning is used and applied to the same kind of scenarios where supervised learning is applicable. However, one must note that this technique uses both unlabeled and labelled data for training. Ideally, a small set of labelled data, along with a large volume of unlabeled data is used, as it takes less time, money and efforts to acquire unlabeled data. This type of machine learning is often used with methods, such as regression, classification and prediction.

1.5.4 Reinforcement Learning

This is mainly used in navigation, robotics and gaming. Actions that yield the best rewards are identified by algorithms that use trial and error methods. There are three major components in reinforcement learning, namely, the agent, the actions and the environment.

1.6 Machine Learning Applications

As we move forward into the digital age, one of the modern innovations we've seen is the creation of Machine Learning. This incredible form of artificial intelligence is already being used in various industries and professions. For Example, Image and Speech Recognition, Medical Diagnosis, Prediction, Classification, Learning Associations, Statistical Arbitrage, Extraction, Regression. Today we're looking at all these Machine Learning Applications in today's modern world.

It is one of the most common machine learning applications. There are many situations where you can classify the object as a digital image. For digital images, the measurements describe the outputs of each pixel in the image.

In the case of a black and white image, the intensity of each pixel serves as one measurement. So, if a black and white image has $N \times N$ pixels, the total number of pixels and hence measurement is N^2 .

In the colored image, each pixel considered as providing 3 measurements of the intensities of 3 main color components i.e., RGB. So, $N \times N$ colored image there are $3N^2$ measurements.

For face detection – The categories might be face versus no face present. There might be a separate category for each person in a database of several individuals.

For character recognition – We can segment a piece of writing into smaller images, each containing a single character. The categories might consist of the 26 letters of the English alphabet, the 10 digits, and some special characters.

Speech recognition (SR) is the translation of spoken words into text. It is also known as “automatic speech recognition” (ASR), “computer speech recognition”, or “speech to text” (STT).

In speech recognition, a software application recognizes spoken words. The measurements in this Machine Learning application might be a set of numbers that represent the speech signal. We can segment the signal into portions that contain distinct words or phonemes. In each segment, we can represent the speech signal by the intensities or energy in different time-frequency bands.

Although the details of signal representation are outside the scope of this program, we can represent the signal by a set of real values.

Speech recognition, Machine Learning applications include voice user interfaces. Voice user interfaces are such as voice dialling, call routing, domestic appliance control. It can also use as simple data entry, preparation of structured documents, speech-to-text processing, and more.

ML provides methods, techniques, and tools that can help in solving diagnostic and prognostic problems in a variety of medical domains. It is being used for the analysis

of the importance of clinical parameters and of their combinations for prognosis, e.g., prediction of disease progression, for the extraction of medical knowledge for outcomes research, for therapy planning and support, and for overall patient management. ML is also being used for data analysis, such as detection of regularities in the data by appropriately dealing with imperfect data, interpretation of continuous data used in the Intensive Care Unit, and for intelligent alarming resulting in effective and efficient monitoring.

It is argued that the successful implementation of ML methods can help the integration of computer-based systems in the healthcare environment providing opportunities to facilitate and enhance the work of medical experts and ultimately to improve the efficiency and quality of medical care.

In medical diagnosis, the main interest is in establishing the existence of a disease followed by its accurate identification. There is a separate category for each disease under consideration and one category for cases where no disease is present. Here, machine learning improves the accuracy of medical diagnosis by analysing data of patients.

The measurements in this Machine Learning applications are typically the results of certain medical tests (example blood pressure, temperature and various blood tests) or medical diagnostics, presence/absence/intensity of various symptoms and basic physical information about the patient (age, sex, weight etc.). On the basis of the results of these measurements, the doctors narrow down on the disease inflicting the patient.

1.7 Learning Associations

Learning association is the process of developing insights into various associations between products. A good example is how seemingly unrelated products may reveal an association to one another. When analysed in relation to buying behaviours of customers.

One application of machine learning- Often studying the association between the products people buy, which is also known as basket analysis. If a buyer buys 'X', would he or she force to buy 'Y' because of a relationship that can identify between them? This leads to the relationship that exists between fish and chips etc.

when new products launch in the market a Knowing these relationships it develops a

new relationship. Knowing these relationships could help in suggesting the associated product to the customer. For a higher likelihood of the customer buying it, it can also help in bundling products for a better package.

This learning of associations between products by a machine is learning associations. Once we found an association by examining a large amount of sales data, Big Data analysts. It can develop a rule to derive a probability test in learning a conditional probability.

1.8 Classification

Classification is a process of placing each individual from the population under study in many classes. This is identified as independent variables.

Classification helps analysts to use measurements of an object to identify the category to which that object belongs. To establish an efficient rule, analysts use data. Data consists of many examples of objects with their correct classification.

For example, before a bank decides to disburse a loan, it assesses customers on their ability to repay the loan. By considering factors such as customer's earning, age, savings and financial history we can do it. This information is taken from the past data of the loan. Hence, Seeker uses to create a relationship between customer attributes and related risks.

1.8.1 Prediction

Consider the example of a bank computing the probability of any of loan applicants faulting the loan repayment. To compute the probability of the fault, the system will first need to classify the available data in certain groups. It is described by a set of rules prescribed by the analysts.

Once we do the classification, as per need we can compute the probability. These probability computations can compute across all sectors for varied purposes

The current prediction is one of the hottest machine learning algorithms. Let's take an example of retail, earlier we were able to get insights like sales report last month / year / 5-years / Diwali / Christmas.

1.8.2 Extraction

Information Extraction (IE) is another application of machine learning. It is the process of extracting structured information from unstructured data. For example, web pages, articles, blogs, business reports, and e-mails. The relational database maintains the output produced by the information extraction.

The process of extraction takes input as a set of documents and produces a structured data. This output is in a summarized form such as an excel sheet and table in a relational database.

Nowadays extraction is becoming a key in the big data industry.

As we know that the huge volume of data is getting generated out which most of the data is unstructured. The first key challenge is handling unstructured data. Now conversion of unstructured data to structured form based on some pattern so that the same can stored in RDBMS.

Apart from this in current days' data collection mechanism is also getting change. Earlier we collected data in batches like End-of-Day (EOD), but now business wants the data as soon as it is getting generated, i.e., in real time.

1.8.3 Regression

We Can apply Machine learning to regression as

Assume that $x = x_1, x_2, x_3, \dots, x_n$ are the input variables and y is the outcome variable. In this case, we can use machine learning technology to produce the output (y) on the basis of the input variables (x). You can use a model to express the relationship between various parameters as below:

$Y = g(x)$ where g is a function that depends on specific characteristics of the model. In regression, we can use the principle of machine learning to optimize the parameters. To cut the approximation error and calculate the closest possible outcome.

We can also use Machine learning for function optimization. We can choose to alter the inputs to get a better model. This gives a new and improved model to work with. This is known as response surface design.

CHAPTER 2

LITERATURE REVIEW

2.1 Learning Audio Embedding's with User Listening Data

Personalized recommendation on new track releases has always been a challenging problem in the music industry. To combat this problem, we first explore user listening history and demographics to construct a user embedding representing the user's music preference. With the user embedding and audio data from user's liked and disliked tracks, an audio embedding can be obtained for each track using metric learning with Siamese networks. For a new track, we can decide the best group of users to recommend by computing the similarity between the track's audio embedding and different user embeddings, respectively. The proposed system yields state-of-the-art performance on content-based music recommendation tested with millions of users and tracks. Also, we extract audio embeddings as features for music genre classification tasks. The results show the generalization ability of our audio embeddings.

2.2 Optimizing Recommendation Algorithms using Self-Similarity Matrices

Most music recommendation algorithms such as proprietary ones designed for curating music in music streaming companies rely mainly on song ratings data to be able to recommend songs to users. This does not always provide a great music experience for many users who pay for these services. As an example, there are over 35 million songs on Spotify alone and according to recent statistical report released by Spotify there are about 4 million songs on Spotify that have never been played not even once. Clearly this is not good news to content creators who trust and pay Spotify as a platform to distribute their music. The root of the problem lies on the music recommendation algorithm adopted and used by many music streaming platforms like Spotify. The algorithm fails to recommend great music to the users which they can enjoy, and this results in the platform experiencing "dark music" or music that has

never received any play in the platform. We propose a framework which provides a unique strategy that can be adopted by music recommendation

2.3 Emotion-Based Music Recommendation System

Music is the form of art known to have a greater connection with a person's emotion. It has got a unique ability to lift up one's mood. If a user receives a recommendation based on his preference, it will also improve his listening experience. Music recommendations have existed for a long time. Still, in most scenarios, the recommendation is decided after learning the user preferences over time, like looking at their past song preferences, the amount of time they listen to the music, etc. This paper suggests a neural network-based approach to song recommendation where their facial expressions detect a person's mood.

2.4 Music Genre Classification Using Acoustic Features And Auto Encoders

Music recommendation and classification systems are an area of interest of digital signal processing and digital music processing. In this study by using digital signal processing techniques and autoencoders, music features are extracted and then with these features music classification and clustering has been done, and with the results music recommendation has been made. Obtained results are compared with each other. In the study, GTZAN dataset has been used. Purpose of this study is to compare the result feature extraction with auto encoders and digital signal processing techniques.

2.5 Designing Music Recommendation System Based On Music Genre By Using BI-LSTM

Many people consider music as an integral part of their lives and use it in their daily activities. Therefore, suggesting a music track based on users' previous searches and tastes can be a pleasant experience for them. The music recommendation system also helps users filter and search for songs based on their interests, age, region, and language. To do so, processing to get to the best recommendations is done based on

artist style and searched words. The approaches and mechanisms applied for data analysis in recommender systems change based on their type. The goal of using recommender systems in these contexts is to encourage users to have higher application utilization. One of the latest recommender systems, which is mentionable, is the content-based approach. In these systems, data content is used instead of user scoring patterns for recommendation generation.

2.6 Unified Transformer Multi-Task Learning For Intent Classification with Entity Recognition

Intent classification (IC) and Named Entity Recognition (NER) are arguably the two main components needed to build a Natural Language Understanding (NLU) engine, which is a main component of conversational agents. The IC and NER components are closely intertwined and the entities are often connected to the underlying intent. Current research has primarily focused to model IC and NER as two separate units, which results in error propagation, and thus, sub-optimal performance. In this paper, we propose a simple yet effective novel framework for NLU where the parameters of the IC and the NER models are jointly trained in a consolidated parameter space. Text semantic representations are obtained from popular pre-trained contextual language models, which are fine-tuned for our task, and these parameters are propagated to other deep neural layers in our framework leading to a faithful unified modelling of the IC and NER parameters. The overall framework results in a faithful parameter sharing when the training is underway, leading to a more coherent learning. Experiments on two public datasets, ATIS and SNIPS, show that our model outperforms other methods by a noticeable margin. On the SNIPS dataset, we obtain a 1.42% improvement in NER in terms of the F1 score, and 1% improvement in intent accuracy score. On ATIS, we achieve 1.54% improvement in intent accuracy score. We also present qualitative results to showcase the effectiveness of our model.

2.7 Building an Arabic Flight Booking Dialogue System Using a Hybrid Rule-Based and Data Driven Approach

Approaches for developing Dialogue Systems (DSs) are typically categorized into rule-based and data-driven. Data-driven DSs require a massive quantity of training data, while rule-based DSs rely on a predefined set of rules and keywords that are to be detected in the user's utterances. The data-driven approaches show more promising results, but owing to a lack of available training data for Arabic task-oriented

DSs, development of an Arabic task-oriented DS has typically been conducted using the rule-based approach. In this article, we propose a hybrid rule-based and data-driven approach in a text based flight booking DS capable of handling customer's utterances. The proposed DS was built through utilizing the Wit.ai natural language interface. The conversation flow was configured using the Wizard of Oz technique, and the DS intents and entities were developed through the use of crowdsourcing training examples. The evaluation results show that the system developed was able to understand user utterances and to self-feed efficiently. Dating With Scambots: Understanding The Ecosystem Of Fraudulent Dating Applications

In this work, we are focusing on a new and yet uncovered way for malicious apps to gain profit. They claim to be dating apps. However, their sole purpose is to lure users into purchasing premium/VIP services to start conversations with other (likely fake female) accounts in the app. We call these apps as fraudulent dating apps. This paper performs a systematic study to understand the whole ecosystem of fraudulent dating apps. Specifically, we have proposed a three-phase method to detect them and subsequently comprehend their characteristics via analyzing the existing account profiles. Our observation reveals that most of the accounts are not managed by real persons, but by Chabot's based on predefined conversation templates. We also analyses the business model of these apps and reveal that multiple parties are actually involved in the ecosystem, including producers who develop apps, publishers who publish apps to gain profit, and the distribution network that is responsible for distributing apps to end users. Finally, we analyses the impact of them to users (i.e., victims) and estimate the overall revenue. Our work is the first systematic study on fraudulent dating apps, and the results demonstrate the urge for a solution to protect users.

2.8 Enhancing Harvested Energy Utilization for Energy Harvesting Wireless Sensor Networks by an Improved Uneven Clustering Protocol

The energy limitation in traditional wireless sensor networks is effectively ameliorated by equipping energy harvesting modules and rechargeable batteries on nodes in energy harvesting wireless sensor networks (EH-WSNs). However, enhancing the harvested energy utilization is still a challenge. In this paper, we proposed an improved uneven clustering protocol to enhance the harvested energy utilization of EH-WSNs. The protocol contains cluster establishment and data collection stages. To reduce the energy consumed for cluster head (CH) selection and reserve more energy for data

transmission, a novel CH selection scheme is proposed to select nodes with better performance as CHs. To further enhance the harvested energy utilization, a dynamic transmission power adjustment scheme is designed for both CHs and cluster members in the data collection stage under the limited capacity of rechargeable batteries. A series of experiments are conducted to verify the effectiveness of the proposed algorithm, and the results demonstrate that the proposed algorithm utilizes the harvested energy more efficiently and performs better than the corresponding competitors.

2.9 Entertainment Chatbot for the Digital Inclusion of Elderly People without Abstraction Capabilities

Current language processing technologies allow the creation of conversational chatbot platforms. Even though artificial intelligence is still too immature to support satisfactory user experience in many mass market domains, conversational interfaces have found their way into ad hoc applications such as call centres and online shopping assistants. However, they have not been applied so far to social inclusion of elderly people, who are particularly vulnerable to the digital divide. Many of them relieve their loneliness with traditional media such as TV and radio, which are known to create a feeling of companionship. In this paper we present the EBER chatbot, designed to reduce the digital gap for the elderly. EBER reads news in the background and adapts its responses to the user's mood. Its novelty lies in the concept of "intelligent radio", according to which, instead of simplifying a digital information system to make it accessible to the elderly, a traditional channel they find familiar -background news- is augmented with interactions via voice dialogues. We make it possible by combining Artificial Intelligence Modelling Language, automatic Natural Language Generation and Sentiment Analysis. The system allows accessing digital content of interest by combining words extracted from user answers to chatbot questions with keywords extracted from the news items. This approach permits defining metrics of the abstraction capabilities of the users depending on a spatial representation of the word space. To prove the suitability of the proposed solution we present results of real experiments conducted with elderly people that provided valuable insights. Our approach was considered satisfactory during the tests and improved the information search capabilities of the participants.

2.10 Problem Statement

One biggest issue is the scalability of algorithms having real-world datasets under the recommendation system, a huge changing data is generated by user-item interactions in the form of ratings and reviews and consequently, scalability is a big concern for these datasets. Significant investments required, too many choices, the complex onboarding processes, Lack of data analytics capability, the 'cold start' problem, Inability to capture changes in user behavior, Privacy concern.

2.11 Aim

Music is an integral part of our life. Songs have power to change our mood .with that we introduced our project to recommend a song based on our mood it boost our energy. Our main aim is to provide suitable song based on users' emotion. At present Chat Bot are only used for costumer care but we used that chat bot to guess the user emotion via chat. with that emotion our API plays a song suitable to our emotion

CHAPTER 3

EXISTING AND PROPOSED WORK

3.1 Existing System

Chat-Bots Can Only Handle Basic Questions: Chatbots Are Still a Basic Artificial Intelligence Technology and So They Can Only Answer the Basic Questions of Customers and Provide General Information That Is Already Available to Them. They Cannot Solve Complicated Queries or Answer Out of Script Questions and Companies Need to Have Human Customer Service Employees That Can Manage These for Them. However, This Is Changing with Time and Currently, More And More Advanced Chat-Bots Are Entering the Market.

Emotion Based system which can recommend the song based on our face expressions and Song recommended system which are used in song apps like WYNK and SPOTIFY which recommend songs based on our previous song list or searches

3.2 Methodology

The purpose of chat bots is to support and scale business teams in their relations with customers. It could live in any major chat applications like Facebook Messenger, Slack, Telegram, Text etc. Chatbot applications streamline interactions between people and services, enhancing customer experience. At the same time, they offer companies new opportunities to improve the customers engagement process and operational efficiency by reducing the typical cost of customer service. This project is focused on building a custom chatbot that will be your fundamental step of the learning curve of building your own professional chatbots.

But you must be tired of the weird chat bots out there in the world which are made for mainly business purposes? In this project, we would be building an extensive Chatbot service, to which you can talk to. And talking to a chatbot wouldn't be business-driven. It would just be casual conversations. Further, on top of it, the chatbot would also be recommending songs to the user based on the tone of the user. This song recommendation feature employs the use of Last.fm API, very much like the popular Spotify API. Also, for tone/emotion analysis of the conversation we will be using the IBM Tone Analyzer API.

Collaborating with these types of APIs is very much critical as in today's world the popular chatbots do much more than having a data-driven conversation; to supplement additional user-oriented features. Also, the reason to choose python to build the chatbot is because python boasts a wide array of open-source libraries for chatbots, including scikit-learn and TensorFlow. It is great for small data sets and more simple analyses; also Python's libraries are much more practical.

There are so many recommended systems such as emotion based recommended system.

It scans the users face and guess the emotion based on users face expression and recommend a song based on that emotion.

3.3 Architecture of Existing System

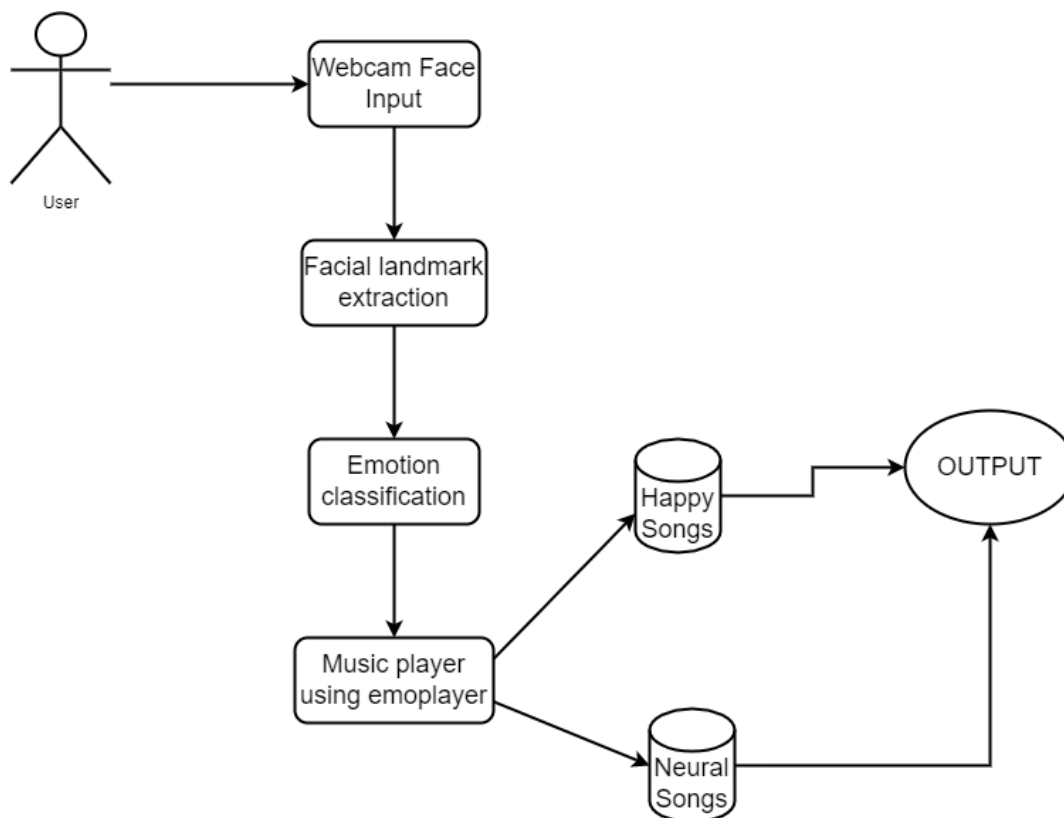


Fig 3.2.1

At first it scans our face and detects our emotion based on our face expressions using CNN and KNN algorithm and recommend a song based on our mood. If our face is happy then it recommend a pop song.

3.4 Proposed System

Chat-bots have 24/7 Availability: Chat-bots are available to solve customer problems 24/7 whether it is day or night! They don't need to sleep after all! This is much more difficult to achieve using human customer service as it would require rotating teams that would be more complicated to manage as well. This means that chat-bots can answer customer queries whenever customers have queries which help in increasing customer loyalty. If companies don't address customer problems as soon as possible, then the customers may switch to another company that provides better customer service.

3.4.1 Recursive Feature Elimination

Regularization techniques which penalize one or more features appropriately to come up with most important features. The following are some of the algorithms used:

LASSO (L1) regularization, Ridge (L2) regularization Regression tasks mainly deal with estimation of numerical values (continuous variables). Some of the examples include estimation of housing price, product price, stock price etc. Some of the following ML methods could be used for solving regressions problems:

3.4.2 Statistical Arbitrage

In finance, statistical arbitrage refers to automated trading strategies that are typical of a short-term and involve many securities. In such strategies, the user tries to implement a trading algorithm for a set of securities based on quantities such as historical correlations and general economic variables. These measurements can be cast as a classification or estimation problem. The basic assumption is that prices will move towards a historical average.

We apply machine learning methods to obtain an index arbitrage strategy. We employ linear regression and support vector regression (SVR) onto the prices of an exchange-traded fund and a stream of stocks. By using principal component analysis (PCA) in reducing the dimension of feature space, we observe the benefit and note the issues in the application of SVR. To generate trading signals, we model the residuals from the previous regression as a mean reverting process.

In the case of classification, the categories might be sold, buy or do nothing for each security. In the case of estimation one might try to predict the expected return of each security over a future time horizon. In this case, one typically needs to use the estimates of the expected return to make a trading decision (buy, sell, etc.)

3.5 Architecture of Proposed System

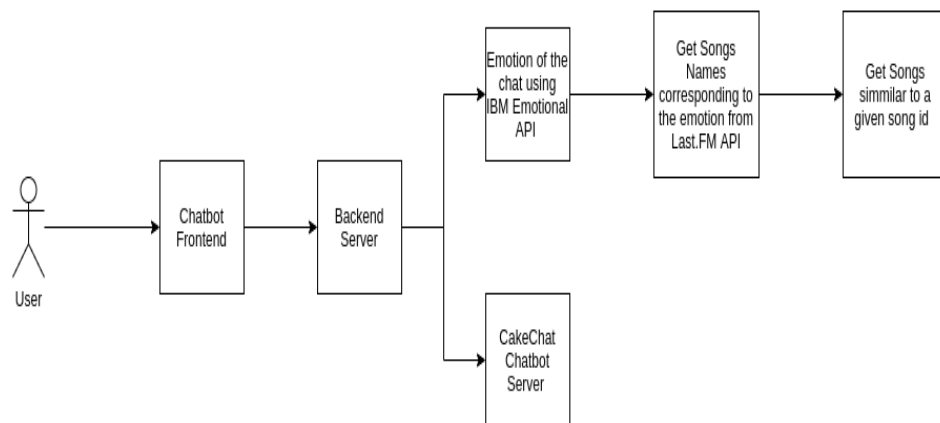


Fig 3.5.1 Architecture of proposed system.

Architecture of above fig shows that the user had to chat with our artificial chat bot and using backend server it guess the emotion of the user based on our chatting style and send the information to IBM emotional API and recommend a song based on our emotion.

CHAPTER 4

EXPERIMENTAL OR MATERIALS AND METHODS; ALGORITHMS USED

4.1 INSTALLATION PROCESS OF VISUAL STUDIO

Before you begin installing Visual Studio:

1. Check the system requirements. These requirements help you know whether your computer supports Visual Studio 2022.
2. Apply the latest Windows updates. These updates ensure that your computer has both the latest security updates and the required system components for Visual Studio.
3. Reboot. The reboot ensures that any pending installs or updates don't hinder your Visual Studio install.
4. Free up space. Remove unneeded files and applications from your system drive by, for example, running the Disk Clean up app.

STEP-1 Download Visual Studio

Next, download the Visual Studio bootstrapper file. To do so, select the following button, choose the edition of Visual Studio that you want, and then save to your Downloads folder.

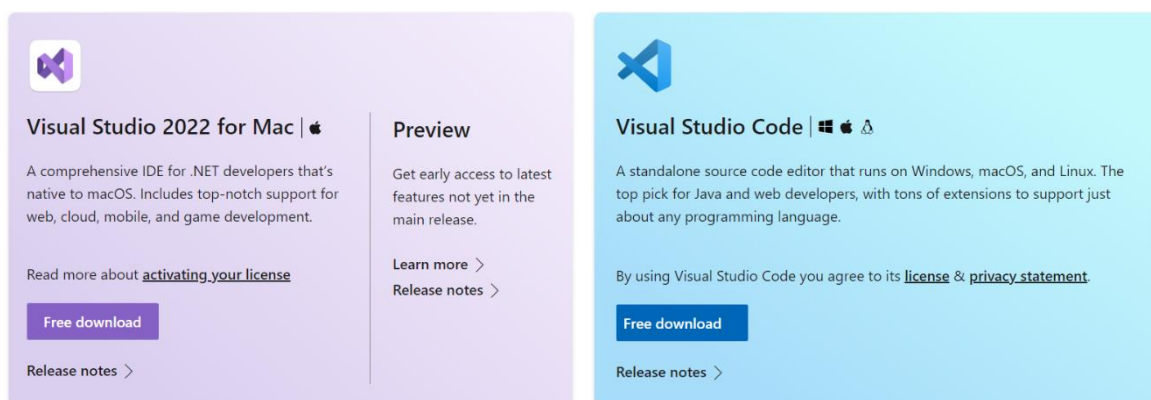


Fig.no 4.1.1 Downloading the Visual Studio Code.

STEP-2 Install the Visual Studio Installer

Run the bootstrapper file to install the Visual Studio Installer. This new lightweight installer includes everything you need to both install and customize Visual Studio. Acknowledge the Microsoft License Terms and the Microsoft Privacy Statement. Choose Continue.

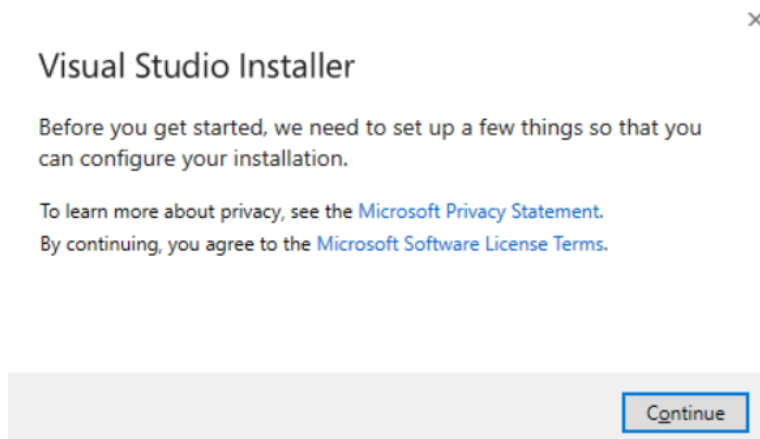


Fig no 4.1.2 Installing the Visual Studio Code.

STEP-3 Choose workloads.

After the installer is installed, you can use it to customize your installation by selecting the feature sets—or workloads—that you want.

- Now install Python extension workload in Visual Studio.

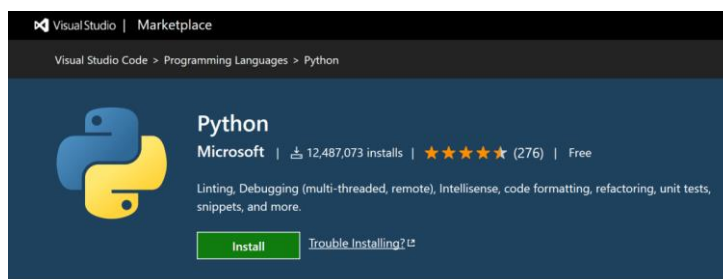


Fig no 4.1.3 Installation of python in Visual Studio Code.

- Install Jupyter Notebook workload in Visual Studio.

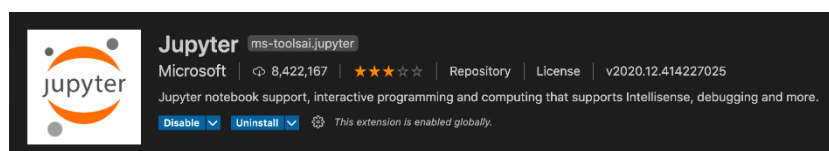


Fig.no 4.1.4 Installation of Jupyter Notebook in Visual Studio Code.

4.2 DEEP NEURAL NETWORKS METHOD

Deep neural networks (DNNs) yield state-of-the-art performance in numerous applications in the field of machine learning and artificial intelligence. Compared to traditional machine learning algorithms such as support vector machines, perceptron's, decision trees, and k-nearest neighbours, DNNs have significant advantages in extracting features at different levels of abstraction and thereby learning more complex patterns. DNNs compute their internal parameters in forward pass and then iteratively refine them during backpropagation to effectively extract input data features. These advantages give DNNs a greater learning capability, outperforming other methods in tasks such as computer vision, natural language processing, machine translation, speech recognition, genomics, quantitative trading, and self-driving cars. DNNs have become a powerful and valuable tool in many industrial and commercial applications. Recently, DNNs have achieved spectacular success across diverse fields: the AlphaGo and Alpha Zero algorithms defeated human world champions in the game of Go, online entertainment platforms use DNNs to construct highly effective systems for personalized content recommendation , and medical professionals utilize deep learning tools to make diagnoses and discover new drugs.

However, as the complexities of the learning tasks and the size of the training data grow, wider utilization of DNNs is challenged by a prohibitively large number of parameters, long training and inference time, and extensive computational and memory resources. For example, the ResNet-50 model contains about 23 million trainable floating-point parameters, with a model size of roughly 100 MB. Such models are not viable for some applications and implementations with strict resource constraints such as edge consumer and industrial devices.

The parameters of typical deep learning models often have a certain degree of redundancy (or sparsity in a transform domain), which can be exploited to reduce the model's computational cost and memory footprint. In traditional approaches, there is an efficiency–accuracy trade-off: using fewer parameters reduces the cost, but also compromises the model's accuracy. Prior works considered construction of compact models while maintaining performance accuracy, such as sparsity constraints, weight clustering and quantification, knowledge distillation, and structured projections. There are many existing research works on addressing these challenges using tensor-based networks.

In this chapter, we focus on the utilization of tensor decompositions and tensor networks. To address the challenges in deep learning, we utilize tensor networks for efficient representations of high-order tensors. This approach enables compression of the parameter tensors in DNNs while preserving the interactions between features (modes). Tensor operations enable us to conduct forward and backward update calculations efficiently in tensor format. Moreover, we exploit the power of parallelization on GPUs to accelerate neural network training and inference. These techniques will enable DNNs to be used in a broader range of applications by lowering the hardware requirements and computational cost and enable deployment on portable smart devices and embedded systems.

Deep neural networks have recently become the standard tool for solving a variety of computer vision problems. Whereas training a neural network is outside the OpenVX scope, importing a pretrained network and running inference on it is an important part of the OpenVX functionality. The concept of the Graph API of nodes representing functions and links representing data is very convenient for implementing deep neural networks with OpenVX. In fact, each neural network unit can be represented as a graph node. OpenVX has a special datatype representing tensors to provide data exchange between these nodes, and the nodes themselves are implemented in the *OpenVX Neural Network Extension*. Another way to import a neural network into OpenVX is by using the *OpenVX Kernel Import Extension*. The Kernel Import Extension can take a pretrained network model and load it into OpenVX as a single node. One of the data formats that can be used is *Neural Network Exchange Format* (NNEF), the standard also developed by the Khronos Group.

4.3 DEVELOPMENT OF DNN-BASED PREDICTION MODEL

A DNN-based prediction model was developed to predict the exhaustion behaviour exhibited during textile dyeing procedures. Typically, a DNN is a machine learning algorithm based on an artificial neural network (ANN) which mimics the principles and structure of a human neural network. An ANN is composed of an input layer, a hidden layer, and an outer layer. If the number of hidden layers is greater than or equal to three, the system is denoted as a DNN. Figure 2 shows a typical DNN structure. During regression analysis or classification problems, a linear estimation function of the type y

$= w^T x + b$ is used to solve the linear problem. Typically, a DNN applies an activation function to a linear estimation function to solve nonlinear problems.

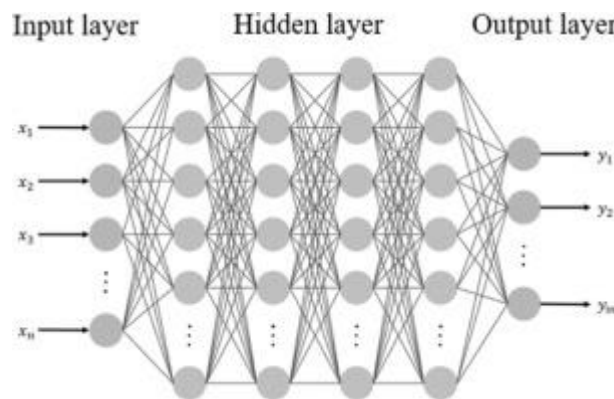


Fig.no 4.3.1 DNN Model.

4.4 METHOD FOLLOWED

MODULE 1: DATA COLLECTION AND PRE-PROCESSING

4.4.1 Data Collection

Data collection refers to the process of gathering information from various sources, which is then used to develop deep learning. The information should be saved in a way that is appropriate for the challenge. We collect data from Kaggle. We train and test our model using this dataset.

Data Collection is one of the most important tasks in building a machine learning model. It is the gathering of task related information based on some targeted variables to analyze and produce some valuable outcome. However, some of the data may be noisy, i.e., may contain inaccurate values, incomplete values, or incorrect values. Hence, it is must to process the data before analyzing it and coming to the results.

The dataset was constructed by beginning with songs and some sentences that could tell the emotions and relate them to songs based on different emotions; some of songs describes admiration, adoration, appreciation of beauty, amusement, anger, anxiety, awe, awkwardness, boredom, calmness, confusion, craving, disgust, empathic pain, entrancement, excitement, fear, horror, interest, joy, nostalgia, relief, sadness, satisfaction, and surprise. The sentences are first collected and then classify the sentences based on emotions of the sentence. The objective of this task is to determine, given a vector that describes the contents of a emotion of the sentence obtained from a dataset.

There are twenty five different classes, and the objective is to differentiate between emotions using the characteristics of a particular word that belongs to emotion. For example “I have the greatest admiration for all those involved in the project” is a sentence belongs to the category of admiration.

4.4.2 Data pre-processing

The stage of data pre-processing, also known as data purification, is an important one. Prior to developing the model, most machine learning engineers spend a significant amount of effort on data pre-processing. Outlier detection, missing value treatments, and noisy data removal are some examples of things that can be done during the pre-processing stage of data management. Data pre-processing can be done by data cleaning, data transformation, data selection. Data cleaning Fill in missing values, smooth noisy data, identify or remove outliers, and resolve inconsistencies. Data transformation may include smoothing, aggregation, generalization, transformation which improves the quality of the data. Data selection includes some methods or functions which allow us to select the useful data for our system.

If information is measured in terms of entropy, then these actions do not make the image more informative; rather, they make it less informative. The purpose of pre-processing is to improve the picture data by eliminating unwanted distortions or enhancing certain visual elements that are important for the subsequent processing and analysis tasks. This is the goal of the process. Data pre-processing is a data mining technique used to turn the raw data into a format that is both practical and effective. Data pre-processing changes the data into a format that can be processed in data mining, machine learning, and other data science tasks more quickly and efficiently. To ensure reliable findings, the techniques are typically applied at the very beginning of the machine learning and AI development pipeline. Although there are many various tools and strategies for preparing data.

4.4.1 Removing the html tags.

An HTML tag is a piece of markup language used to indicate the beginning and end of an HTML element in an HTML document.

As part of an HTML element, HTML tags help web browsers convert HTML documents into web pages. For example, the `<p>` tag is used to organize text content into *paragraph* elements and the `<img>` tag is used to embed *image* elements.

Many tags, though not all, use an opening tag and closing tag to wrap around the content that they are used to modify. Closing tags are denoted with a backslash like this: `</tag_name>`. HTML tags are not visible in the browser.

As we are uploading a sentence on the website it would be in the form of these html tags for the computer understanding. So, by removing these tags of the sentence we will be getting the main sentence to concentrate. For example, "I'm sad" this would be in the form of `<head>I'm sad</head>` after performing the remove function we will be having sentence in the form of I'm sad.

4.4.2 Removing punctuations and numbers.

Punctuation is the tool that allows us to organize our thoughts and make it easier to review and share our ideas. The standard English punctuation is as follows: period, comma, apostrophe, quotation, question, exclamation, brackets, braces, parenthesis, dash, hyphen, ellipsis, colon, semicolon. Below is an explanation of some punctuation that is commonly misused.

A number is an arithmetic value used for representing the quantity and used in making calculations. A written symbol like "3" which represents a number is known as numerals. A number_system is a writing system for denoting numbers using digits or symbols in a logical manner.

4.4.3 Removing single characters.

A character is a single visual object used to represent text, numbers, or symbols. For example, the letter "A" is a single character. With a computer, one character is one byte (8 bits). Such type of single characters is removed from a sentence.

4.4.4 Removing multiple spaces.

Multiple spaces are nothing but, the spaces between the words of sentence. These spaces must be removed for neglecting the extra unwanted symbols that occur in the sentence when converted into numbers.

4.5 IMPORTING LIBRARIES

Python library

Machine Learning, as the name suggests, is the science of programming a computer by

which they can learn from different kinds of data. A more general definition given by Arthur Samuel is – “Machine Learning is the field of study that gives computers the ability to learn without being explicitly programmed.” They are typically used to solve various types of problems in life. In the older days, people used to perform Machine Learning tasks by manually coding all the algorithms and mathematical and statistical formula. This made the process time consuming, tedious, and inefficient. But in the modern days, it has become very much easy and efficient compared to the olden days by various python libraries, frameworks, and modules. Today, Python is one of the most popular programming languages for this task and it has replaced many languages in the industry, one of the reasons is its vast collection of libraries. Python libraries that are used in Machine Learning are:

- Numpy
- Scipy
- Scikit-learn.
- Theano
- TensorFlow
- Keras
- PyTorch
- Pandas
- Matplotlib

4.5.1 Numpy

NumPy is a very popular python library for large multi-dimensional array and matrix processing, with the help of a large collection of high-level mathematical functions. It is very useful for fundamental scientific computations in Machine Learning. It is particularly useful for linear algebra, Fourier transform, and random number capabilities. High-end libraries like TensorFlow use NumPy internally for manipulation of Tensors.

4.5.2 Scipy

SciPy is a very popular library among Machine Learning enthusiasts as it contains different modules for optimization, linear algebra, integration and statistics. There is a difference between the SciPy library and the SciPy stack. The SciPy is one of the core packages that make up the SciPy stack. SciPy is also very useful for speech manipulation.

4.5.3 Skikit-learn.

Skikit-learn is one of the most popular ML libraries for classical ML algorithms. It is built on top of two basic Python libraries, viz., NumPy and SciPy. Scikit-learn supports most of the supervised and unsupervised learning algorithms. Scikit-learn can also be used for data mining and data-analysis, which makes it a great tool who is starting out with ML.

4.5.4 Theano

We all know that Machine Learning is basically mathematics and statistics. Theano is a popular python library that is used to define, evaluate and optimize mathematical expressions involving multi-dimensional arrays in an efficient manner. It is achieved by optimizing the utilization of CPU and GPU. It is extensively used for unit-testing and self-verification to detect and diagnose different types of errors. Theano is a very powerful library that has been used in large-scale computationally intensive scientific projects for a long time but is simple and approachable enough to be used by individuals for their own projects.

4.5.5 TensorFlow

TensorFlow is a very popular open-source library for high performance numerical computation developed by the Google Brain team in Google. As the name suggests, TensorFlow is a framework that involves defining and running computations involving tensors. It can train and run deep neural networks that can be used to develop several AI applications. TensorFlow is widely used in the field of deep learning research and application.

4.5.6 Keras

Keras is a very popular Machine Learning library for Python. It is a high-level neural networks API capable of running on top of TensorFlow, CNTK, or Theano. It can run seamlessly on both CPU and GPU. Keras makes it really for ML beginners to build and design a Neural Network. One of the best things about Keras is that it allows for easy and fast prototyping.

4.5.7 PyTorch

PyTorch is a popular open-source Machine Learning library for Python based on Torch, which is an open-source Machine Learning library which is implemented in C with a wrapper in Lua. It has an extensive choice of tools and libraries that supports on Computer Vision, Natural Language Processing(NLP) and many more ML programs.

It allows developers to perform computations on Tensors with GPU acceleration and also helps in creating computational graphs.

4.5.8 Pandas

Pandas is a popular Python library for data analysis. It is not directly related to Machine Learning. As we know that the dataset must be prepared before training. In this case, Pandas comes handy as it was developed specifically for data extraction and preparation. It provides high-level data structures and wide variety tools for data analysis. It provides many inbuilt methods for groping, combining and filtering data.

4.5.9 Mathpoltlib

Matpoltlib is a very popular Python library for data visualization. Like Pandas, it is not directly related to Machine Learning. It particularly comes in handy when a programmer wants to visualize the patterns in the data. It is a 2D plotting library used for creating 2D graphs and plots. A module named pyplot makes it easy for programmers for plotting as it provides features to control line styles, font properties, formatting axes, etc. It provides various kinds of graphs and plots for data visualization, viz., histogram, error charts, bar chats, etc.,

4.6 MODEL MAKING AND TRAINING

4.6.1 Model making

Machine learning models are powerful tools used to perform vital tasks and solve complex problems efficiently and effectively. An exponential increase in data across the modern world means organizations from a range of sectors are ready to deploy machine learning models. These models have a huge range of uses, whether machine learning in finance proactively monitoring bank transfers for signs of fraud, or machine learning in healthcare powering the next generation of diagnostic tools.

The process of building a machine learning model is often complex, driven by specialists in data science. But an understanding of the process is important as machine learning is adopted by more and more organizations. This guide explores the basics of building a machine learning model, breaking the process up into six steps.

Data input

After finding the best algorithm we are used that algorithm for finding the rain fall. Then we are going to give a input to the algorithm and we will find the output based on the output.

Convolution

It computes the output of the neurons linked to particular regions in the input volume, so that each neuron computes the dot product between its weights and the area of the input volume to which it is connected. Let's say we want to use 12 filters, which would give us a volume of [323212].

One_hot preprocessing

One-hot encodes a text into a list of word indexes of size n. This function receives as input a string of text and returns a list of encoded integers each corresponding to a word (or token) in the given input string.

Pad_sequences Preprocessing

Pads_sequences to the same length. This function transforms a list (of length num_samples) of sequences (lists of integers) into a 2D Numpy array of shape (num_samples, num_timesteps). num_timesteps is either the maxlen argument if provided, or the length of the longest sequence in the list.

Sequential model

A Sequential model is appropriate for a plain stack of layers where each layer has exactly one input tensor and one output tensor. The Sequential model API is a way of creating deep learning models where an instance of the Sequential class is created and model layers are created and added to it.

Activation layer

Activations can either be used through an Activation layer, or through the activation argument supported by all forward layers. Applies the rectified linear unit activation function.

With default values, this returns the standard ReLU activation: $\max(x, 0)$, the element-wise maximum of 0 and the input tensor.

Modifying default parameters allows you to use non-zero thresholds, change the max value of the activation, and to use a non-zero multiple of the input for values below the threshold.

Dropout layer

The Dropout layer randomly sets input units to 0 with a frequency of rate at each step during training time, which helps prevent overfitting. Inputs not set to 0 are scaled up by $1 / (1 - \text{rate})$ such that the sum over all inputs is unchanged.

Note that the Dropout layer only applies when training is set to True such that no values are dropped during inference. When using `model.fit`, training will be appropriately set to True automatically, and in other contexts, you can set the kwarg explicitly to True when calling the layer.

Dense layer

Dense implements the operation: $\text{output} = \text{activation}(\text{dot}(\text{input}, \text{kernel}) + \text{bias})$ where activation is the element-wise activation function passed as the activation argument, kernel is a weights matrix created by the layer, and bias is a bias vector created by the layer (only applicable if `use_bias` is True).

Flatten layer.

Keras. layers. flatten function flattens the multi-dimensional input tensors into a single dimension, so you can model your input layer and build your neural network model, then pass those data into every single neuron of the model effectively. You can understand this easily with the fashion MNIST dataset.

LSTM layer (Long Short-Term Memory layer)

Based on available runtime hardware and constraints, this layer will choose different implementations (cuDNN-based or pure-TensorFlow) to maximize the performance. If a GPU is available and all the arguments to the layer meet the requirement of the cuDNN kernel (see below for details), the layer will use a fast cuDNN implementation.

Bidirectional layer

Bidirectional recurrent layers are defined as connecting two hidden layers of the opposite directions to same output. Because of this generative deep learning, output layer gets the information from past or backwards and the future or forward states simultaneously.

Conv1d layer

1D convolution layer (e.g., temporal convolution). This layer creates a convolution kernel that is convolved with the layer input over a single spatial (or temporal) dimension to produce a tensor of outputs. If use_bias is True, a bias vector is created and added to the outputs.

GRU layer (Gated Recurrent Unit)

The GRU, known as the Gated Recurrent Unit, is an RNN architecture, which is similar to LSTM units. The GRU comprises of the reset gate and the update gate instead of the input, output and forget gate of the LSTM.

Pooling Layer

This layer is utilized in the process of performing a downsampling operation along the spatial dimensions (width, height), which ultimately produces a volume with the dimensions [16x16x12]. The Pooling layer is responsible for reducing the spatial size of the Convolved Feature. This is to decrease the computational power required to process the data through dimensionality reduction. Furthermore, it is useful for extracting dominant features which are rotational and positional invariant, thus maintaining the process of effectively training the model.

It is used to extract features that are invariant to rotation and position. Pooling can be categorized into two types i.e.

1. Max Pooling
2. Avg Pooling.

After the convolution and pooling layer, the model is enabled to understand and extract features from an image. Max Pooling returns the maximum value from the portion of the image covered by the Kernel. On the other hand, Average Pooling returns the average of all the values from the portion of the image covered by the Kernel.

Max Pooling also performs as a Noise Suppressant. It discards the noisy activations altogether and performs de-noising along with dimensionality reduction. On the other hand, Average Pooling simply performs dimensionality reduction as a noise suppressing mechanism. Hence, we can say that Max Pooling performs a lot better than Average Pooling.

The Convolutional Layer and the Pooling Layer, together form the i -th layer of a Convolutional Neural Network. Depending on the complexities in the images, the number of such layers may be increased for capturing low levels details even further, but at the cost of more computational power. After going through the above process, we have successfully enabled the model to understand the features. Moving on, we are going to flatten the final output and feed it to a regular Neural Network for classification purposes.

Embedding layer

Keras provides an embedding layer that converts each word into a fixed-length vector of defined size. The one-hot-encoding technique generates a large sparse matrix to represent a single word, whereas, in embedding layers, every word has a real-valued vector of fixed length.

Tokenizer preprocessing

Tokenization is the process of dividing text into a set of meaningful pieces. These pieces are called tokens. For example, we can divide a chunk of text into words, or we can divide it into sentences. Tokenizers convert raw string input into integer input suitable for a Keras Embedding layer. They can also convert back from predicted integer sequences to raw string output.

Sk_learn model.

Scikit-learn (Sklearn) is the most useful and robust library for machine learning in Python. It provides a selection of efficient tools for machine learning and statistical modeling including classification, regression, clustering and dimensionality reduction via a consistency interface in Python.

LabelBarizer preprocessing.

Scikit-learn's LabelBinarizer converts input labels into binary labels, each example belongs to a single class or not. Scikit-learn's MultiLabelBinarizer converts input labels into multilabel labels, each example can belong to multiple classes.

Creating a RESNET Model

Also, these networks have performed fairly in various deep learning competitions with very less error percentage. They also solve the problem generally observed in large networks that is with increase in depth of networks, saturation and degradation in accuracy start taking place. We have tried to 42 employ the best practices for training

the models. We have used cyclic learning rates that change their values cyclically between epochs and differential learning rates that change their values according to layers such that the training rate varies layer-wise as initial layers usually represent primitive features and inner layers represent high-level features. This helps in improving the accuracy greatly and reducing overfitting if any.

Defining Loss function and Optimizer

Loss function and optimizer are the two most deciding factors for a model being a great model or an ill-trained model, we have used Categorical Cross Entropy Loss as Loss function and Adam as optimizer.

The term "Loss Function" refers to the function that is responsible for computing this inaccuracy. As a result of this, the performance of the model is significantly impacted by the choice of loss function, as different loss functions will provide unique errors for the same prediction. The mean square error is one of the loss functions that is used the most frequently. This function works by calculating the square of the difference between the actual value and the anticipated value. When dealing with different kinds of tasks, such as regression and classification, different loss functions need to be utilized.

Error can be understood as a function of the model's internal parameters, which include weights and bias. It is necessary to reduce the amount of estimated error in order to make accurate predictions. Back propagation is the method that is used to accomplish this in neural networks. Typically, the current error is "propagated backwards" to an earlier layer, where it is utilized to make adjustments to the weights and bias in order to ensure that the error is reduced to its smallest possible value. A function known as the Optimization Function is used to make adjustments to the weights.

We are looking for measurements or a function that we can apply to improve the performance of our model. The loss function indicates the accuracy of our model's predictions. Loss will be at its lowest if model projections are closest to the actual values, and at its highest if predictions are farthest from the original values. We are best served by a categorical cross entropy loss function because our model is based on DNN.

The output label is assigned one-hot category encoding value in form of 0s and 1. The output label, if present in integer form, is converted into categorical encoding using keras. The score that penalizes the probabilities based on how far they are from the

predicted value is then calculated. Depending on how near or far the value is to the actual value.

Next, we use Adam optimizer. An algorithm for gradient descent minimization is called adaptive moment estimation. When dealing with complex problems involving a lot of data or factors, the strategy is incredibly effective. It is effective and uses little memory. It combines the "gradient descent with momentum" algorithm and the "RMSP" algorithm, intuitively.

4.6.2 Training the model.

Training a model simply means learning (determining) good values for all the weights and the bias from labeled examples. In supervised learning, a machine learning algorithm builds a model by examining many examples and attempting to find a model that minimizes loss; this process is called empirical risk minimization.

Loss is the penalty for a bad prediction. That is, loss is a number indicating how bad the model's prediction was on a single example. If the model's prediction is perfect, the loss is zero; otherwise, the loss is greater. The goal of training a model is to find a set of weights and biases that have *low* loss, on average, across all examples. For example, Figure 3 shows a high loss model on the left and a low loss model on the right. Note the following about the figure:

- The arrows represent loss.
- The blue lines represent predictions.

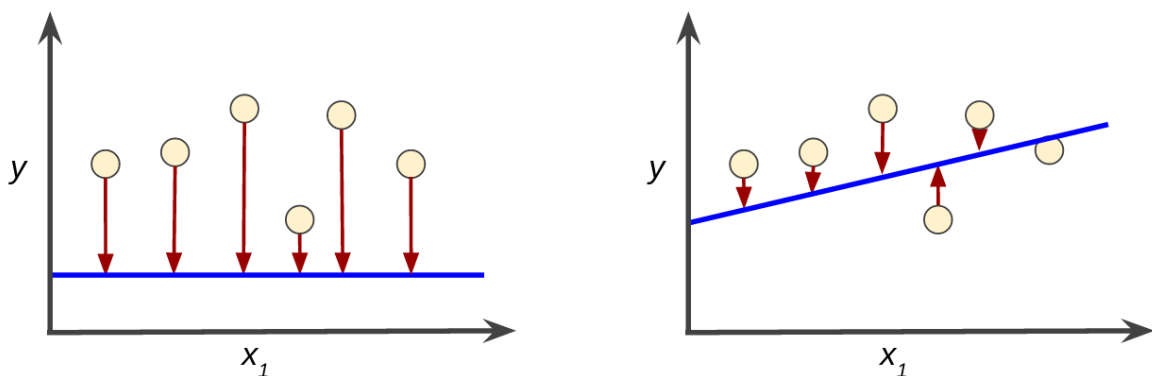


FIG.NO 4.6.1 Training Model.

Notice that the arrows in the left plot are much longer than their counterparts in the right plot. Clearly, the line in the right plot is a much better predictive model than the line in the left plot.

You might be wondering whether you could create a mathematical function—a loss function—that would aggregate the individual losses in a meaningful fashion.

Squared loss: a popular loss function.

The linear regression models we'll examine here use a loss function called **squared loss** (also known as **L₂ loss**).

Mean square error (MSE) is the average squared loss per example over the whole dataset. To calculate MSE, sum up all the squared losses for individual examples and then divide by the number of examples:

$$MSE = \frac{1}{N} \sum_{(x,y) \in D} (y - \text{prediction}(x))^2$$

where:

- (x, y) is an example in which.
 - X is the set of features (for example, chirps/minute, age, gender) that the model uses to make predictions.
 - Y is the example's label (for example, temperature).
- $\text{Prediction}(x)$ is a function of the weights and bias in combination with the set of features x .
- D is a data set containing many labeled examples, which are (x,y) pairs.
- N is the number of examples in D .

Although MSE is commonly used in machine learning, it is neither the only practical loss function nor the best loss function for all circumstances.

4.7 Flask

Flask is a popular open-source Python web application framework that allows developers to build web applications quickly and efficiently.

Flask is designed to be lightweight and modular, giving developers the flexibility to choose the components they need for their project. It provides a simple and intuitive interface for building web applications and offers features such as routing, template rendering, form handling, and session management.

One of the advantages of using Flask is its extensibility through its rich ecosystem of extensions, which can help developers with tasks such as database integration, authentication, and API development. Flask also integrates well with other popular Python libraries and frameworks, making it a versatile choice for web development.

Overall, Flask is a powerful tool for building web applications in Python, with a low barrier to entry and a high degree of flexibility.

Why Flask

We need a webserver to run our program in a website . so we can use flask

Using flask we can render html pages ,angular frame works and also css file .

With flask we can easily do routing

The interesting fact In the flask is we don't need any separate file to create file . we can directly use it in our python file it self . what we have to do is we have to import flask module from flask and we have to create a app and run it .

```
from flask import Flask, request
from flask_cors import CORS
```

Cors is for security purposes.

Whenever we run the flask application it creates a live webserver using different ports like 8000,5000 we can directly click that link to render our project in the browser or we can directly go the browser and just open localhost:(port number) to render our project.

```
PS C:\Users\asus\Downloads\final> python -u "c:\Users\asus\Downloads\final\chatbot_music\app.py"
* Serving Flask app 'app'
* Debug mode: on
WARNING: This is a development server. Do not use it in a production deployment. Use a production WSGI server instead.
* Running on http://127.0.0.1:8000
Press CTRL+C to quit
* Restarting with stat
* Debugger is active!
* Debugger PIN: 469-817-570
```

With using flask we can directly use routings from one page to another easily.

```
@app.route('/reset', methods=['GET', 'POST'])
def reset():
    if request.method == 'POST':
        session['message_list'] = []
        session['emotion_1'] = ""
        session['emotion_2'] = ""
```

And we can also able to use methods like [GET,POST] using API . we can get the data from one server to another and also post from one another.

We can use debug==true to refresh automatically without refreshing everytime whenever we edit our code.

```
if __name__ == '__main__':
    app.run(debug=True, port=8000)
```

CHAPTER – 5

RESULTS AND DISCUSSION, PERFORMANCE ANALYSIS

5.1 RESULT

Hence, the data is collected and saved in a data set. The code has been written and analyzed through google visual studio. The web page is designed to get the results. There are two types of data stored in the data set one is emotions related songs and the other is the emotion detecting sentences. This data is trained for a several times using a deep learning algorithm and then tested by the deep learning algorithm. Then the tested data is used as the true data and used to train the data for the further purposes. When we upload a sentence into the web page then by the algorithm which we developed reads the sentence and gives the respective result for the emotion of sentence and plays the songs based on that emotion.

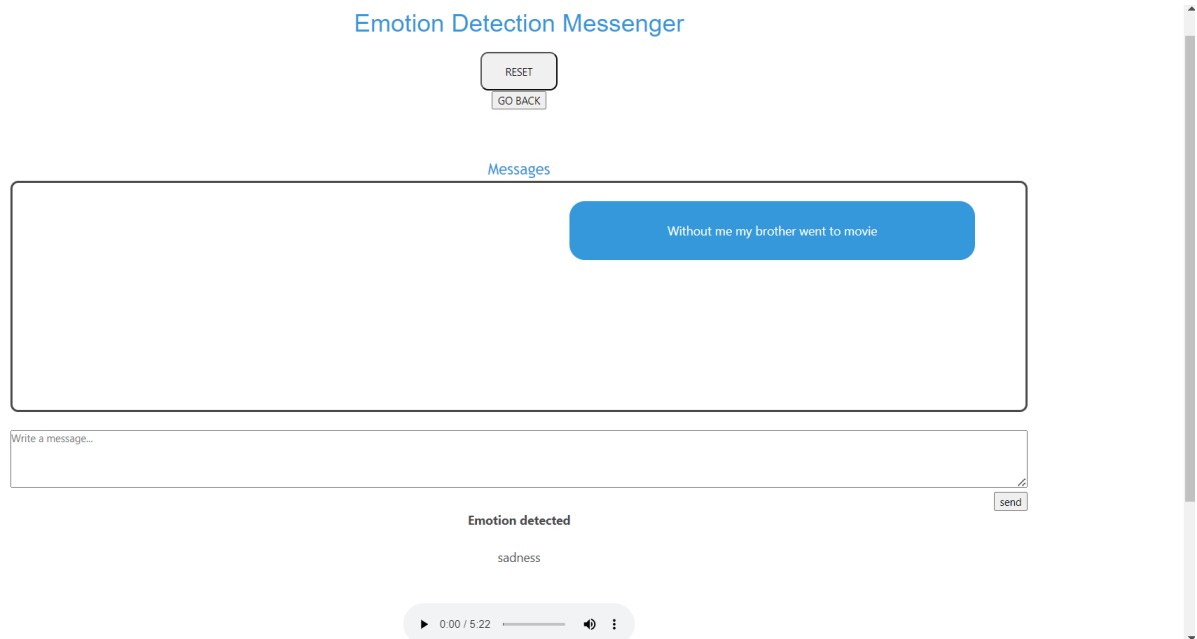


Fig.no:5.1.1 The emotion sadness is detected.

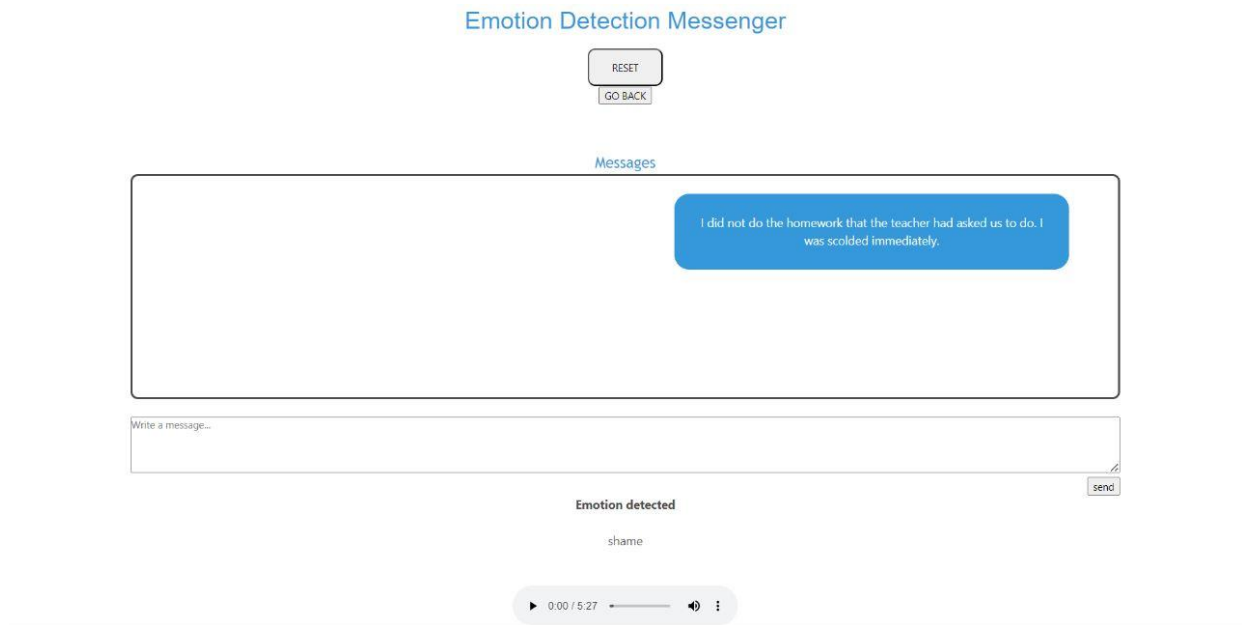


Fig.no:5.1.2 The emotion shame is detected.

5.2 DISSCUSSIONS

These choices must be made in all applications, and they undoubtedly have a profound effect on the success, or appropriateness, of learning. Yet the exact nature of this effect is unknown, and a systematic study of these aspects is needed.

The first decision concerned granularity. In our application three different approaches are possible. One of them works with the whole sentence and its output simply states whether the given sentence contains an emotion or not. The second approach works with the main words related to sentence and provides the user with songs recommending of those emotion related. Finally, the third approach classifies individual messages, phone calls, chats and this data is stored in a dataset and recommend the songs by analyzing the data. The approach operating with the main words represents the finest granularity, whereas the approach operating with sentence represents the coarsest granularity.

In this project we have used the second approach that is operating with main words of a sentence that represent the finest granularity. Finer granularity provides more examples, compare the millions of sentences, with the 34,000 feelings and 27 emotions. Moreover, in our application a higher misclassification rate can be tolerated at the emotion detection

level.80% accuracy on the emotions in a human is likely to identify more than 80% of the emotions. On the other hand, emotions can be described only with an impoverished set of features, and the result need not necessarily seem coherent to the user. We decided the system would classify emotions.

Our decision to classify emotions had important consequences for the general design of the system. Together with the granularity of the interface, which according to Robert Plutchik requirement was at the sentence level, it has constrained the options concerning the output of our system. We could not have ranked emotions according to their probability of being happy, sad, admired, tired, fear etc. because our unit of output was a single emotion-based song. We could not have classified feelings, because our system was to support decisions about emotions rather than feelings.

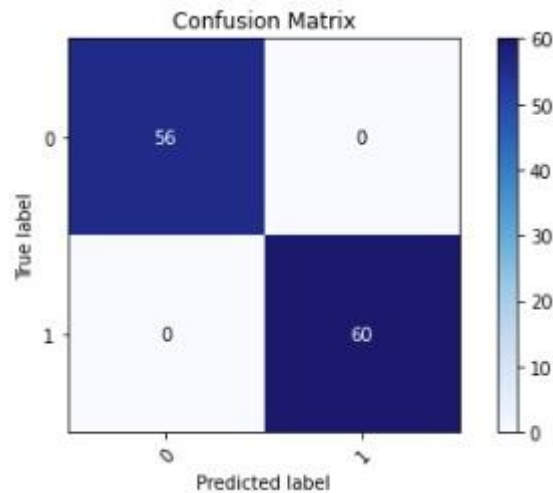
We classify the emotions based on the confusion matrix which gives the matrix for “TRUE” and “PREDICTED” values.

		PREDICTED:	
		Positive	Negative
TRUE	Positive	A	B
	Negative	C	D

Table. No 5.1.1 Confusion matrix.

In this confusion matrix the A, B, C and D represents the following:

- A True positive: The number of times our actual positive values are equal to the predicted positive. You predicted a positive value, and it is correct.
- B False negative: The number of times our model wrongly predicts negative values as positives. You predicted a negative value, and it is actually positive.
- C False positive: The number of times our model wrongly predicts negative values as positives. You predicted a negative value, and it is actually positive.
- D True negative: The number of times our actual negative values are equal to predicted negative values. You predicted a negative value, and it is actually negative.

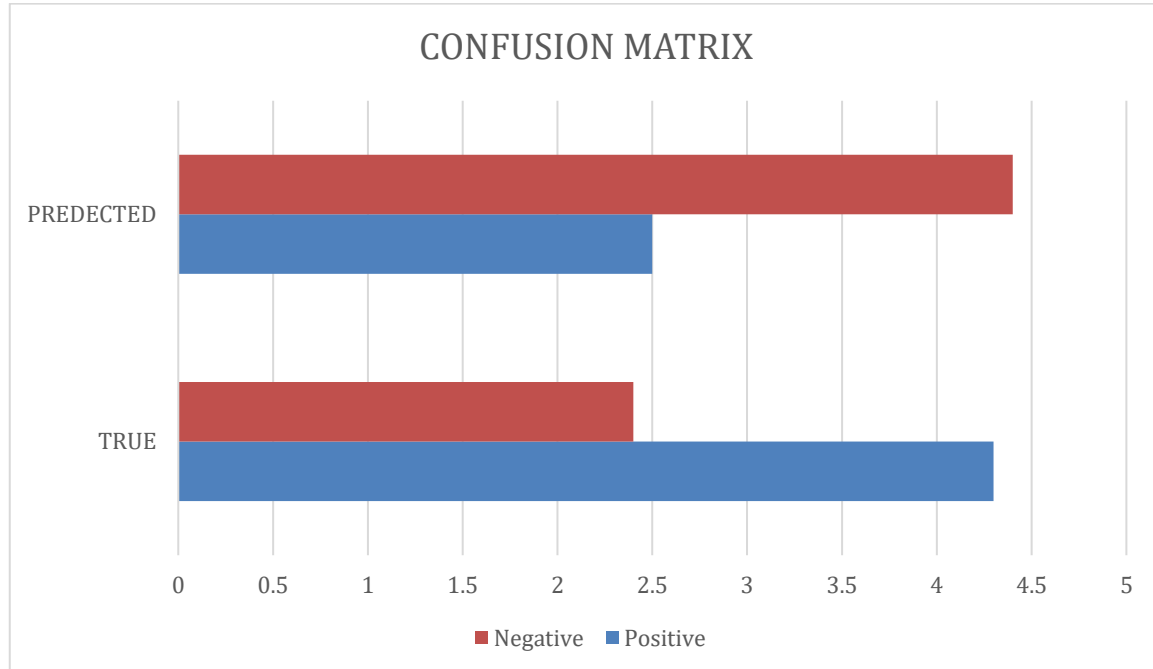


Graph.No 5.1.1 Confusion matrix.

By Fig.no 5.1.1 we can notice that the ratio of the true and predicted values is 15:14 where, the true value is 15 and the predicted value is 14.

The accuracy of true and predicted values is known by the formula:

$$\text{Accuracy} = \frac{TP+TN}{TP+TN+FP+FN}$$



Graph.no 5.1.2 Confusion matrix analysis.

5.2.1 TRAINED AND TESTED DATA

Supervised Learning (SL) is intended as the way an ML model is trained by a human being. This means that the human (the Data Scientist, the Machine Learning Engineer,

or whatever it is) trains the ML model in a set of known data.

Typically, machine learning algorithms accept parameters that can be used to control certain properties of the training process and of the resulting ML model. In Amazon Machine Learning, these are called training parameters. You can set these parameters using the Amazon ML console, API, or command line interface (CLI). If you do not set any parameters, Amazon ML will use default values that are known to work well for a large range of machine learning tasks.

You can specify values for the following training parameters:

- Maximum model size
- Maximum number of passes over training data.
- Shuffle type
- Regularization type
- Regularization amount

TESTING PROCESS

We all make mistakes and if left unchecked, some of these mistakes can lead to failures or bugs that can be very expensive to recover from. Testing our code helps to catch these mistakes or avoid getting them into production in the first place. Testing therefore is very important in software development. Used effectively, tests help to identify bugs, ensure the quality of the product and to verify that the software does what it is meant to do.

TYPES OF TESTS

Unit Testing

Unit testing includes the design of test belongings that validate that the internal program logic is operative properly, and that program input produces valid outputs. All choice branches and internal code flow should be authorized. It is the testing of separate software units of the request .it is done after the close of an individual unit before integration. This is a structural testing, that relies on data of its structure and is invasive. Unit tests achieve basic tests at factor level and test a specific commercial process, application, and/or system formation. Unit tests ensure that each single path of business process completes accurately to the documented provisions and contains clearly defined inputs and probable results.

Integration Testing

Integration tests are calculated to test integrated software components to regulate if they actually run as one program. Testing is occasion driven and is more concerned with the basic result of screens or fields. Integration tests validate that although the workings were individually approval, as shown by positively unit testing, the grouping of components is correct and dependable. Integration testing is specifically aimed at revealing the problems that arise from the grouping of components.

Functional Testing

Functional tests provide systematic protests that functions tested are accessible as stated by the business and technical necessities, system certification and user guides.

Functional difficult is centered on the subsequent items:

Valid Input is used to identify classes of valid input that must be accepted.

Invalid Input is used to identify classes of illegal input that must be disallowed.

Functions are used to identify purposes that must be exercised.

Output is used to classify modules of request outputs.

Systems/Procedures are used to interfacing systems or events must be appealed.

Organization and grounding of functional tests is focused on supplies, key functions, or special test belongings. In addition, systematic coverage pertaining to identify Business process flows; data fields, predefined processes, and succeeding processes must be well-thought-out for testing. Before functional testing is complete, supplementary tests are identified and the operative value of recent tests is resulted.

System Testing

System testing confirms that the entire combined software system meets supplies. It tests a configuration to ensure known and predictable outcomes. An example of system testing is the arrangement-oriented system mixing test. System testing is based on procedure similes and flows, emphasizing pre-driven process links and addition points.

White Box Testing

White Box Testing is a challenge in which the software tester has information of the inner workings, construction and language of the software, or at least its drive. It is used to test areas that cannot be stretched from a black box level.

Black Box Testing

Black Box Testing is testing software short of any knowledge of the inner mechanisms, building or language of the module actuality tested. Black box tests, like most other kinds of tests, must be printed from a final source document, such as requirement or necessities file, such as specification or requirement file. It is a testing in which the software below test is treated as a black box .you cannot “see” into it. The test provides inputs and responds to outputs without seeing how the software works.

5.2.2 TEST STRATEGY AND APPROACH

Field testing will be performed by hand and practical tests will be written in the feature.

Test Objectives

- All field entries must work properly.
- Pages must start from the recognized link.
- The access screen, messages and replies must not be delayed.
- Features to be verified.
- Verify that the accesses are of the correct setup.
- No replacement passes should be allowed.
- All links must take the user to the right page.

Integration Testing

Software incorporation testing is the incremental addition testing of two or more combined software workings on a single platform to produce disappointments caused by interface faults.

The task of the combination test is to check that workings or software submissions.

Acceptance Testing

User Reception Testing is a serious phase of any project and needs important contributions from the end user. It also confirms that the system meets practical necessities.

ALPHA TESTING

In software expansion, alpha test will be a test amongst the teams to confirm that your creation works. Originally, the term alpha test was destined for the first stage of testing in a software development process. The first phase covers component testing, module

testing, and scheme testing. It also allows us to test the produce on the last common denominator tackles to make sure transfer times are suitable and preloads work.

BETA TESTING

In software advance, a beta test is the second opinion of software challenging in which a selection of the planned viewers tries the product out. Beta testing can be restrained from "pre-release testing." Beta test changes of software are now spread to curriculum establishments and teachers to give the database a "real-world" test.

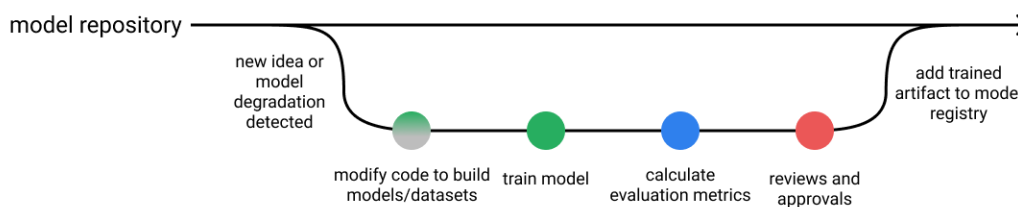
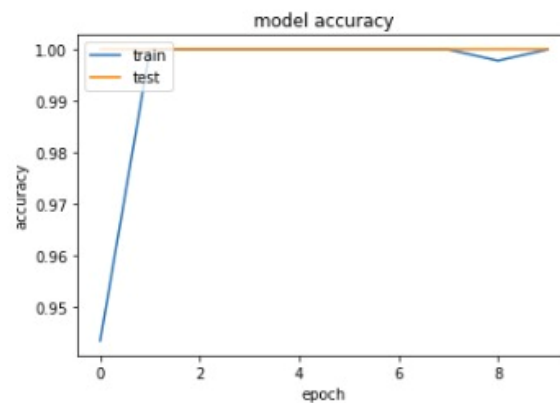


Fig.no 5.2.1 A typical workflow of a model development.



Graph. No 5.2.1 Model accuracy.

5.3 PERFORMANCE ANALYSIS

5.3.1 LONG SHORT-TERM MEMORY NETWORKS(LSTM)

LSTMs are a type of Recurrent Neural Network (RNN) that can learn and memorize long-term dependencies. Recalling past information for long periods is the default behaviour.

LSTMs retain information over time. They are useful in time-series prediction because they remember previous inputs. LSTMs have a chain-like structure where four interacting

layers communicate in a unique way. Besides time-series predictions, LSTMs are typically used for speech recognition, music composition, and pharmaceutical development.

- First, they forget irrelevant parts of the previous state.
- Next, they selectively update the cell-state values.
- Finally, the output of certain parts of the cell state

Below is a figure 5.3.1 of how LSTMs operate:

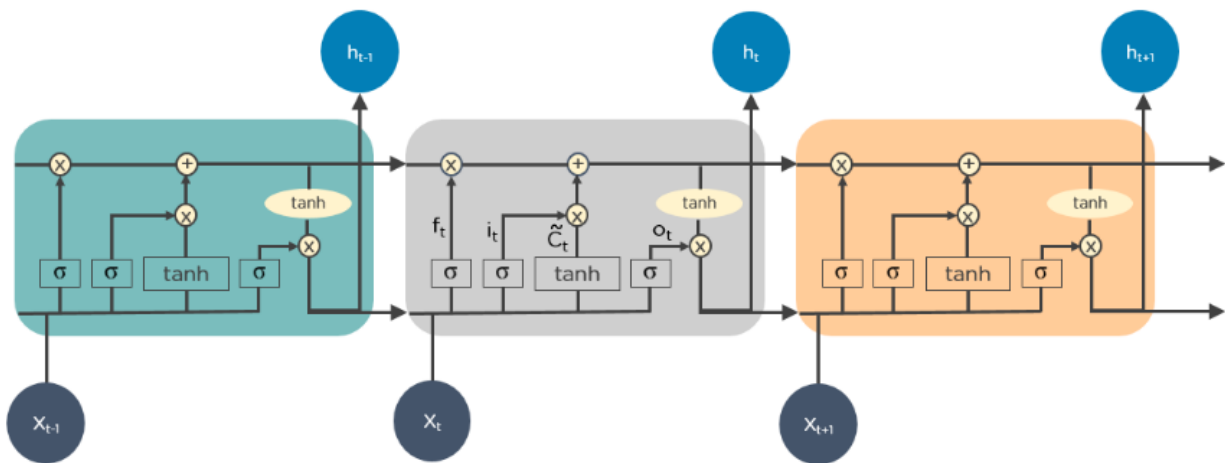


Fig.no:5.3.1 Operation of LSTM.

Long short-term memory (LSTM) is an artificial neural network used in the fields of artificial intelligence and deep learning. Unlike standard feedforward neural networks, LSTM has feedback connections. Such a recurrent neural network (RNN) can process not only single data points (such as images), but also entire sequences of data (such as speech or video). This characteristic makes LSTM networks ideal for processing and predicting data. For example, LSTM is applicable to tasks such as unsegmented, connected handwriting recognition, speech recognition, machine translation, speech activity detection, robot control, video games, and healthcare.

The name of LSTM refers to the analogy that a standard RNN has both "long-term memory" and "short-term memory". The connection weights and biases in the network change once per episode of training, analogous to how physiological changes in synaptic strengths store long-term memories; the activation patterns in the network change once

per time-step, analogous to how the moment-to-moment change in electric firing patterns in the brain store short-term memories. The LSTM architecture aims to provide a short-term memory for RNN that can last thousands of timesteps, thus "long short-term memory".

A common LSTM unit is composed of a cell, an input gate, an output gate and a forget gate. The cell remembers values over arbitrary time intervals and the three *gates* regulate the flow of information into and out of the cell. Forget gates decide what information to discard from a previous state by assigning a previous state, compared to a current input, a value between 0 and 1. A (rounded) value of 1 means to keep the information, and a value of 0 means to discard it. Input gates decide which pieces of new information to store in the current state, using the same system as forget gates. Output gates control which pieces of information in the current state to output by assigning a value from 0 to 1 to the information, considering the previous and current states. Selectively outputting relevant information from the current state allows the LSTM network to maintain useful, long-term dependencies to make predictions, both in current and future time-steps.

LSTM networks are well-suited to classifying, processing and making predictions based on time series data, since there can be lags of unknown duration between important events in a time series. LSTMs were developed to deal with the vanishing gradient problem that can be encountered when training traditional RNNs. Relative insensitivity to gap length is an advantage of LSTM over RNNs, hidden Markov models and other sequence learning methods in numerous applications.

5.3.2 DIFFERENT TYPES OF EMOTIONS DETECTED BY LSTM MODEL

Shame: A painful emotion caused by consciousness of guilt, shortcoming, or impropriety.

E.g. I felt shame when I found that I was almost failed in exam.

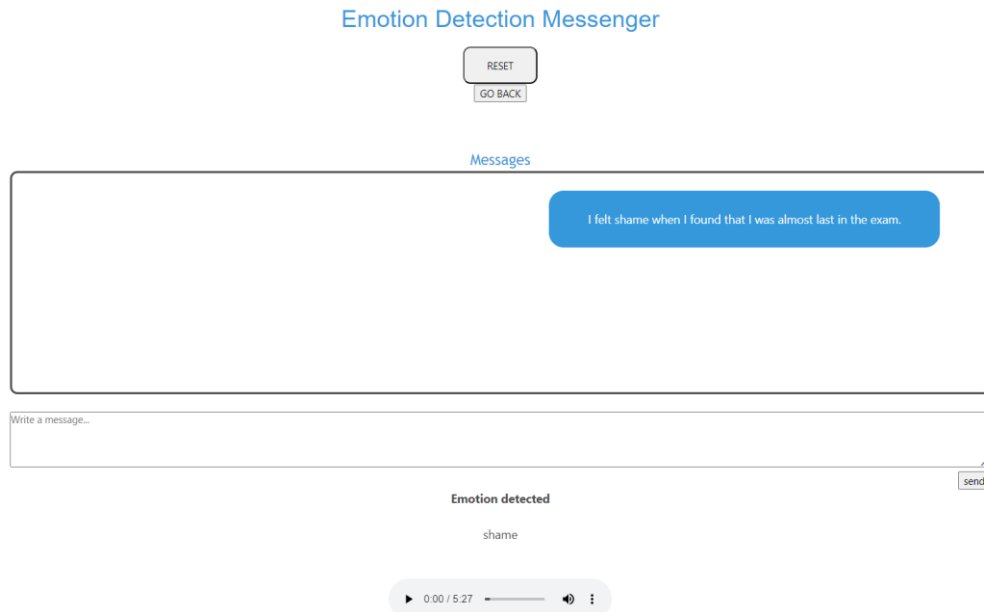


Fig.no: 5.3.2 Detection of shame.

Fear: It's painful agitation in the presence or anticipation of danger. Fear is the most general term and implies anxiety and usually loss of courage.

E.g. When one has arranged to meet someone and that person arrives late, in the meantime one starts thinking about all that could have gone wrong.

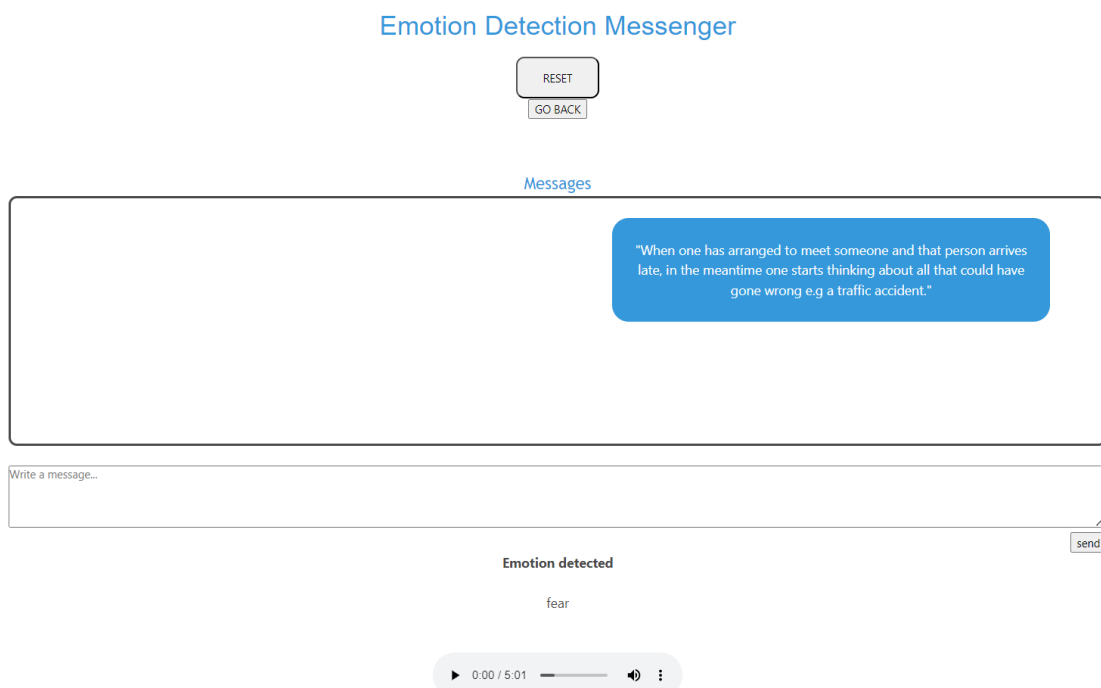


Fig.no: 5.3.3 Detection of fear.

Joy: A feeling of great pleasure or happiness that comes from success, good fortune, or a sense of well-being.

E.g. Finally, she accepted my request.

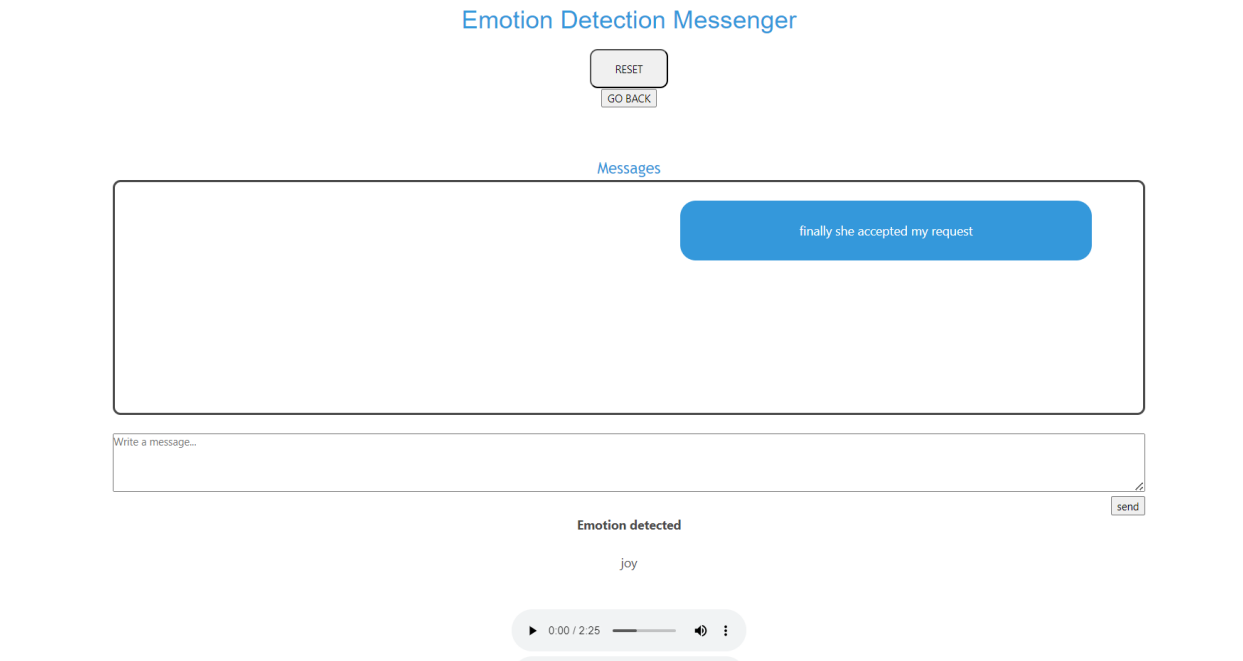


Fig.no 5.3.4 Detection of joy.

Anger: An emotion characterized by antagonism toward someone or something you feel has deliberately done you wrong.

E.g. A stranger beat me without my mistake.

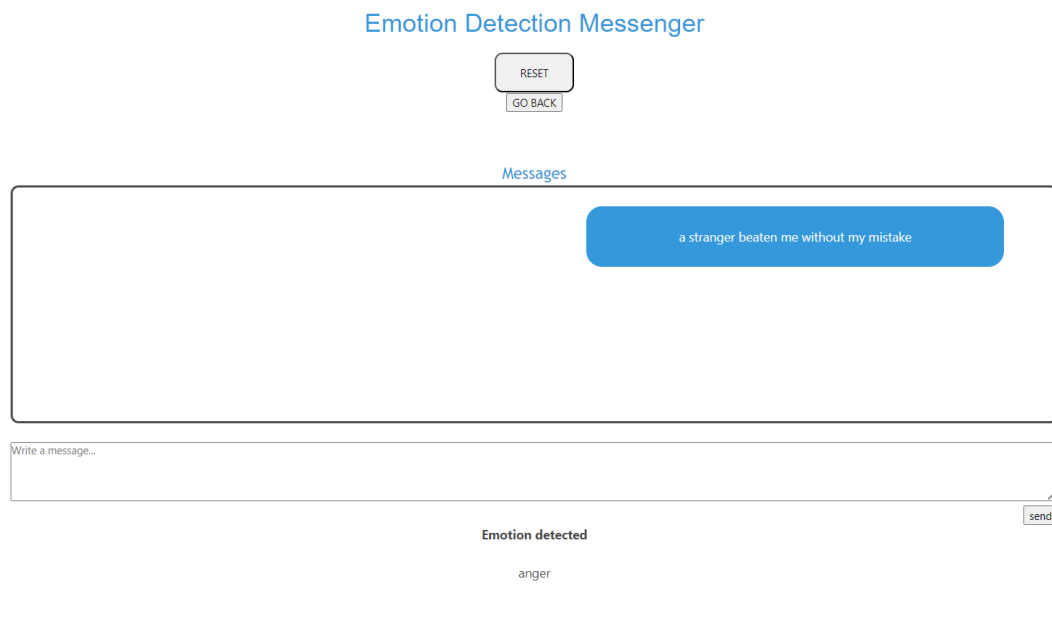


Fig.no: 5.3.5 Detection of anger.

Sadness: An emotional pain associated with, or characterized by, feelings of disadvantage, loss, despair, grief, helplessness, disappointment and sorrow.

E.g. My mom said that I'm the worst guy.

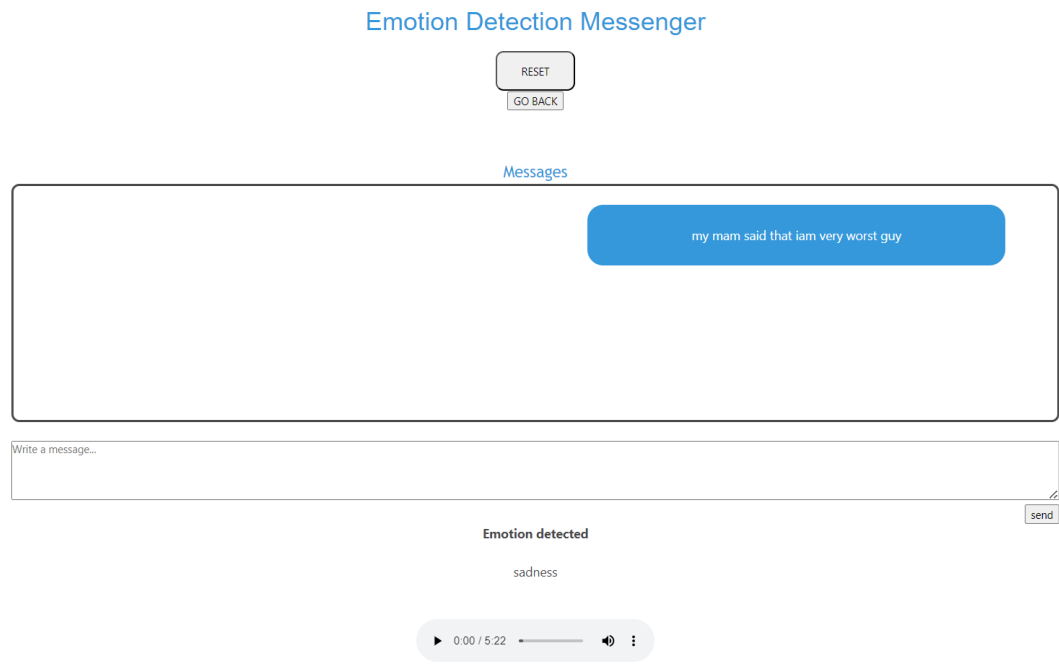


Fig.no: 5.3.6 Detection of Sadness.

CHAPTER – 6

SUMMARY AND CONCLUSIONS

We have presented a survey and methodology for building the chatbot song recommender system. To perform this, we first identified various approaches for building a chatbot known to date. We then evaluated the considered algorithms which are useful in building of our system in terms of their ability to work on the recommendation process of the system. We also gathered all the requirements needed for building our system and studied the overall process involved in chatbot's working. Lastly, we summarized the deployment requirements of our system. On the conclusion note our 'Chatbot Song Recommender System' is used to facilitate the use by physically challenged people to automate and give them better music player experience. The application solves the basic needs of music listeners without troubling them as existing applications do. We have generated a good quality song recommendation, but there is a huge scope of improvement in it. First, starting and ending song recommendations can be added in every newly generated tune to give a tune a better start and better ending. By doing this, our generated song recommendation will become melodious. Second, the model can be trained with more tunes. Here, we have trained our model with only 405 song recommendable tunes. By training the model with more song recommendable tunes, our model will not only expose to more variety of song recommend but the number of classes will also increase. By this more melodious and at the same time more variety of song recommendation can be generated through the model. Third, models can also be trained with multi-instrument tunes. As of now, the song recommends generated is of only one piece of instrument. It would be interesting to listen to what song recommend the model will produce if it is trained on multi-instrument song recommend. Finally, a method can be added into the model which can handle unknown notes in the song recommend. By filtering unknown notes and replacing them with known notes, models can generate more robust quality song recommendations.

REFERENCES

- [1] Aioli, F. (2012). A preliminary study on a recommender system for the million songs dataset challenge. PREFERENCE LEARNING: PROBLEMS AND APPLICATIONS IN AI
- [2] Anagha S.Dhavalikar, and Dr. R. K. Kulkarni, "Face Detection and Facial Expression Recognition System," International Conference on Electronics and Communication System (ICECS - 2014), 2014.
- [3] Arto Lehtiniemi and Jukka Holm, "Using Animated Mood Pictures in Music Recommendation," 16th International Conference on Information Visualisation, 2012.
- [4] Cremonesi, Paolo, Yehuda Koren, and Roberto Turrin. "Performance of recommender algorithms on topn recommendation tasks." Proceedings of the fourth ACM conference on Recommender systems. ACM, 2010
- [5] C. L. Krumhansl and D. L. Schenck, 'Can dance reflect the structural and expressive qualities of music? A perceptual experiment on Balanchine's choreography of Mozart's divertimento No. 15,' *Musicae Scientiae*, vol. 1, no. 1, pp. 63–85, Mar. 1997.
- [6] E. J. Humphrey, J. P. Bello, and Y. LeCun, "Moving beyond feature design: deep architectures and automatic feature learning in music informatics," in Proc. 13th Int'l Conf. Music Info. Retrieval, pp. 403- 408, October 2012.
- [7] F. Abdat, C. Maaoui and A. Pruski, "Human-computer interaction using emotion recognition from facial expression," UKSim 5th European Symposium on Computer Modelling and Simulation, 2011.
- [8] F. Ofli, E. Erzin, Y. Yemez, and A. M. Tekalp, "Learn2Dance: Learning statistical music-to-dance mappings for choreography synthesis," *IEEE Trans. Multimedia*, vol. 14, no. 3, pp. 747–759, Jun. 2012.
- [9] G.T. Shrivakshan, and C. Chandrasekar. "A comparison of various edge detection techniques used in image processing," *International Journal of Computer Science Issues (IJCSI)*, vol. 9(5), pp. 272-276, 2012.
- [10] J. B. Schafer, D. Frankowski, J. Herlocker, and S. Sen, "Collaborative filtering recommender systems," *The Adaptive Web: Methods and Strategies of Web Personalization*, pp. 291-324, 2007.
- [11] K. Lu and Y. Jia, "Audio-Visual Emotion Recognition With Boosted Coupled HMM," no. *Icpr*, pp. 1148–1151, 2012.

- [12] Koren, Yehuda. "Recommender system utilizing collaborative filtering combining explicit and implicit feedback with both neighborhood and latent factor models."
- [13] Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep
- [14] Mahiux Ellis 2-11 <http://labrosa.ee.columbia.edu/millionsong/tasteprofile>
- [15] M. Bejani, D. Gharavian, and N. M. Charkari, "Audiovisual emotion recognition using ANOVA feature selection method and multiclassifier neural networks," Neural Comput. Appl., vol. 24, no. 2, pp. 399–412, 2014.
- [16] McFee, B., BertinMahieux,T., Ellis, D. P., Lanckriet, G. R. (2012, April). The million-song dataset challenge. In Proceedings of the 21st international conference companion on World Wide Web (pp. 909916).ACM.
- [17] M. J. Pazzani and D. Billsus, "Contentbased recommendation systems," TheAdaptive Web: Methods and Strategies ofWeb Personalization, pp. 325-341, 2007.
- [18] M. Lee, K. Lee, and J. Park, 'Music similarity-based approach to generating dance motion sequence,' Multimedia Tools Appl., vol. 62, no. 3, pp. 895–912, Feb. 2013.
- [19] R. E. Fan, K. W. Chang, C. J. Hsieh, X. R. Wang, and C. J. Lin, "LIBLINEAR: a library for large linear classification," J. Machine Learning Research, vol. 9, pp. 1871- 1874, 2008.
- [20] Renuka R. Londhe, Dr. Vrushshen P. Pawar, "Analysis of Facial Expression and Recognition Based On Statistical Approach," International Journal of Soft Computing and Engineering (IJSCE) vol. 2, May 2012.
- [21] R. C. Gonzalez and R. E. Woods, "Digital Image Processing", Pearson Prentice Hall, 2008.
- [22] R. Fan, S. Xu, and W. Geng, "Example-based automatic music-driven conventional dance motion synthesis," IEEE Trans. Vis. Compute. Graphics, vol. 18, no. 3, pp. 501– 515, Mar. 2012.
- [23] Sparse Matrices <http://docs.scipy.org/doc/scipy/reference/sparse.html>
- [24] S. Rani, D. Bansal, and B. Kaur, "Detection of edges using mathematical morphological operators", Open Transactions on Information Processing, vol. 1(1), pp. 17- 26, 2014.
- [25] S. Yang and B. Bhanu, "Facial Expression Recognition Using Emotion Avatar Image," pp. 866–871, 2011.
- [26] T. Bertin et al., The Million Song Dataset, Proc. of the 12th International Society for Music Information Retrieval Conference, 2011

- [27] T.-H. Wang and J.-J.J. Lien, "Facial Expression Recognition System Based on Rigid and Non-Rigid Motion Separation and 3D PoseEstimation," J. Pattern Recognition, vol. 42, no. 5, pp. 962-977, 2009.
- [28] V. Oord, S. Dieleman, and B. Schrauwen, "Deep content- based music recommendation," in Proc. 26th Int'l Conf. Neural Info.Process. Systems, pp.2643-2651, December 2013.
- [29] W. Hsu, and C. J. Lin, "A comparison of methods for multiclass support vector machines,"IEEE Tran. Neural Networks, vol. 13, no. 2, pp. 415-425, 2002.
- [30] Y.-H. Su, "Rhythm of music seen through danc: Probing music–dance coupling by audio visual meter perception,' in Proc. OSF, Nov. 2017, pp. 1–20.
- [31] Yong-Hwan Lee, Woori Han and Youngseop Kim, "Emotional Recognition from Facial Expression Analysis using Bezier Curve Fitting," 16th International Conference on Network-Based Information Systems, 2013.
- [32] Z. Qiyuan, H. Xianxiang, T. Lilong, and Z. Bing, "A method of deleting noise in a binary image based on mathematical morphology," 8th International Conference on Electronic Measurement and Instruments, pp. 2-787 – 2-790, 2007.