SAVITRIBAI PHULE PUNE UNIVERSITY

A PROJECT REPORT ON

EXPRESSIVE ENGLISH
TEXT-TO-SPEECH SYNTHESIS SYSTEM

SUBMITTED TOWARDS THE
PARTIAL FULFILLMENT OF THE REQUIREMENTS OF

BACHELOR OF ENGINEERING (Computer Engineering)

BY

Pranjal Bhor	Roll No: B120054399
Vaibhav Chaudhari	Roll No: B120054254
Prathamesh Dharangutte	Roll No: B120054267
Murtaza Raja	Roll No: B120054367

Under The Guidance of

Dr. G. P. Potdar

DEPARTMENT OF COMPUTER ENGINEERING
Pune Institute of Computer Technology
Dhankawadi, Pune-411043

Pune Institute of Computer Technology
DEPARTMENT OF COMPUTER ENGINEERING

CERTIFICATE

This is to certify that the Project Entitled

Expressive english text-to-speech synthesis system

Submitted by

Pranjal Bhor	Roll No: B120054399
Vaibhav Chaudhari	Roll No: B120054254
Prathamesh Dharangutte	Roll No: B120054267
Murtaza Raja	Roll No: B120054367

is a bonafide work carried out by Students under the supervision of Dr. G. P. Potdar and it is submitted towards the partial fulfillment of the requirement of Bachelor of Engineering (Computer Engineering).

Dr. G. P. Potdar
Internal Guide
Dept. of Computer Engg.

Dr. R. B. Ingle
H.O.D
Dept. of Computer Engg.

Dr. P. T. Kulkarni
Principal
Pune Institute of Computer Technology

Signature of Internal Examiner

Signature of External Examiner

PROJECT APPROVAL SHEET

Expressive English Text-to-Speech Synthesis System

Is successfully completed by

Pranjal Bhor	Roll No: B120054399
Vaibhav Chaudhari	Roll No: B120054254
Prathamesh Dharangutte	Roll No: B120054267
Murtaza Raja	Roll No: B120054367

at

DEPARTMENT OF COMPUTER ENGINEERING

PUNE INSTITUTE OF COMPUTER TECHNOLOGY

SAVITRIBAI PHULE PUNE UNIVERSITY,PUNE

ACADEMIC YEAR 2016-2017

Dr. G. P. Potdar   Dr. R. B. Ingle
Internal Guide   H.O.D
Dept. of Computer Engg.   Dept. of Computer Engg.

Abstract

Many English text-to-speech conversion are available like Adobe Acrobat reader, Jovie and KMouth. But their output is extremely monotonic and robotic in nature. This limits their ability to be used for reading emotion rich novels. We try to solve this problem by using Machine Learning, Natural Language Processing and Digital Signal Processing concepts. Our system first predicts the emotions present in the text and then according to the emotion the system tries to modify the audio output of the text. For emotion prediction, we use a trained emotion classifier, which has been trained on the emotion datasets which are available publicly. Also, there is another system based on Natural Language Processing approach which tries to identify the arrangement of the sentence and based on which emotion of the sentence is predicted. The voice synthesis is done by a trained Hidden Markov Model based synthesis system. Presently, we identify and incorporate most prominent emotions in the text viz. neutral, joy, surprise, sadness and angry.

Acknowledgments

It gives us great pleasure in presenting the project report on **'Expressive English Text-to-Speech Synthesis System'**.

I would like to take this opportunity to thank my internal guide **Dr. G. P. Potdar** *for giving me all the help and guidance I needed. I am really grateful to him for his kind support. His valuable suggestions were very helpful.*

I am also grateful to **Dr. R. B. Ingle**, *Head of Computer Engineering Department, Pune Institute Of Computer Technology for his indispensable support, suggestions.*

In the end our special thanks to **Mr. Shadrach Karsulkar** *for providing various resources such as laboratory with all needed software platforms, continuous Internet connection, for Our Project.*

<div align="right">

Pranjal Bhor
Vaibhav Chaudhari
Prathamesh Dharangutte
Murtaza Raja
(B.E. Computer Engg.)

</div>

Contents

List of Figures

List of Tables

CHAPTER 1
SYNOPSIS

1.1   Project Title

Expressive English Text-to-speech synthesis system

1.2   Project Option

The project group members decided to opt for unsponsored in-house project. This led the members to find their own idea for the project.

1.3   Internal Guide

Dr. G. P. Potdar

1.4   Technical Keywords (As per ACM Keywords)

F: Theory Of Computation

F.2: ANALYSIS OF ALGORITHMS AND PROBLEM COMPLEXITY

F.2.1: Numerical Algorithms and Problems (G.1, G.4, I.1)
B. Computations on matrices
For TF-IDF representation, the feature vector of a particular sentence is represented as a sparse matrix using python scipy library. These matrices are then used for further calculations. In deep learning approach, the weights of a layer are internally represented using matrices. Various operations like convolution and element-wise addition are performed depending on the structure of the network.

I: Computing Methodologies

I.2: ARTIFICIAL INTELLIGENCE

I.2.7: Natural Language Processing
A. Text analysis
Text mining task of emotion detection is addressed in this project. It involves lexical analysis, part of speech tagging and term-weighting usng TF-IDF. For deep learning, word

embeddings are used to map a word to a continuous vector space.

B. Speech recognition and synthesis
In this project we synthesize expressive speech from input text. Synthesized speech can be created by concatenating pieces of recorded speech that are stored in a database. Systems differ in the size of the stored speech units; a system that stores phones or diphones provides the largest output range, but may lack clarity. The two primary technologies generating synthetic speech waveforms are concatenative synthesis and formant synthesis. Concatenative synthesis is based on the concatenation (or stringing together) of segments of recorded speech. Formant synthesis does not use human speech samples at runtime. Instead, the synthesized speech output is created using additive synthesis and an acoustic model. Parameters such as fundamental frequency, voicing, and noise levels are varied over time to create a waveform of artificial speech. Many systems based on formant synthesis technology generate artificial, robotic-sounding speech that would never be mistaken for human speech. As our main motivation behind the project was to generate human-like speech we use concatenative synthesis method.

I.5: PATTERN RECOGNITION

I.5.4: Applications

A. Signal Processing Signal Processing algorithms are used to perform operations on the neutral audio signal to synthesize expressive speech. Following are some of the functions performed: Speaking style and speaker identity adaptation supporting various voice conversion algorithms. Gaussian Mixture Model based voice conversion algorithms. Prosody transformation algorithms for voice conversion. Mean and standard deviation transformation of f0. Smoothing algorithms for voice conversion.

1.5 Problem Statement

Develop a software program that reads a text file containing English sentences such that the emotions in the sentences are reflected in the audio.

1.6 Abstract

Many English text-to-speech conversion are available like Adobe Acrobat reader, Jovie and KMouth. But their output is extremely monotonic and robotic in nature. This limits their ability to be used for reading emotion rich novels. We try to solve this problem by using Machine Learning, Natural Language Processing and Digital Signal Processing concepts. Our system first predicts the emotions present in the text and then according to the emotion the system tries to modify the audio output of the text. For emotion prediction, we use a trained emotion classifier, which has been trained on the emotion datasets which are available publicly. Also, there is another system based on Natural Language Processing approach which tries to identify the arrangement of the sentence and based on which emotion of the sentence is predicted. The voice synthesis is done by a trained Hidden Markov Model based synthesis system. Presently, we identify and incorporate most prominent emotions in the text viz. neutral, joy, surprise, sadness and angry.

1.7 Goals and Objectives

The goal of the project is to create a software system that will be able to identify emotions from english text and accordingly produce affective output. There are two objectives associated with our project. First objective is to identify emotion associated with each and every sentence of the text. And the second objective is to modify neutral speech to incorporate emotion associated with the text.

1.8 Relevant mathematics associated with the Project

Let S be the set denoting the system:
S = $\{s, e, X, Y, F_{me}, DD, NDD, S_c, F_c\}$
where,
s = start state

= Classifier for the sentences is trained

e = end state

= System gives speech with emotions as output

X = set of input

= {text file}

Y = set of output

= {speech with emotions}

F_{me} = set of functions

= $\{F_{ip}, F_{ml}, F_{clas.}, F_{s/p}, F_{op}, F_1, F_2, F_3\}$

where,

F_{ip} = Function to take input X

F_{ml} = Function to train the classifier using Machine Learning

$F_{clas.}$ = Function to classify the input text

$F_{s/p}$ = Function to synthesize the audio using Hidden Markov Model

F_{op} = Function to write the output speech file

F_1 = Get emotion using Naive Bayes Classifier

Naive Bayes classifiers are a family of simple probabilistic classifiers based on applying Bayes' theorem. As a general statement, we can state Bayes' theorem as follows

$$P(\omega_j|x) = \frac{p(x|\omega_j) \cdot P(\omega_j)}{p(x)} \tag{1.1}$$

The data is represented by $\mathbf{x}$ and labels are represented by ω_j.

F_2 = Get emotions using LSTM classifier

RNNs maintain a hidden vector $\mathbf{h}$, which is updated at time step t as follows:

$$\mathbf{h}_t = \tanh(\mathbf{W} * \mathbf{h}_{t-1} + \mathbf{I} * \mathbf{x}_t) \tag{1.2}$$

where tanh is the hyperbolic tangent function, $\mathbf{W}$ is the recurrent weight matrix and I is a projection matrix. The hidden state $\mathbf{h}$ is then used to make a prediction

$$\mathbf{y}_t = \text{softmax}(\mathbf{W} * \mathbf{h}_{t-1}) \tag{1.3}$$

where *softmax* provides a normalized probability distribution over the possible classes and $\mathbf{W}$ is a weight matrix.

Training vanilla RNNs is known to be particularly difficult.

LSTMs address the problems commonly found in training RNNs by incorporating gating functions into their state dynamics . At each time step, an LSTM maintains a hidden vector $\mathbf{h}$ and a memory vector $\mathbf{m}$ responsible for controlling state updates and outputs. More concretely, we define the computation at time step t as follows :

$$
\begin{aligned}
\mathbf{g}^u &= \sigma(\mathbf{W}^u * \mathbf{h}_{t-1} + \mathbf{I}^u * \mathbf{x}_t) \\
\mathbf{g}^f &= \sigma(\mathbf{W}^f * \mathbf{h}_{t-1} + \mathbf{I}^f * \mathbf{x}_t) \\
\mathbf{g}^o &= \sigma(\mathbf{W}^o * \mathbf{h}_{t-1} + \mathbf{I}^o * \mathbf{x}_t) \\
\mathbf{g}^c &= \tanh(\mathbf{W}^c * \mathbf{h}_{t-1} + \mathbf{I}^c * \mathbf{x}_t) \\
\mathbf{m}_t &= \mathbf{g}^f \odot + \mathbf{g}^u \odot \mathbf{g}^c \\
\mathbf{h}_t &= \tanh(\mathbf{g}^o \odot \mathbf{m}_{t-1})
\end{aligned}
\tag{1.4}
$$

F_3 = Function to generate audio using Hidden Markove Model The task is to compute in a best way, given the parameters of the model, the probability of a particular output sequence. This requires summation over all possible state sequences.The probability of observing a sequence:

$$
Y = y(0), y(1), ..., y(L-1)
\tag{1.5}
$$

of length L is given by

$$
P(Y) = \sum_X P(Y|X)P(X),
\tag{1.6}
$$

where the sum runs over all possible hidden-node sequences

$$
X = x(0), x(1), ...x(L-1)
\tag{1.7}
$$

here σ is the logistic sigmoid function, $\mathbf{W}^u, \mathbf{W}^f, \mathbf{W}^o, \mathbf{W}^c$ are recurrent weight matrices and $\mathbf{I}^u, \mathbf{I}^f, \mathbf{I}^o, \mathbf{I}^c$ are projection matrices.

DD = Deterministic Data
= ϕ
NDD = Non Deterministic Data
= {X}
S_c = Success case
= Y contains speech with emotions
F_c = Failure case
= $\overline{S_c}$

1.9 Names of Conferences / Journals where papers can be published

1. Association for Computational Linguistics: Human Language Technologies.

2. The SIGNLL Conference on Computational Natural Language Learning.

1.10 Review of Conference/Journal Papers supporting Project idea

1. Association for Computational Linguistics: Human Language Technologies conference is held to facilitate a platform for the researchers who work in Computational Linguistics. Research related to natural language understanding by the computer is published here. This is why, a research paper related to our topic can be published in this conference.

2. SIGNLL is a top-tier conference, yearly organized by SIGNLL (ACL's Special Interest Group on Natural Language Learning). The conference includes research papers from the field of Natural Language Processing which includes understanding of the natural language by computer and processing them. As our project deals with Natural Language Processing, this conference can be used as a platform to publish our research paper.

1.11 Plan of Project Execution

Task Name	Start	End	Days
Study of emotion classification algorithms	8/2	8/16	14
Collecting and cleaning datasets	8/18	8/30	14
Study of human audio features	9/1	9/8	7
Study of algorithms for modifying audio	9/9	10/4	25
Deciding implementation approach	10/6	11/10	35
Implementing classifier module	12/1	12/15	14
Implementing audio feature modification module	12/16	12/31	15
Performance improvement	1/2	1/14	12
Testing and deploying the project	1/15	1/30	15
Documentation and open source release of the project	2/1	2/26	25

CHAPTER 2
TECHNICAL KEYWORDS

2.1   Area of Project

The project titled "Expressive English text-to-speech synthesis system" belongs to the domains of Machine Learning, Natural Language Processing and Digital Signal Processing. Since, we are trying to predict the emotions from text, a machine learning classifier is used for that purpose. Also, there are some cases wherein we have to check the sentence structure of prediction of emotions. This is where we use Natural Language Processing. Digital Signal Processing is required to modify neutral speech audio into affective audio.

2.2   Technical Keywords

F: Theory Of Computation

F.2: ANALYSIS OF ALGORITHMS AND PROBLEM COMPLEXITY

F.2.1: Numerical Algorithms and Problems (G.1, G.4, I.1)
B: Computations on matrices
For TF-IDF representation, the feature vector of a particular sentence is represented as a sparse matrix using python scipy library. These matrices are then used for further calculations. In deep learning approach, the weights of a layer are internally represented using matrices. Various operations like convolution and element-wise addition are performed depending on the structure of the network.

I: Computing Methodologies

I.2: ARTIFICIAL INTELLIGENCE

I.2.7: Natural Language Processing
A: Text analysis
Text mining task of emotion detection is addressed in this project. It involves lexical analysis, part of speech tagging and term-weighting usng TF-IDF. For deep learning, word embeddings are used to map a word to a continuous vector space.

B. Speech recognition and synthesis
In this project we synthesize expressive speech from input text. Synthesized speech can be created by concatenating

pieces of recorded speech that are stored in a database. Systems differ in the size of the stored speech units; a system that stores phones or diphones provides the largest output range, but may lack clarity. The two primary technologies generating synthetic speech waveforms are concatenative synthesis and formant synthesis. Concatenative synthesis is based on the concatenation (or stringing together) of segments of recorded speech. Formant synthesis does not use human speech samples at runtime. Instead, the synthesized speech output is created using additive synthesis and an acoustic model. Parameters such as fundamental frequency, voicing, and noise levels are varied over time to create a waveform of artificial speech. Many systems based on formant synthesis technology generate artificial, robotic-sounding speech that would never be mistaken for human speech. As our main motivation behind the project was to generate human-like speech we use concatenative synthesis method.

I.5: PATTERN RECOGNITION

I.5.4: Applications

A. Signal Processing Signal Processing algorithms are used to perform operations on the neutral audio signal to synthesize expressive speech. Following are some of the functions performed: Speaking style and speaker identity adaptation supporting various voice conversion algorithms. Gaussian Mixture Model based voice conversion algorithms. Prosody transformation algorithms for voice conversion. Mean and standard deviation transformation of f0. Smoothing algorithms for voice conversion.

CHAPTER 3
INTRODUCTION

3.1 Project Idea

To create a software program, which will take English language text as input and then will identify emotions of each and every sentence present in the text. It will try to speak the sentences such that the detected emotions are incorporated in the audio.

3.2 Motivation behind the Project

The project group members are avid readers. We usually read novels and articles from various different sources. When we tried to use Adobe PDF reader's inbuilt 'read-out-aloud' feature for listening to the novel, we realized that it was highly monotonous and robotic in nature. This type of text-to-speech is unsuitable for listening to novels which are quite rich with different types of emotions. We then used some other text-to-speech engines like IvonaTTS, Jovie and KMouth (of Okular PDF reader). But the results were similar as that of Adobe PDF reader. Then we tried some other text-to-speech engines like MaryTTS and PSOLA. They have some ability to handle emotional audio synthesis, but they lack emotion detection facility. This inspired us to build a system which will first try to identify emotion associated with English text and then try to speak the sentences with emotions associated with it.

3.3 Literature Survey

Emotion in speech synthesis is an important aspect as it makes the speech more natural sounding and may help to convey more information about context apart from just neutral speech [1]. The focus of this paper is the manipulation of a robotic voice signal for the purpose of adding emotional expression. In particular, its main aim was to design the emotion expressed by a robotic voice by manipulating specific acoustic parameters such as pitch, amplitude and speech rate. The three basic emotions considered were: anger, happiness and sadness.

There are two main aspects of our project i.e emotion classification and modification of neutral speech according to the identified emotion [1]. [2] extensively focuses on identifying emotions from English text. The authors have developed 2 classifiers, one Naive-Bayes and other is Max-entropy, for identification of emotions. Also, they have used Stanford Parser's Univer-

sal dependency for identifying sentence construction and then predicting the emotions. These three emotion prediction sub-systems then vote on the emotions they have retrieved and the emotion which gets maximum votes is selected as the overall emotion of the sentence.The approach for emotion classification used in [2] provided good results.

Researchers in [3] improve upon [2] by providing a simple and memory optimized way for classifying emotions using customized decision tree. They have modified the already existing Digg dataset and SemEval Affective Text-2007 by adding a new feature, emotion intensity. [4] provided us with a novel way of vector representation. It discusses emotion categorization using word embeddings is discussed on Chinese news corpus. Word embeddings can capture semantic context and can be used to infer similarity between words. This approach requires very little feature engineering and yields substantial success. This increases the efficiency of emotion prediction in [3].

The other main module of our system is modifying the neutral speech waveform according to the predicted emotion. We first worked with formant synthesis in which various parameters of speech like f0, intensity, pitch etc are modified to obtain desired emotion [6]. This paper presents algorithms that allows a robot to express its emotions by modulating the intonation of its voice. It describes a technique which allows to continuously control both the age of a synthetic voice and the quantity of emotions that are expressed.

A system for emotion dectection and affective audio synthesis is presented in [6]. The formant synthesis technique did not provide satisfactory results for speech synthesis as discussed in [6] and [10]. All the papers before this dealt with a particular aspect of this project. This paper provides an integrated approach for classification of emotion and speech synthesis. It presents a Text-to-speech engine which identifies emotions from text using a Machine Learning classifier. Then a voice is synthesized using Hidden Markov Modelling technique [2]. The voice is modified using various emotions' characteristics classified using Self-Organizing Maps (SOM).

CHAPTER 4

PROBLEM DEFINITION AND SCOPE

4.1 Problem Statement

Develop a software program that reads a text file containing English sentences such that the emotions in the sentences are reflected in the audio.

4.1.1 Goals and objectives

The goal of the project is to create a software system that will be able to identify emotions from english text and accordingly produce affective output. There are two objectives associated with our project. First objective is to identify emotion associated with each and every sentence of the text. And the second objective is to modify neutral speech to incorporate emotion associated with the text.

4.1.2 Statement of scope

The project aims at identifying 4 different emotions (happy, angry, sad and neutral) from the english text and producing speech that incorporates the emotions in the output. The input of the project is English text or English PDF file. The output of the project is speech signal with emotions of the text incorporated in it. Also, there are huge possibilities of construction a sentence with English language. Our scope is limited to identifying emotions of simple English sentences with arguably contains only one emotion. Only one negation is allowed in the sentence which can or cannot qualify the affective words in the sentence.

4.2 Major Constraints

Large and accurate datasets for emotion classification using Machine Learning approaches are not available publicly because the researchers who create those datasets use university funding to crowdsource that job. We have used the datasets by requesting them for the same. So, dataset is extremely small by the standards required to train a highly accurate classifier.

Machine Learning performs lot of computations on the input data while training the model. Due to unavailability of faster computing devices, the classifier model training takes enormous amount of time.

Our system predicts only five different types of emotions namely, joy, sadness, surprise, anger and neutral due to the limitations posed by the dataset.

As most of the novels, storybooks and scripts use text or Portable Document Format (PDF), our system supports only the aforementioned formats for input.

The accent of the voice output of our system is US English. This is due to the dataset which we have used (CMU's Arctic voice dataset) to train the voice synthesis module.

4.3 Methodologies of Problem solving and efficiency issues

4.3.1 Emotion classification

In order to efficiently and accurately extract emotions from the input text, we extensively tried and used several techniques out of which the selected ones are discussed below.

4.3.1.1 Natural Language Processing (NLP)

This method involves understanding the English language, its subtleties, grammar, word synonyms, structure of sentences and efficiently processing each word so as to get the required output. This methology uses Stanford Parser[15][16] to create an Universal Dependency tree. This helps us in identifying dependency of words on each other. Part-of-speech (POS) of each word is identified using Tree Tagger Tool[16]. Emotion of each word is identified using NRC lexicon[12] word set which contains about 14000 words tagged with their emotions. We have used the $python-nltk$ package for the same.

4.3.1.2 Long Short Term Memory (LSTM)

A LSTM network is an artificial neural network that contains LSTM units instead of, or in addition to, other network units. A LSTM unit is a recurrent network unit that excels at remembering values for either long or short durations of time. The key to this ability is that it uses no activation function within its recurrent components. Thus, the stored value is not iteratively squashed over time, and the gradient or blame term does not tend to vanish when Backpropagation through time is applied to train it which helps in our

case. It efficiently identifies the structure of the sentences giving more accurate results than traditional neural networks for our dataset. We have used *sklearn* and *keras* python packages for training LSTM.

4.3.1.3 Naive Bayes

Naive Bayes is a simple technique for constructing classifiers: models that assign class labels to problem instances, represented as vectors of feature values, where the class labels are drawn from some finite set. In our case the class labels were the emotions. It is not a single algorithm for training such classifiers, but a family of algorithms based on a common principle: all naive Bayes classifiers assume that the value of a particular feature is independent of the value of any other feature, given the class variable which can be assumed safely for our work since we are working on independent sentences without considering the context in which the sentence appears. We have used *sklearn* python package for this classifier.

4.3.2 Affective Speech Synthesis

In order to efficiently and accurately synthesize affective audio from emotion-tagged sentences, we extensively tried and used several techniques out of which the selected ones are discussed below.

4.3.2.1 Hidden Markov Modelling (HMM)

A hidden Markov model (HMM) is a finite state machine which generates a sequence of discrete time observations. At each time unit, the HMM changes states at Markov process in accordance with a state transition probability, and then generates observational data **o** in accordance with an output probability distribution of the current state. We have used HTS [11], a tool to create voice synthesis module using HMM technique. In our case, the state transition probabilities depend upon the change of phonemes for the given word. We have used *festival* and *MaryTTS* speech building modules for synthesizing the voice after training it with the arctic dataset provided by Carnegie Mellon University (CMU)[14].

4.3.2.2 Digital Signal Processing (DSP)

HMM module generates speech in neutral affect, so in order to generate affective waveform from the neutral audio, we require signal processing modules that are provided by *MaryTTS* signal processing libraries[13] which are open source. These libraries include the latest research available in the

field of affective audio generation and it is continuously updated with the state-of-the-art techniques in DSP.

4.4 Outcomes

We define outcomes as the changes, benefits, learning or other effects that happen as a result of our work. The break-through achieved by our project is that it is the first end-to-end system which takes English text as input and produces affective output. Our work is multi-disciplinary in the sense that it extensively includes concepts and techniques from domains such as Machine Learning, Deep Learning, Natural Language Processing, Digital Signal Processing and High Performance Computing to name a few.

4.5 Applications

These days, the publisher not only publishes a printed version of the book but also releases an audio book. These audio books are generated by hiring voice actors in the recording studio where they read the entire book. Our project can work as an effective replacement for the process of audio book generation thereby saving both time and money of the publishers.

There are many PDF readers available which lack the emotional speech output producing capability for the novels and story books. These PDF readers can integrate our project by using an Application Programming Interface (API) and can effectively produce affective speech output.

Robots, these days, are now capable of speaking sentences in different languages. But their output is monotonous and devoid of any emotion. Our project's API can be used in the robots to eradicate this problem and thus, can help to make robots sound more human-like.

4.6 Hardware Resources Required

Our project requires following hardware resources to run smoothly and successfully

Sr. No.	Parameter	Min. Requirement	Justification
1	CPU Speed	1.0 GHz	Fast processor required for fast proceesing
2	RAM	4 GB	LSTM, Naive Bayes and NRC Lexicon model loading
3	Audio output device	-	To listen to output audio

Table 4.1: Hardware Requirements

4.7 Software Resources Required

Our project requires following software resources to run smoothly and successfully

Sr. No.	Parameter	Suggested	Justification
1	Any Linux distribution	Ubuntu 16.04	Easy dependency installation
2	Python	Python 2.7	Stable and large community support against python 3.0
3	MaryTTS	MaryTTS 5.2	Advanced and stable signal processing functionalities
4	NLTK	NLTK 3.2.2	Provides English processing tools
5	Scikit-learn	Scikit-learn 0.18.1	Provides machine learning algorithms

Table 4.2: Software Requirements

CHAPTER 5
PROJECT PLAN

5.1 Project Estimates

5.1.1 Reconciled Estimates

5.1.1.1 Cost Estimate

Our project falls in organic class of software product.

For organic:
Effort = 2.4 * $(KLOC)^{1.05}$
=4.969271 PM

Where,
KLOC is the estimated size of the software expressed in kilo
lines of code

Duration = 2.5 * $(Effort)^{0.38}$
= 4.5976 Months

5.1.1.2 Time Estimates

Learning about the tools, technologies and theory associated with our project which includes literature survey, Machine Learning, Deep Learning and Linguistic characteristics of Human speech took about 5 months.

Time required to implement the software system was about 2 months.

5.1.2 Project Resources

We have identified following three types of resources required for the successfull completion of our project within the stipulated time period.

5.1.2.1 Human Resources

1. Internal Guide: Dr. Girish P. Potdar

2. Group members:

    i Pranjal Bhor
    ii Vaibhav Chaudhari
    iii Prathamesh Dharangutte
    iv Murtaza Raja

5.1.2.2 Hardware Resources

The following hardware resources were required during the incubation phase
of our project

Sr. No.	Parameter	Required	Justification
1	CPU Speed	2.0 GHz	Faster computation for training classifier
2	RAM	8 GB	LSTM and Naive Bayes matrices temporary storage
3	Nvidia GPU	GTX 860m	LSTM is neural network which requires GPU power to get trained
4	Audio output device	-	To test the audio output

Table 5.1: Hardware Requirements during incubation

5.1.2.3 Software Resources

Following are the software resources required by our project while in its incubation phase

Sr. No.	Parameter	Suggested	Justification
1	Any Linux distribution	Ubuntu 16.04	Easy dependency installation
2	Python	Python 2.7	Stable and large community support against python 3.0
3	MaryTTS	MaryTTS 5.2	Advanced and stable signal processing functionalities
4	NLTK	NLTK 3.2.2	Provides English processing tools
5	Scikit-learn	Scikit-learn 0.18.1	Provides machine learning algorithms
6	Nvidia CUDA	CUDA 8.0	Provides drivers for running code on Nvidia GPU
7	CudNN	CudNN v5.0	Required to run Neural Network training algorithms on Nvidia GPU

Table 5.2: Software Requirements during incubation

5.2 Risk Management w.r.t. NP Hard analysis

5.2.1 Risk Identification

For risks identification, review of scope document, requirements specifications and schedule is done. Answers to questionnaire revealed some risks. Each risk is categorized as per the categories mentioned in Pressman. Please refer table 5.3 for all the risks

5.2.2 Risk Analysis

The risks for the Project can be analyzed within the constraints of time and quality

ID	Risk Description	Probability	Impact		
			Schedule	Quality	Overall
1	Risk 1	Medium	Low	Medium	Medium
2	Risk 2	Low	Low	Medium	Medium
2	Risk 3	Medium	Medium	Medium	Medium

Table 5.3: Risk Table

Probability	Value	Description
High	Probability of occurrence is	$> 75\%$
Medium	Probability of occurrence is	$26 - 75\%$
Low	Probability of occurrence is	$< 25\%$

Table 5.4: Risk Probability definitions

Impact	Value	Description
Very high	$> 10\%$	Schedule impact or Unacceptable quality
High	$5 - 10\%$	Schedule impact or Some parts of the project have low quality
Medium	$< 5\%$	Schedule impact or Barely noticeable degradation in quality Low Impact on schedule or Quality can be incorporated

Table 5.5: Risk Impact definitions

5.2.3 Overview of Risk Mitigation, Monitoring, Management

Following are the details for each risk.

Risk ID	1
Risk Description	If some of the words in the sentence are unknown to the classifier then the emotion of the sentence may not be identified correctly.
Category	Post-deployment
Source	Input text
Probability	Medium
Impact	Medium
Response	Mitigate
Strategy	Give only one emotion for the entire sentence
Risk Status	Identified

Risk ID	2
Risk Description	If all the words in the sentence are unknown to the classifier, then the sentence cannot be labelled.
Category	Post-deployment
Source	Input text
Probability	Low
Impact	Medium
Response	Mitigate
Strategy	Making classifier using larger dataset
Risk Status	Identified

Risk ID	3
Risk Description	A sentence might contain more that one emotion which might make it difficult to process its audio.
Category	Development
Source	Classifier label
Probability	Medium
Impact	Medium
Response	Accept
Strategy	Processing audio using available technique
Risk Status	Identified

5.3 Project Schedule

5.3.1 Project task set

Task 1: Learning and understanding the basics of machine learning and digital signal processing techniques.

Task 2: Generating a large dataset for emotion identification.

Task 3: Implementing and comparing various techniques available for speech processing and selecting the suitable one.

Task 4: Creating a classifier that classifies the emotions in english text with reasonable accuracy.

Task 5: Creating a speech signal processing module.

Task 6: Combining speech signal processing module and classifier module to complete the system.

Task 7: Testing and fixing of bugs in the system.

Task 8: Documentation and open source release of the project.

5.3.2 Task network
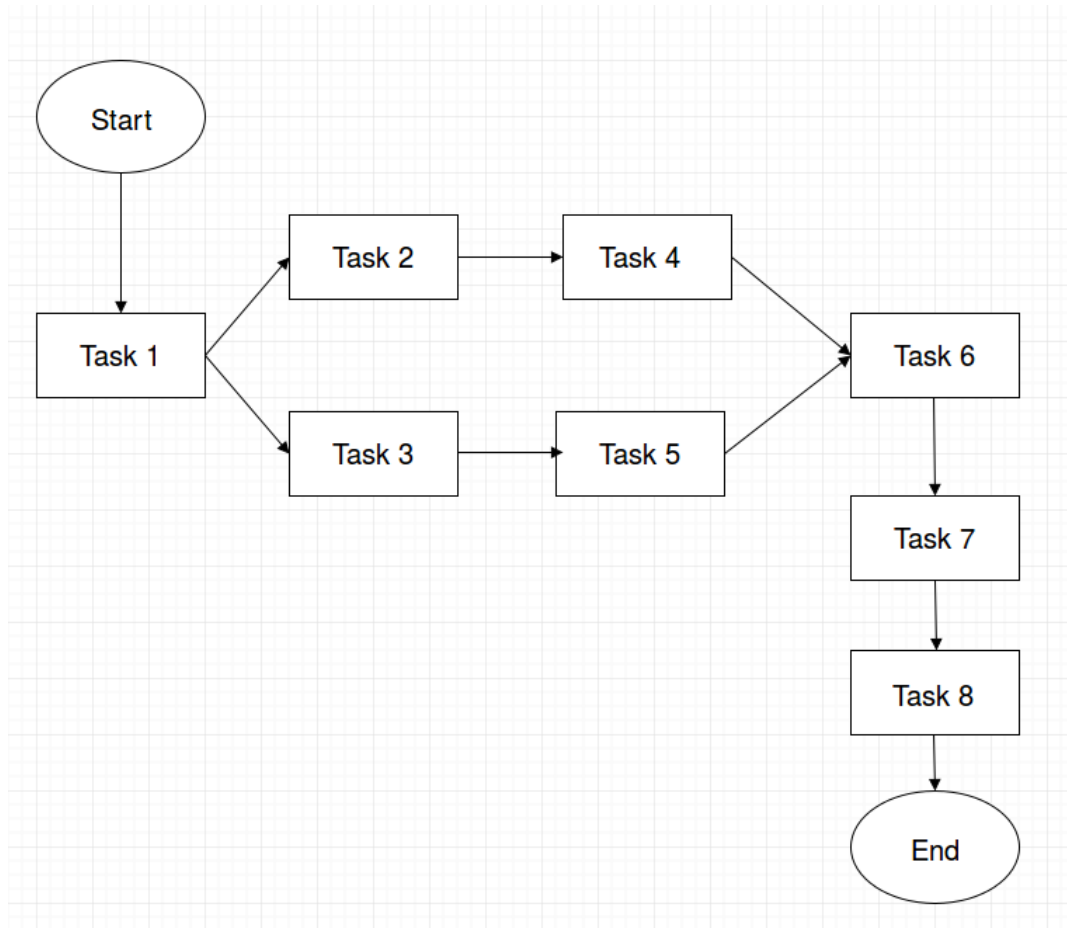
5.3.3 Timeline Chart
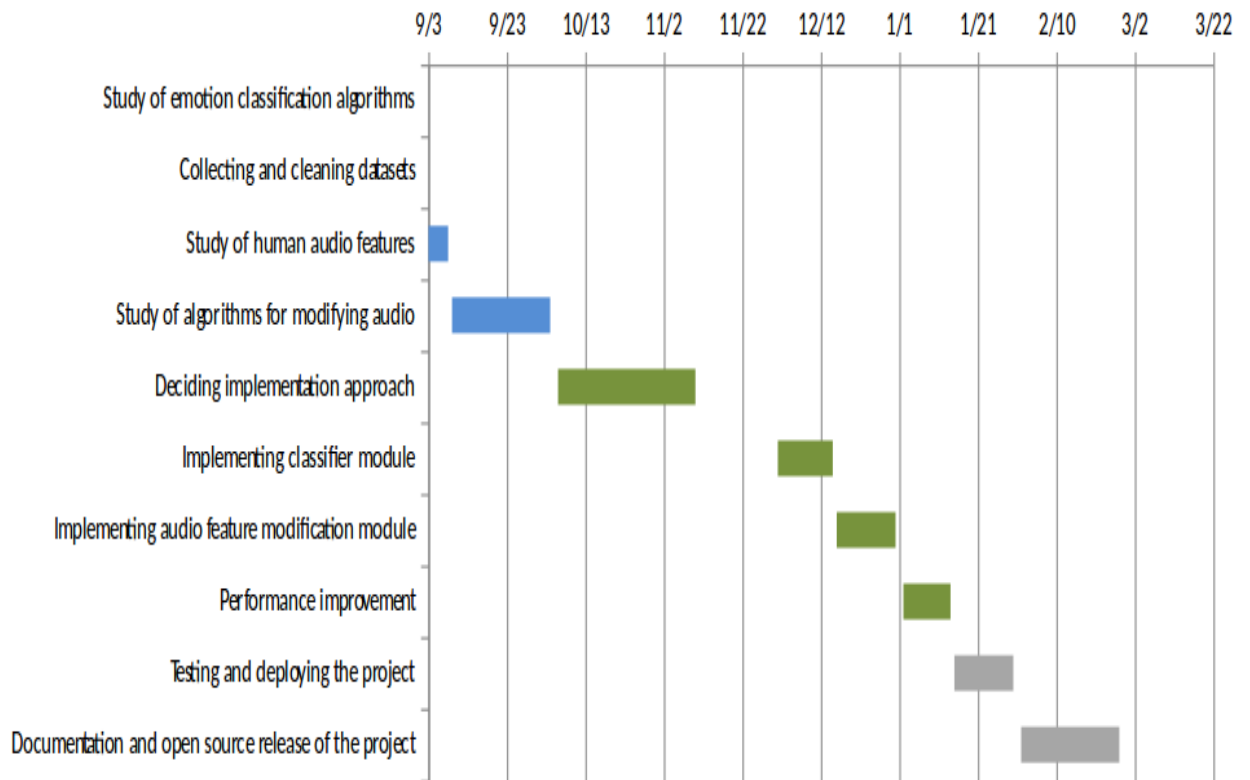
5.4 Team Organization

5.4.1 Team structure

The team structure for the project is identified. Roles are defined.

Pranjal Bhor: Multi-threaded execution of the system and NLP emotion classifier developer.

Vaibhav Chaudhari: Speech Synthesis System and signal processing developer.

Prathamesh Dharangutte: Developer of Machine Learning based emotion classifier.

Murtaza Raja: UI and NLP based emotion classifier developer.

5.4.2 Management reporting and communication

E-mail
It was used to communicate structural and design changes in the software to be developed.

Whatsapp
It was used to communicate in-short the work progress and status reporting

Verbal communication
It was used for extensive discussion about the strategy, implementation and future to-do things about the project.

CHAPTER 6

SOFTWARE REQUIREMENT SPECIFICATION

6.1 Introduction

6.1.1 Purpose and Scope of Document

The purpose of this document is to present a detailed description of emotion labelling and speech signal processing mechanisms with a text-to-speech engine. It will explain the purpose and features of the system, the interfaces of the system, what the system will do, and how the system will react to external stimuli. This document is intended for both the users and the developers of the system. This software system is designed to enhance the emotional aspect of the neutral text-to-speech engine. The system can be used by the PDF readers, audio books generating studios and video games companies to automate the reading of the english text.

6.1.2 Overview of responsibilities of Developer

There are certain responsibilities which have to be taken care of by the developers of this project. First and foremost is to accurately define the scope and goals of the project. According to the scope, project design is made. Project plan of development is made. Each developer holds certain responsibility of specific module developement. The module development should contain novel approach to the problem encountered. Different types of testing are to be carried out on the final product made.

6.2 Usage Scenario

This section provides various usage scenarios for the system to be developed.

6.2.1 5.2.1 User profiles

User: User is the actor who uses the system. That is, user gives input to the system and gets the system's output.

Developer: Developer is the actor who is responsible for designing, developing and maintaining the software system.

Researcher: Researcher is responsible for formulating new features for the system, which are not currently available.

Sr No.	Use Case	Description	Actors	Assumptions
1	English text input	The input file contains only the english text	User	The input is english text file
1	Non english text input	The input file contains non-english sentences	User	The input file contains english text

Table 6.1: Use Cases

6.2.2 Use-cases

6.2.3 Use Case View

Use Case Diagram. Example is given below
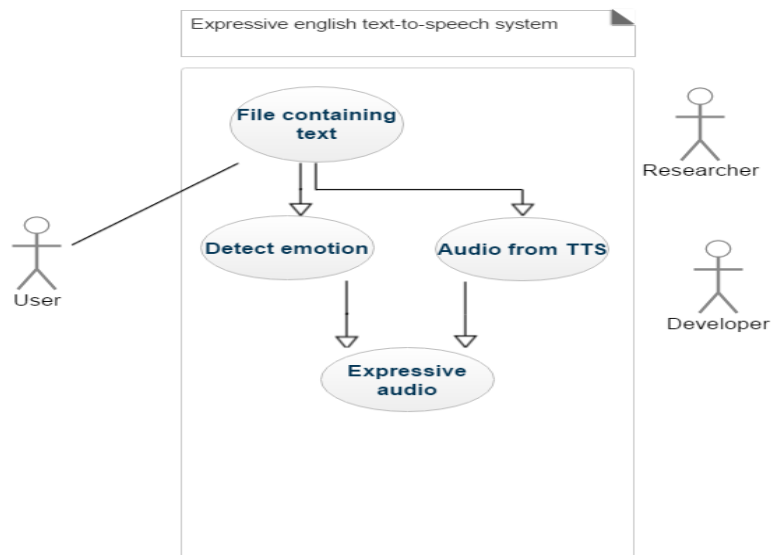
Figure 6.1: Use case diagram

6.3   Data Model and Description

6.3.1   Data Description

There are multiple types of data in the system. The machine learning classifier is a trained model which contains all the data on which it was trained along with its relationships with the output. The input data to the system consists of English text or PDF file. This data is to be loaded in memory in-order to work on it. Each and every sentence from the input file is to be classified into one of the 5 emotions. This also requires a data structure. The output of the system is audio data.

6.3.2   Data objects and Relationships

The input to the system is English text or PDF file. The data that will be manipulated by the system is Neutral audio speech.

6.4   Functional Model and Description

6.4.1   Data Flow Diagram

6.4.1.1   Level 0 Data Flow Diagram
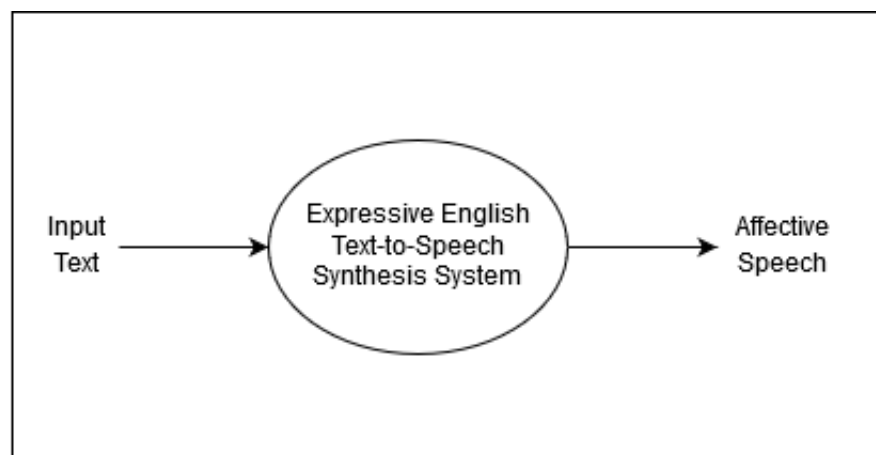
Figure 6.2: Level 0 Data flow diagram

6.4.1.2   Level 1 Data Flow Diagram
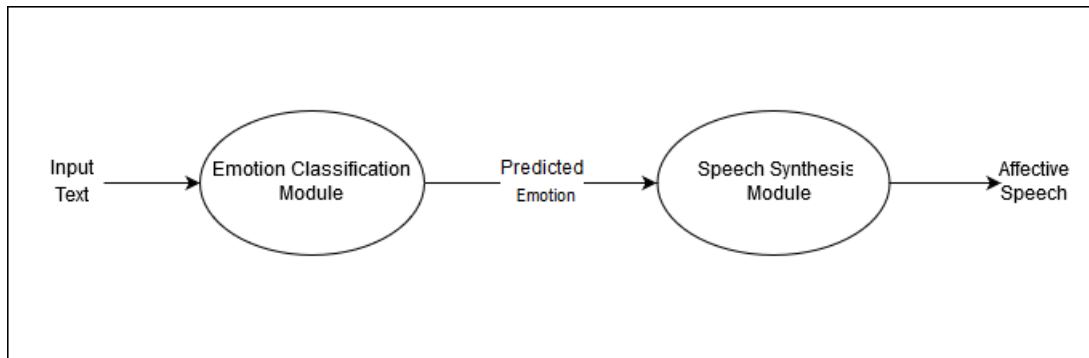
Figure 6.3: Level 1 Data flow diagram

6.4.1.3   Level 2 Data Flow Diagrams
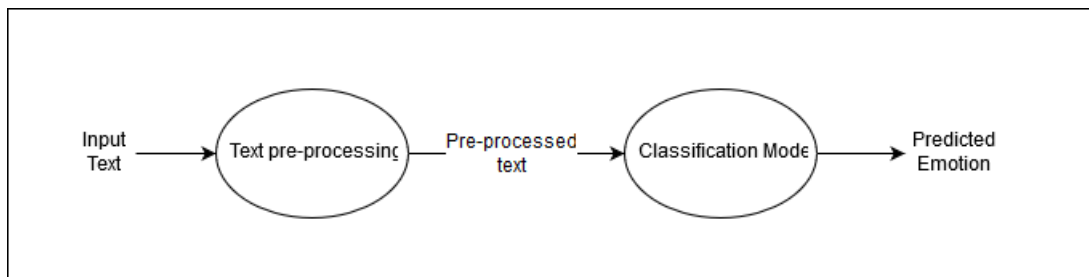
Figure 6.4: Level 2 Data flow diagram - Emotion Classification
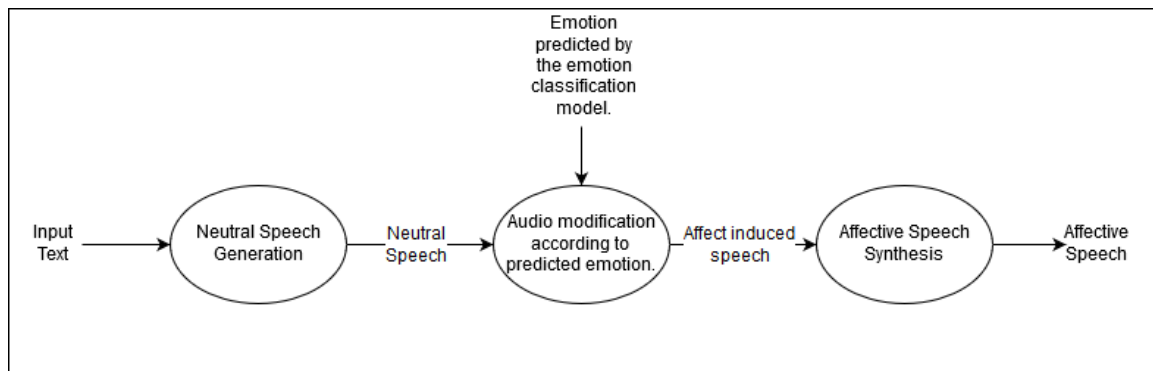
Figure 6.5: Level 2 Data flow diagram - Speech Synthesis

6.4.1.4 Level 3 Data Flow Diagrams
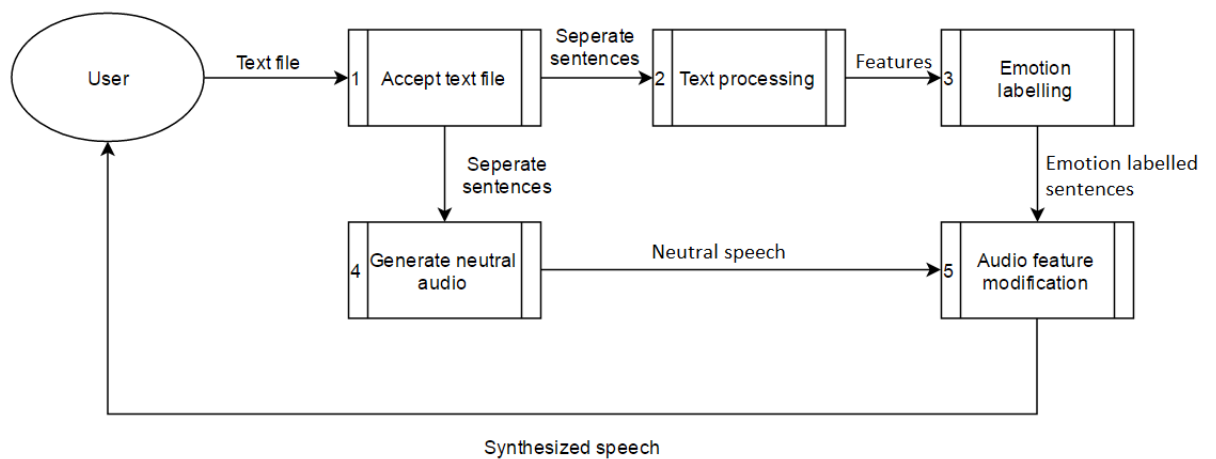
Figure 6.6: Level 3 Data flow diagram

6.4.2   Activity Diagram:
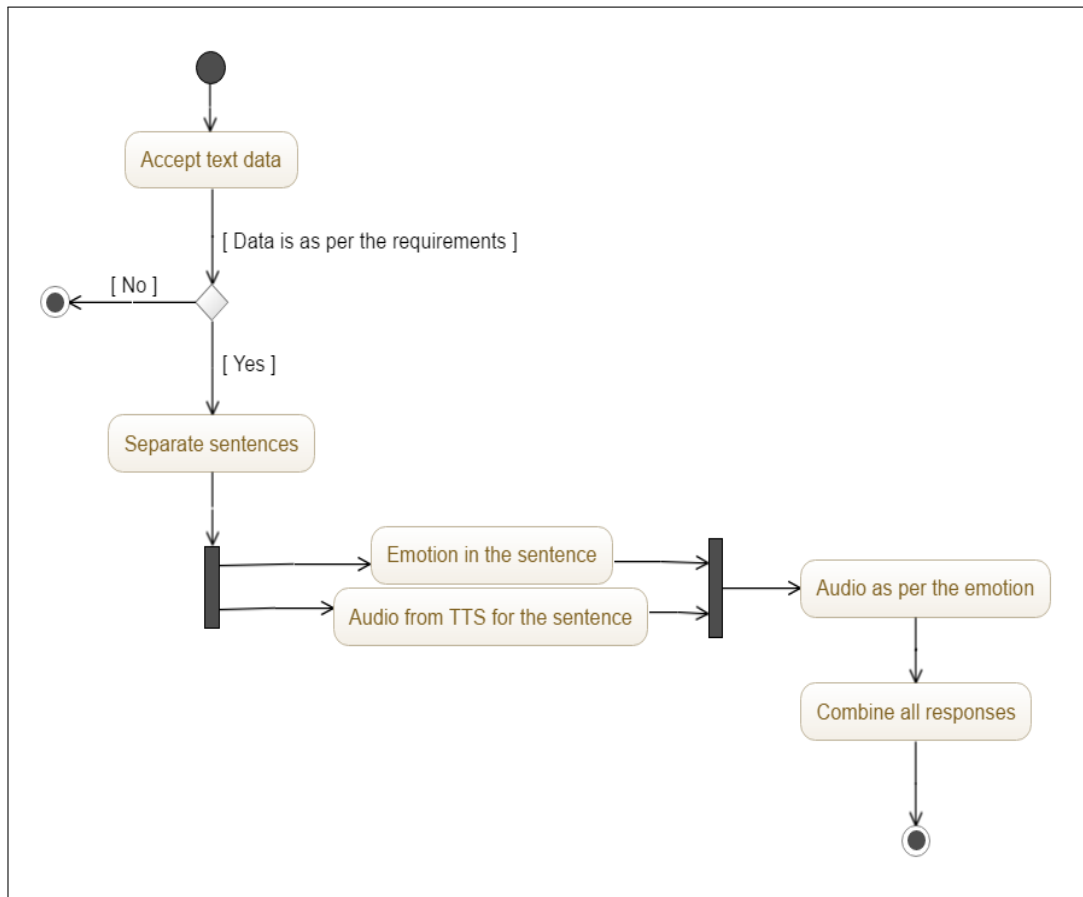
Figure 6.7: Activity diagram

6.4.3   Non Functional Requirements:

Interface Requirements

The Graphical User Interface of the project should contain facility to select an English text or PDF file. It should also contain a mechanism to display the emotions of each and every sentence in the text file. There should be a media player type mechanism present in the system which the user can use to listen and seek the audio in-window. It can also contain facility to pause and stop the audio playing.

Performance Requirements

The system should be as fast as possible. The file loading and emotion
detection phase should be extremely fast. Each sentence can be considered
independently and so can processed through multi-threading. Similar goes
for the audio genenration. Every sentence's audio can be generated indepen-
dently of other sentences.

Software quality attributes

The software should be available all the time whenever user needs it. For a
standalone software, it should be able to run whenever invoked. A web based
application should have up-time of more than 99%. The software should be
reliable and should not give erratic or unwanted behaviour. It should be as
modular as possible so that modifiability of the software can be improved.

6.4.4   Design Constraints

There are no design constraints which will impact the subsystem. The soft-
ware contains simple GUI as mentioned above and implementing that con-
straints the project developers in no significant way.

6.4.5   Software Interface Description

The software interface contains a mechanism through which an user will be
able to load a file and can take an audio file of the input as the output. In the
future, an Application Programming Interface (API) can be provided for the
developers' use which will directly communicate with the comercially hosted
servers.

6.4.6   State Diagram:
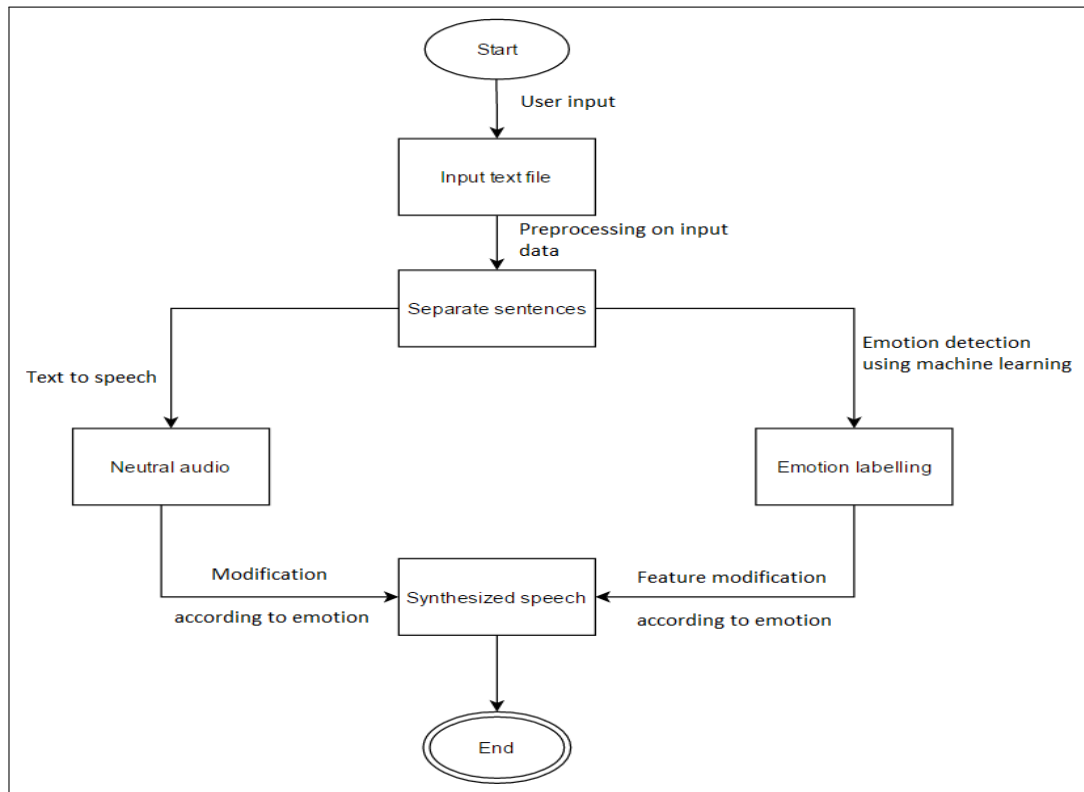
Figure 6.8: State transition diagram

CHAPTER 7

DETAILED DESIGN DOCUMENT USING APPENDIX A AND B

7.1 Introduction

Architechture design gives an overview of the software system. The architechture consists of and Emotion Detection and Signal Processing (EDSP) engine. The EDSP engine consists of emotion classifier which includes Navie Bayes and Long Short Term Memory (LSTM) classifiers. It also includes Text-to-Speech generation engine and signal processing engine which modifies the neutral voice output into affective voice output.

7.2 Architectural Design
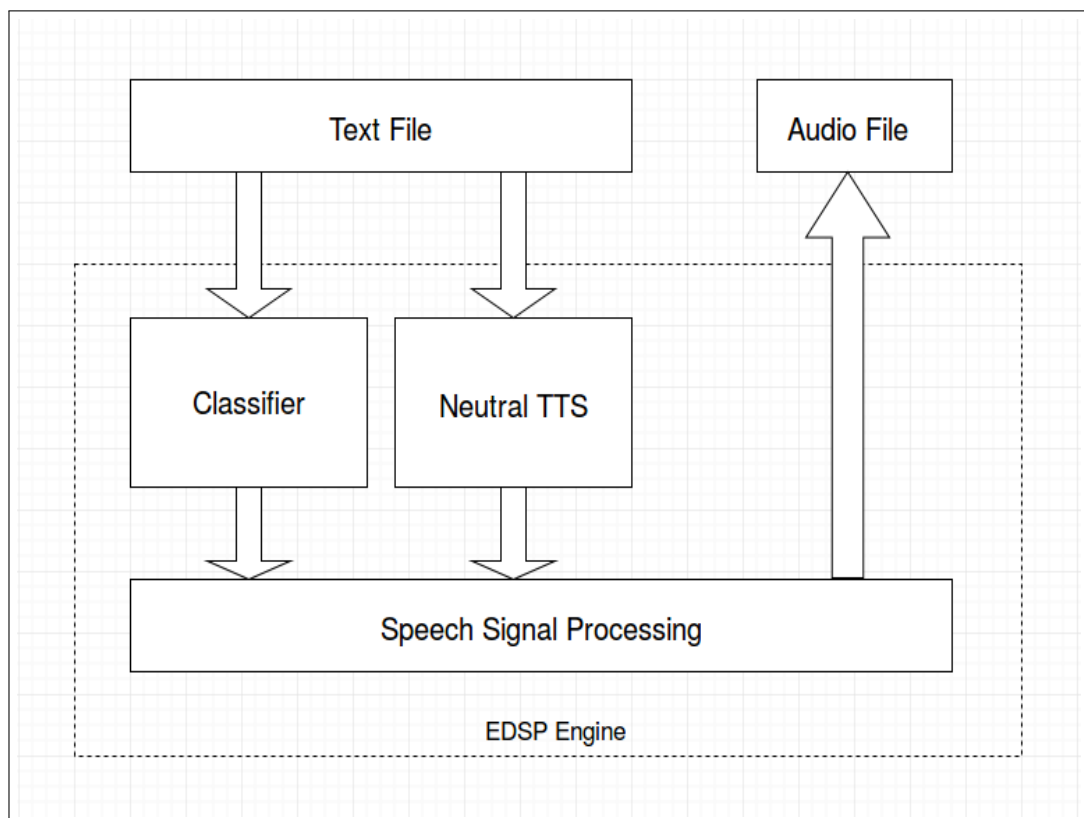
Figure 7.1: Architecture diagram

7.3 Data design (using Appendices A and B)

The internal and global software data structures have been presented below.

7.3.1 Internal software data structure

Multi-dimensional arrays

This data structure is used to store the machine learning classifier data. The classifier learns the data and maps it with the output. This mapping is stored in multi-dimensional arrays.

Dictionary

The input file contains many sentences. The emotions of these sentences are found out and they are stored in a dictionary where sentence is the key and its emotion is the value. Tree

The system uses Stanford Parser to find out which phrases are qualified by specific words. The parser generates a parse tree out of the input English sentence.

7.3.2 Global data structure

There is no global data structure in the system. All the data sturctures are used internally by respective components.

7.3.3 Database description

Input data is stored in plain English text files or PDF files. Internally there is no use of database for handling of any types of tasks. The audio output is stored in .wav file format.

7.4 Component Design

7.4.1 Class Diagram
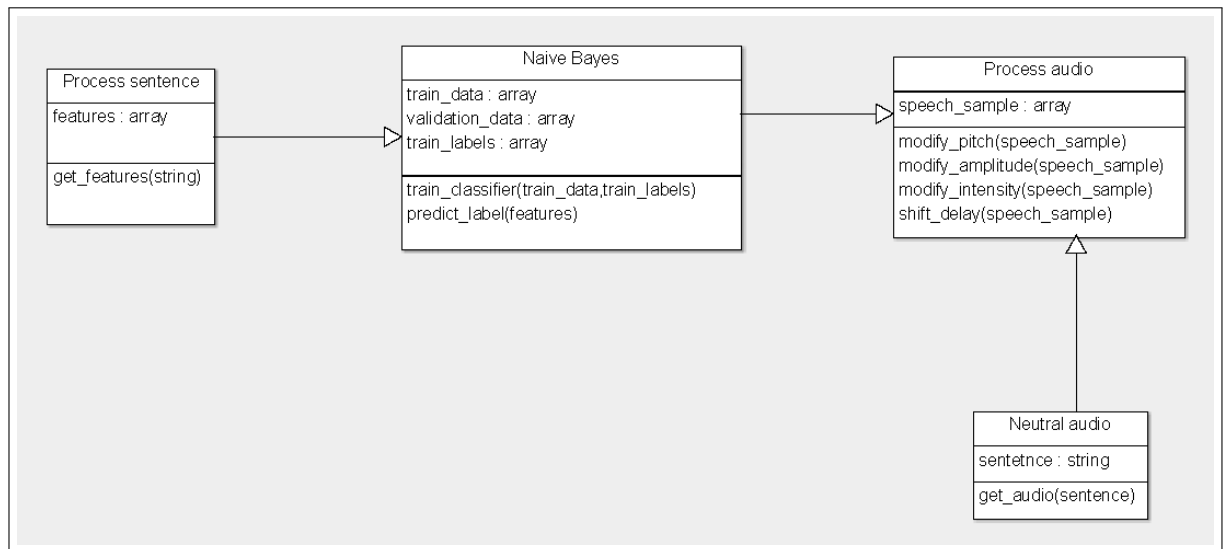
Figure 7.2: Class Diagram

CHAPTER 8
PROJECT IMPLEMENTATION

8.1 Introduction

Our system consists of a simple GUI wherein user has to load his text/PDF file containing English sentences. As soon as the file is loaded, each sentence from the file gets tagged with its most probable emotion. After clicking on speak button, affective audio for the sentences is generated and stored as a .wav file which is automatically played in the GUI. The user can control the audio file playing using a small media player provided.

8.2 Tools and Technologies Used

Python 2.7 Programming language

Python provides excellent libraries to automate most of the trivial work for system design. The members of the project group were already experienced in Python 2.7 and that is why we chose this as primary language for system implementation.

MaryTTS signal processing engine

MaryTTS is a research based Text-to-speech engine which supports many different languages including English. It has its own signal processing engine which modifies the neutral audio according to the input parameters given. This engine helps us to get emotional audio by modifying neutral audio.

Festival

It is a Hidden Markov Modelling (HMM) tool for building synthetic voice. As HMM is another form of machine learning, it requires training data in the form of voice audio and its text. As we want to have our own text-to-speech engine, we had to build our own voice which is why festival is used.

NLTK and SCIKIT libraries

NLTK contains many functions which carry out trivial tasks efficiently like stop words removal and tokenizing the words. It is also used to traverse on the parse tree generated by Stanford Parser. SCIKIT provides machine learning algorithms. We use the implementation of machine learning algorithms provided by SCIKIT python.

8.3 Methodologies/Algorithm Details

8.3.1 Algorithms

8.3.1.1 Overall System Algorithm

Algorithm 1 Overall system algorithm

1: **procedure** GENERATE_AUDIO(sentence)
2:     naive_emotion ← GET_EMOTION_FROM_NAIVE(sentence)
3:     lstm_emotion ← GET_EMOTION_FROM_LSTM(sentence)
4:     nlp_emotion ← GET_EMOTION_FROM_NLP(sentence)
5:     final_emotion ← MOST_VOTED_EMOTION(naive emotion,lstm emotion,nlp emotion)
6:     neutral_audio ← HMM_AUDIO_SYNTHESIS(sentence)
7:     affective audio ← SIGNALPROC(sentence,emotion)
8:     return save audio(affective audio)
9: **end procedure**

1: **procedure** SENTENCE_HANDLER(sentence)
2:     final_audio = ϕ
3:     **for** each sentence in text **do**
4:         temp_audio ← GENERATE_AUDIO(sentence)
5:         final_audio ← final_audio $\bigcup$ temp_audio
6:     **end for**
7: **end procedure**

8.3.1.2 Naive Baye's Classification Algorithm

Algorithm 2 Naive Baye's Classification Algorithm

1: **procedure** GET_EMOTION_FROM_NAIVE(sentence)
2:     classifier ← load_naive_classifier()
3:     emotion ← classifier.classify(sentence)
4:     return emotion
5: **end procedure**

8.3.1.3   Get emotion from LSTM

Algorithm 3 Get emotion from LSTM

1: **procedure** GET_EMOTION_FROM_LSTM(sentence)
2:     classifier ← load_lstm_classifier()
3:     emotion ← classifier.classify(sentence)
4:     return emotion
5: **end procedure**

8.3.1.4   HMM Audio Synthesis

Algorithm 4 HMM Audio Synthesis Algorithm

1: **procedure** HMM_AUDIO_SYNTHESIS(sentence)
2:     linguistic_specification = get_linguistic_specification(sentence)
3:     final_hmm_sequence = ϕ
4:     **for all** representation ∈ linguistic_specification **do**
5:         hmm_sequence ← generate_hmm(representation)
6:         final_hmm_sequence ← final_hmm_sequence ∪ hmm_sequence
7:     **end for**
8:     voice_output ← generate_voice_from_hmm(final_hmm)
9:     return voice
10: **end procedure**

8.3.1.5   Get emotion from NLP module

Algorithm 5 Get emotion from NLP module

1: **procedure** GET_EMOTION_FROM_NLP(sentence)
2:     split sentence on spaces and store in word_list
3:     word_list ← lemmatize(word_list)
4:     word_dictionary ← ϕ
5:     **for** each word in word_list **do**
6:         word_emotion ← NRCclassifier(word)
7:         word_dictionary{word} ← word_emotion
8:     **end for**

Algorithm 5 Get emotion from NLP module (continued)

9:      emotion_dictionary ← {'neutral':0,'joy':0,'sadness':0,'surprise':0,'anger':0}

10:     dependency_tree = StanfordParser(sentence)

11:     get_index ← 0

12:     Indices ← ϕ

13:     **for** each word in word_dictionary **do**

14:         emotion_dictionary[word_dictionary[word]]+= 20

15:         get_index ← find_max_qualified_word(dependency_tree)

16:         **for** $i = 0$ and $i <$ length(dependency_tree) **do**

17:            **if** dependency_tree[i].description is 'amod' or 'advmod' **then**

18:               **if** dependency_tree[i].head = get_index **then**

19:                  **if** dependency_tree[i].form in strong_qual_words **then**

20:                     emotion_dictionary[word_dictionary[word]]+= 100

21:                  **else if** dependency_tree[i].form in avg_qual_words **then**

22:                     emotion_dictionary[word_dictionary[word]]+= 50

23:                  **end if**

24:               **end if**

25:            **end if**

26:         **end for**

27:         Indices ← Indices $\bigcup$ get_index

28:     **end for**

29:     answer_emotion ← get_max_valued_emotion(emotion_dictionary)

30:     **if** negation present in the sentence **then**

31:         **if** the negation word qualifies an index in Indices **then**

32:            **if** max_emotion_value $>$ 20 **then**

33:               answer_emotion ← 'neutral'

34:            **else if** max_emotion_value == 20 **then**

35:               **if** answer_emotion == 'joy' **then**

36:                  answer_emotion ← 'sadness'

37:               **else if** answer_emotion == 'sadness' **then**

38:                  answer_emotion ← 'joy'

39:               **else**

40:                  answer_emotion ← 'neutral'

41:               **end if**

42:            **end if**

43:         **end if**

44:     **end if**

45:     return emotion

46: **end procedure**

8.4 Verification and Validation for Acceptance

Many different sentences denoting various emotions like joy, sadness, surprise, angry and neutral were given to the system. The system predicted the emotions and the people who tested the system agreed on the satisfactory level of the output. The corresponding audio generated was also approved by the users who tested the system. This provided verification as well as validation for the project and it got acceptance.

CHAPTER 9

SOFTWARE TESTING

9.1 Type of Testing Used

Since our project focuses mainly on two important tasks Classification of sentences based on emotions using machine learning (A) and conversion of monotonic speech into expressive speech (B), we have analyzed different testing strategies based on the aforementioned tasks.

9.1.1 Unit testing

This testing was very valuable for our project since we were able to independently test the emotion identification module and the speech synthesis module with good results. We essentially decoupled the two subsystems and tested them.

9.1.1.1 Classification module

This subsystem was tested by providing some sentences rich with five emotions - joy, sadness, anger, surprise and neutral. With an accuracy of about 80%, it produced the results accordingly. It labelled more sentences as neutral because the database was biased toward it.

9.1.1.2 Speech synthesis module

For voice synthesis and digital signal processing unit, same sentences were given as input. The affective audio generated by the unit was perceived to contain the emotion tagged as reported by the test users. The sentences which depicted anger as the emotion, were spoken a bit fast by the system.

9.1.2 Integration testing

Having tested the individual subsystems, the emotion prediction and voice synthesis modules were then integrated. A sentence was passed to the input function of the system. The emotion classifier correctly identified the emotion tag of the sentence and then using this emotion tag, the affective audio was generated.

9.1.2.1 Multithreading fix

It was discovered during testing that a single MaryTTS engine does not handle simultaneous requests and is not thread safe. Therefore, we spawned multiple instances of the engine and then fired simultaneous requests to them.

9.1.3 Validation testing

Since the ideas presented through this project are still in research phase, a usable product is a feat that cannot be achieved provided the current technology in this field. Still, we tried to test this software, targeting the audio book publishers as our client.

9.1.3.1 Emotional speech as output

This software provides the affective audio for emotions - joy, sadness, anger, surprise and neutral for the input provided by the user.

9.1.3.2 Input characteristics

Since majority of text is either available as .txt or as .pdf, this software has the ability to process both of them.

9.1.4 Performance and load testing

We used load testing to test the system under various loads. The workstation on which testing was performed consisted of a 2.5 GHz quad-core machine with 8 GB of main memory running Ubuntu operating system.

9.1.5 GUI testing

We made a very simple GUI in the beginning but soon found out that it is not user friendly and does not have a native-look-and-feel. Then we made another GUI which solved the problems of the previous one and offered better functionality in terms of audio playing and emotion selection.

9.1.5.1 Emotion selection

A drop down menu was associated with each and every sentence that represented the emotion of the sentence. The user has the ability to change this according to his or her preference.

9.1.5.2 Audio playback

A media player like functionality was added to the GUI so that the user can play, pause, stop, seek and change the volume of the audio being played.

9.1.5.3 Scrollable interface

If the number of sentences exceeded the minimum which can fit on the screen, the windows automatically provides scroll bars that can be used to navigate between the sentences.

9.2 Test Cases and Results

9.2.1 Unit testing

9.2.1.1 Classifier module

No.	Input	Expected Output	Actual Output
1.	I am very delighted	joy	joy
2.	My dog died	sadness	sadness
3.	Oh my God, I got that!	surprise	surprise
4.	I am very angry at that behaviour	anger	anger
5.	Those were tough times for me	neutral	neutral
6.	I am not happy	sadness	sadness
7.	I am not very sad	neutral	neutral

Table 9.1: Test cases for classifier module

9.2.1.2   Speech synthesis module

No.	Input emotion	Expected Output	Actual Output
1.	joy	speaks in an overall pleasant way	as expected
2.	sadness	the pitch drops and tempo decreases	as expected
3.	surprise	the pitch increases, some words stressed	as expected
4.	anger	speaks a bit fast	speaks very fast
5.	neutral	speaks normally	as expected

Table 9.2: Test cases for speech synthesis module

9.2.2   Integration testing

No.	Input	Expected Output	Actual Output
1.	affective sentence	emotional speech generated	as expected
2.	neutral sentence	neutral speech generated	as expected
3.	text file with various sentences	proper affective audio generated corresponding to emotions	as expected
4.	pdf file with various sentences	proper affective audio generated corresponding to emotions	as expected

Table 9.3: Test cases for integration testing

9.2.3 Validation testing

No.	Feature	Implemented	Details
1.	produces affective voice on emotional text input	yes	can process 5 emotions
2.	handles text files	yes	can process .txt and .pdf
3.	automatically tags sentences with emotions	yes	currently 80% accurate
4.	support for other languages	no	future scope, data set limitations
5.	support for other accents	yes	training data to be provided

Table 9.4: Test cases for validation testing

9.2.4 Performance and load testing

No.	Input	Time taken to execute
1.	5-10 sentences.	2-3 seconds.
2.	1 page.	About 15 seconds.
3.	10-15 pages.	About 3 minutes.
4.	A novel of about 300 pages.	About 1 hour 5 minutes.
5.	3-4 instances of our software running at the same time.	Sometimes system gets hung up.

Table 9.5: Test cases for load testing

9.2.5 GUI testing

No.	Case	Expected Output	Actual Output
1.	loading files	.txt and .pdf files loaded	as expected
2.	changing emotion of a sentence	drop down updated	as expected
3.	sentences overflow the viewport	proper scroll bars are added accrdingly	as expected
4.	audio playing capability	play, pause, stop, seek, volume	as expected
5.	performing computational intensive task	show progress to user	no progress shown

Table 9.6: Test cases for GUI testing

CHAPTER 10
RESULTS

10.1 Screenshots

The welcome screen of our system looks as shown in Figure 10.1. It is a simple screen with some information about our project and the team members. It contains a only one button to load the English text file or PDF file.
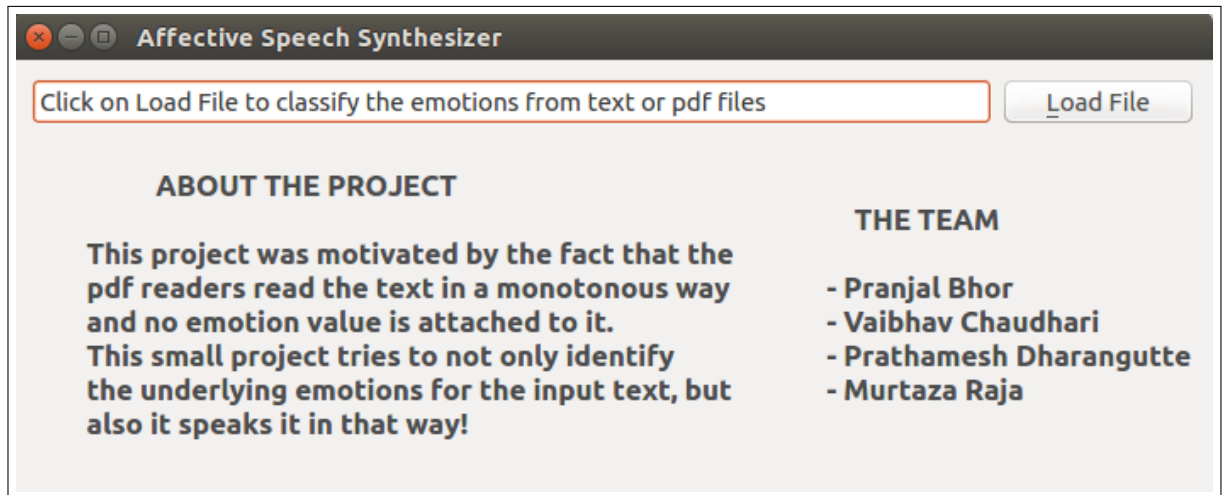
Figure 10.1: Intro screen

When the file has been loaded in the interface, the emotions from each sentence are predicted and displayed in the GUI. The interface expands itself and adds new components to handle the sentences and their emotions. The size of the GUI increases because of this. Also, there is a selectable dropdown against each sentence which can be used by the user to change the emotion of the sentece if he/she is not satisfied by what is given by our classifier. This GUI is shown in Figure 10.2.
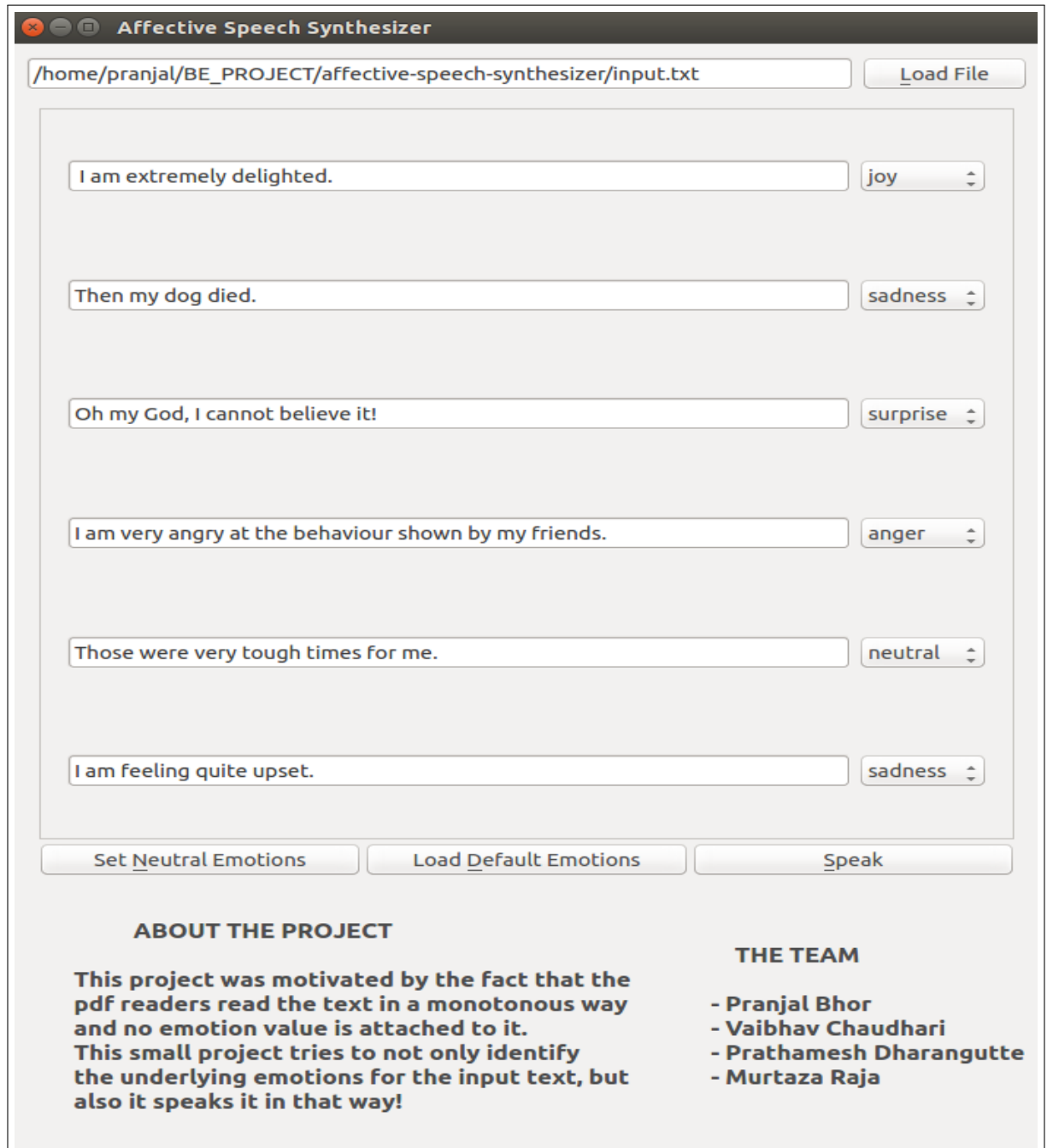
Figure 10.2: Emotion prediction screen

After the emotions are predicted from the sentences, the user clicks on speak button. Then the affective audio of each and every sentence is generated and is automatically played by the software. A new component of media player is loaded for that purpose. The user can pause and stop the audio, can increase or decrease the volume and can even seek the slider to whatever position he wants.
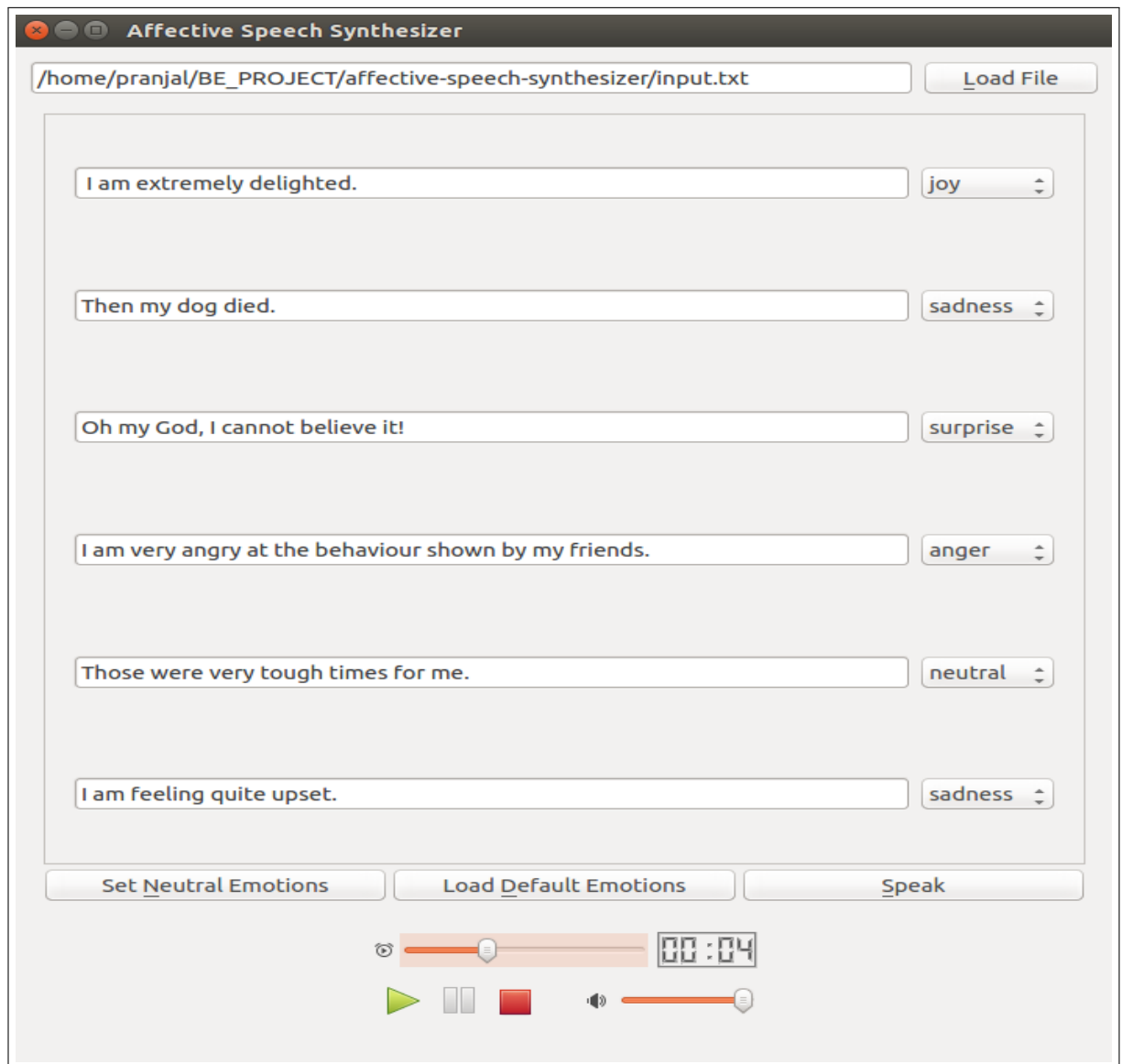
Figure 10.3: Affective audio playing screen

10.2 Accuracy of the Predictions

10.2.1 Theory of Precision, recall and F1-score

Precision (P) is defined as the number of true positives (T_p) over the number of true positives plus the number of false positives (F_p).

$$\mathbf{P} = \frac{\mathbf{T}_p}{\mathbf{T_p} + \mathbf{F_p}} \tag{10.1}$$

Recall (R) is defined as the number of true positives (T_p) over the number of true positives plus the number of false negatives (F_n).

$$\mathbf{R} = \frac{\mathbf{T}_p}{\mathbf{T_p} + \mathbf{F_n}} \tag{10.2}$$

These quantities are also related to the (F_1) score, which is defined as the harmonic mean of precision and recall.

$$\mathbf{F}_1 = 2 * \frac{\mathbf{P} * \mathbf{R}}{\mathbf{P} + \mathbf{R}} \tag{10.3}$$

The respective precision, recall and F1 scores of the classifiers used in our project are shown in the figures below.

10.2.2 Precision, recall and F1 score for Naive Bayes Classifier

```
The accuracy score is 77.76%
Predicted  anger  joy  neutral  sadness  surprise
Actual
anger        190   33      85       32         7
joy           19  218     148       19        10
neutral       27   34    1890       31         3
sadness       38   30      99      190         7
surprise       4   15      70        4        12
              precision    recall   f1-score   support

      anger       0.68      0.55       0.61        347
        joy       0.66      0.53       0.59        414
    neutral       0.82      0.95       0.88       1985
    sadness       0.69      0.52       0.59        364
   surprise       0.31      0.11       0.17        105

avg / total       0.76      0.78       0.76       3215
```

Figure 10.4: Precision, recall and F1 of Naive Bayes

10.2.3 Precision, recall and F1 score for LSTM classifier

```
The accuracy score is 82.08%
Predicted  anger  joy  neutral  sadness  surprise
Actual
anger        231   22      29       46        10
joy           24  262      33       34        21
neutral       48   82    1886       42        41
sadness       39   34      32      236         9
surprise       5   14       5        6        24
              precision    recall   f1-score   support

      anger       0.67      0.68       0.67        338
        joy       0.63      0.70       0.66        374
    neutral       0.95      0.90       0.92       2099
    sadness       0.65      0.67       0.66        350
   surprise       0.23      0.44       0.30         54

avg / total       0.84      0.82       0.83       3215
```

Figure 10.5: Pprecision, recall and F1 of LSTM

10.2.4 Accuracy score of NLP classifier

```
The accuracy score is 45.06%
Predicted  anger  joy  neutral  sadness  surprise
Actual
anger        128   41      138       27         6
joy           34  207      154       20         7
neutral      285  499     1023      171        31
sadness       81   66      113       85         4
surprise      20   25       36        9         6
```

Figure 10.6: Accuracy result of NLP classifier

CHAPTER 11

DEPLOYMENT AND MAINTENANCE

11.1 Installation and un-installation

The program is a runnable python script. It does not require installation like a traditional software. But some dependencies have to be installed to run the software. The steps are as follows:

1. Install project dependencies which includes: marytts-5.2, python 2.7, NLTK library, StanfordParser library, textract library, PyQT 4 and Phonon PyQT 4.

2. Download project source code

3. Execute 'run_mary_server.sh' script

4. Execute 'qtmain.py' script

As there is no traditional installation associated with the system. The following steps are enough to remove the project from the system:

1. Remove project dependencies which were installed in the installation steps.

2. Delete the project source code

CHAPTER 12

CONCLUSION AND FUTURE SCOPE

We developed a software that converts input text to its appropriate expressive speech. Two main modules namely emotion classifier and speech synthesis are present in our system. For emotion classification we worked with various techniques like Naive Bayes Classifier, SVM and LSTM. We observed that each technique has its ups and downs and there is no clear winner. Each technique is well suited for certain constructs of a sentence but may not work so well with other constructs. The accuracy of emotion classification is also an important aspect in this process. The accuracy of our Naive Bayes Classifier is about 78% and LSTM classifier is about 81%.One major requirement for the accurate prediction is the availability of a large, well formed and diverse dataset with high attribute accuracy. Improving the quality of the available dataset in the future and increasing the training examples will certainly improve the accuracy of emotion prediction.

Currently our system classifies the input text into only 5 emotions viz. joy,sadness,anger,surprise and neutral. But there are many subtleties in English language like sarcasm which cannot be correctly classified by present algorithms. The future work can include creating dataset and algorithms to identify more than above mentioned 5 emotions which can include fear, anger and disgust.

In our project we consider only one sentence at a time. The context of a paragraph or section is not taken into consideration. Also, the association or dependency of the speaker with the text is not considered. The future work can be done in this direction so as to identify the context of the paragraph and associativity of the speaker with the words he or she is speaking. This will certainly make the system Artificially Intelligent in audio generation.

At present our system only supports English language and only American accent due to the limitation of the audio dataset. In the future, dataset of Indian voice actors can be created and the Hidden Markov Model can be trained on it so as to create the voice with Indian accent. Similarly, audio can be recorded in any Indian regional language and its model can be trained so that the system can support languages other than English.

Furthermore, we generate expressive speech for joy,anger,sadness,surprise and neutral emotions but we do not consider their intensities. The future work can be directed in the direction of modifying neutral speech into expressive speech along with varying intensity depending on the context in which it is written.

One of the import limitations of our project includes that we consider only one domninant emotion in the sentence containing multiple emotions. An algorithm for the same can be developed in the future for synthesizing speech with multiple emotions such that all the emotions are perfectly incorporated in the audio.

The audio of the emotion anger produced by the system is not upto the mark of idenfication by a human listener. This is the limitation can be overcome in future by improving the anger audio synthesis parameters of MaryTTS engines.

CHAPTER 13
REFERENCES

ANNEXURE A
REFERENCES

[1] Vanmassenhove, Eva, Joo P. Cabral, and Fasih Haider. "Prediction of emotions from text using sentiment analysis for expressive speech synthesis." 9th ISCA Speech Synthesis Workshop, Sunnyvale, USA, Septemper. 2016.

[2] M. Schrder and J. Trouvain (2003). The German Text-to-Speech Synthesis System MARY: A Tool for Research, Development and Teaching. International Journal of Speech Technology, 6, pp. 365-377.

[3] Pierre-Yves, Oudeyer. "The production and recognition of emotions in speech: features and algorithms." International Journal of Human-Computer Studies 59.1 (2003): 157-183.

[4] Perikos, Isidoros, and Ioannis Hatzilygeroudis. "Recognizing emotions in text using ensemble of classifiers." Engineering Applications of Artificial Intelligence 51 (2016): 191-201.

[5] Perikos, Isidoros, and Ioannis Hatzilygeroudis. "Recognizing emotion presence in natural language sentences." International Conference on Engineering Applications of Neural Networks. Springer Berlin Heidelberg, 2013.

[6] Johnstone, Tom. The effect of emotion on voice production and speech acoustics. University of Western Australia, 2001.

[7] Alm, Cecilia Ovesdotter, Dan Roth, and Richard Sproat. "Emotions from text: machine learning for text-based emotion prediction." Proceedings of the conference on human language technology and empirical methods in natural language processing. Association for Computational Linguistics, 2005.

[8] Pang, Bo, and Lillian Lee. "Opinion mining and sentiment analysis." Foundations and trends in information retrieval 2.1-2 (2008): 1-135.

[9] Christiane Fellbaum, Ed. 1998. WordNet: An Electronic Lexical Database. MIT Press, Cambridge, Mass.

[10] Chaffar, Soumaya, and Diana Inkpen. "Using a heterogeneous dataset for emotion analysis in text." Canadian Conference on Artificial Intelligence. Springer Berlin Heidelberg, 2011.

[11] Yamagishi, Junichi. "An introduction to hmm-based speech synthesis." Technical Report (2006).

[12] Mohammad, Saif M., and Peter D. Turney. "Crowdsourcing a worde-motion association lexicon." Computational Intelligence 29.3 (2013): 436-465.

[13] Marc Schr oder, Marcela Charfuelan, Sathish Pammi, Ingmar Steiner. Open source voice cre- ation toolkit for the MARY TTS Platform. 12th Annual Conference of the International Speech Communication Association - Interspeech 2011, Aug 2011, Florence, Italy. ISCA, pp.3253-3256, 2011. <hal-00661061>

[14] Kominek, John, Alan W. Black, and Ver Ver. "CMU ARCTIC databases for speech synthesis." (2003).

[15] Danqi Chen and Christopher D Manning. 2014. A Fast and Accurate Dependency Parser using Neural Networks. Proceedings of EMNLP 2014

[15] Dan Klein and Christopher D. Manning. 2003. Accurate Unlexical-ized Parsing. Proceedings of the 41st Meeting of the Association for Computational Linguistics, pp. 423-430.

[16] Schmid, Helmut. "Probabilistic part-ospeech tagging using decision trees." New methods in language processing. Routledge, 2013.

ANNEXURE B

LABORATORY ASSIGNMENTS ON PROJECT ANALYSIS OF ALGORITHMIC DESIGN

Assignment

To develop the problem under consideration and justify feasibilty using
concepts of knowledge canvas and IDEA Matrix.

I	D	E	A
Increase Accuracy	Drive Interaction	Educate Machine	Accelerate Processing
Improve Efficiency	Deliver Results	Evaluate Predictions	Associate Knowledge
Ignore Risks	Decrease Errors	Eliminate Inaccuracy	Avoid Failure

Table B.1: IDEA Matrix

ANNEXURE C
PROJECT PLANNER

	9/3	9/23	10/13	11/2	11/22	12/12	1/1	1/21	2/10	3/2	3/22

Study of emotion classification algorithms

Collecting and cleaning datasets

Study of human audio features

Study of algorithms for modifying audio

Deciding implementation approach

Implementing classifier module

Implementing audio feature modification module

Performance improvement

Testing and deploying the project

Documentation and open source release of the project

ANNEXURE D
PLAGIARISM REPORT

Plagiarism report

PLAGIARISMA

Scholar Google Plagiarism Software

Search Plagiarism   Google Scholar   Google Books   Article Rewriter   Spell Checker   Sign In   **GET FREE ACCESS**

" Some features that most students and teachers find interesting with the Google Scholar is its ability to search not only plagiarized phrases but also correct missing quotation marks and citations. This software scans your research paper, essay, coursework or dissertation completely from its database of internet resources, past exam papers, past essays, old dissertation papers and published journals.

X

Paste your text here (190+ languages supported!):

speech according to the emotion in
the text which will resemble closely to the human
speech. Our project, in general,
attempts to create an emotion based text-to-
speech conversion system, and in par-
ticular, tries to solve the problem of speech
distinction between angry and happy
emotions which has not been reasonably addressed
in the past. WE DO NOT STORE UPLOADED CONTENT

Enter URL to check:

[   ] Check URL

⦿ Articles and patents   ◯ Legal opinions and journals

☐ exact search

Select file: [Choose file] No file chosen   Upload

Valid file formats - PDF, DOC, DOCX, RTF, ODT, TXT, HTML.

Check Duplicate Content

You are using a limited version of plagiarism checker.
Quick sign-in with social networks:

87% Unique

Total 968 chars , 154 words, 5 unique sentence(s).

Custom Writing Services - *Paper writing service you can trust. Your assignment is our priority! Papers ready in 3 hours! Proficient writing: top academic writers at your service 24/7! Receive a premium level paper!*

ANNEXURE E

TERM-II PROJECT LABORATORY ASSIGNMENTS

Assignment 1

Title:
Review of design and necessary corrective actions taking into consideration the feedback report of Term I assessment, and other competitions/conferences participated like IIT, Central Universities, University Conferences or equivalent centers of excellence etc.

Earlier state of the project:
Simple sentences could be easily tagged with their emotions. But when some qualifying word comes, the emotion identification failed. It gave erratic results.
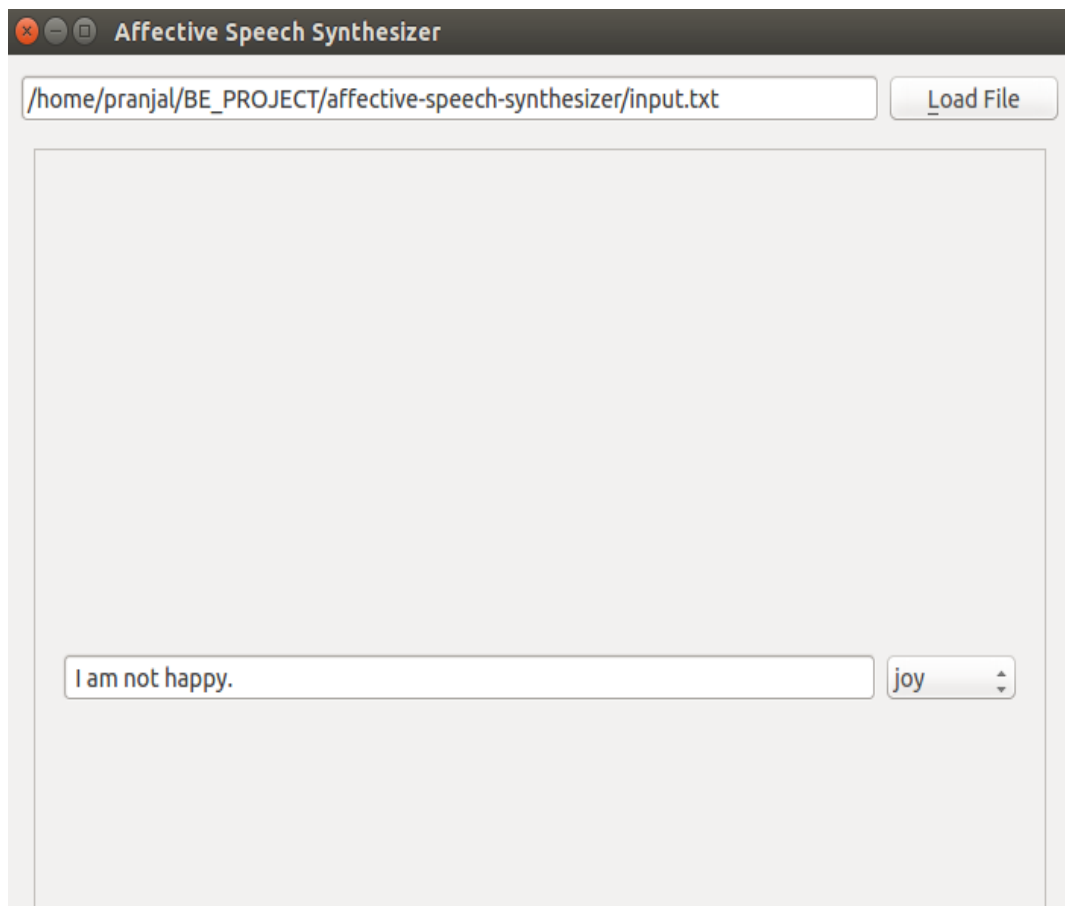

Figure E.1: I am not happy wrongly classified as joy

After Corrective actions:
Sentences containing not and qualifiers like very, extremely are handled.
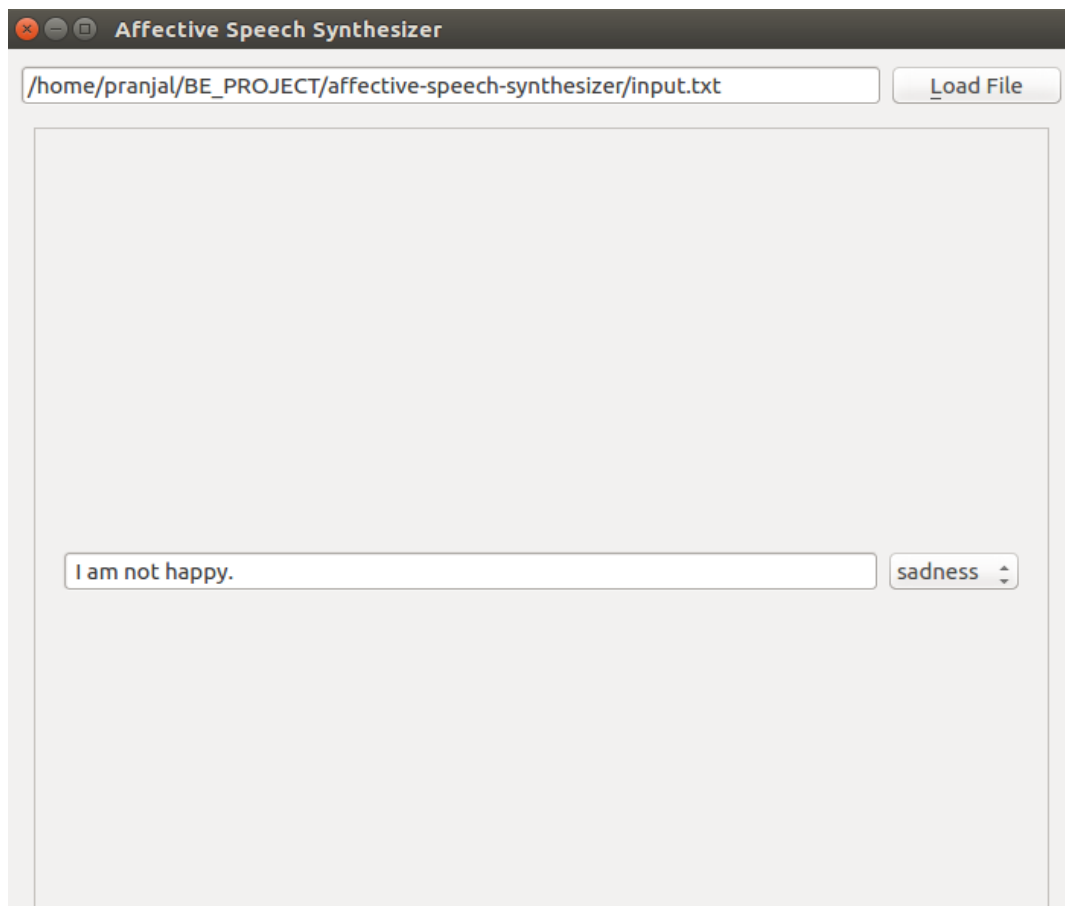

Figure E.2: I am not happy correctly classified as sadness

Assignment 2

Title:
Project workstation selection, installations along with setup and installation report preparations.

Workstation Selection:
Platform Linux platform is suitable because most of the dependencies of our software can be easily setup in it.
Number of cores in the machine Multi-core system is preferable because our software exploits multithreading for faster execution.

Dependencies:

1. marytts-5.2

2. python 2.7

3. NLTK library

4. Stanford Parser library

5. textract library

6. PyQT 4

7. Phonon PyQT 4

Installation:

1. Install project dependencies which includes: marytts-5.2, python 2.7, NLTK library, StanfordParser library, textract library, PyQT 4 and Phonon PyQT 4.

2. Download project source code

3. Execute 'run mary server.sh' script

4. Execute 'qtmain.py' script

Assignment 3

Title:
Programming of the project functions, interfaces and GUI (if any) as per 1 st Term term-work submission using corrective actions recommended in Term-I assessment of Term-work.
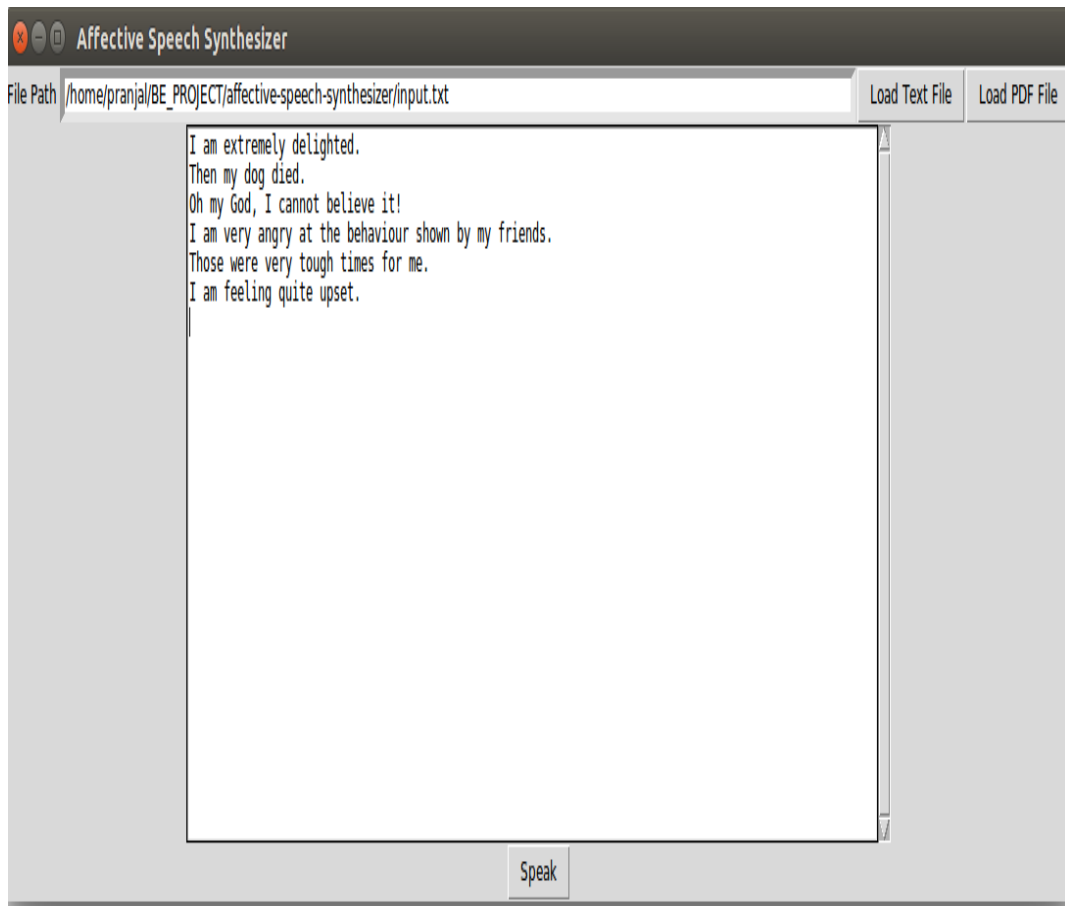
Project GUI as per 1st Term term-work:

Figure E.3: Naive GUI made using python TKinter

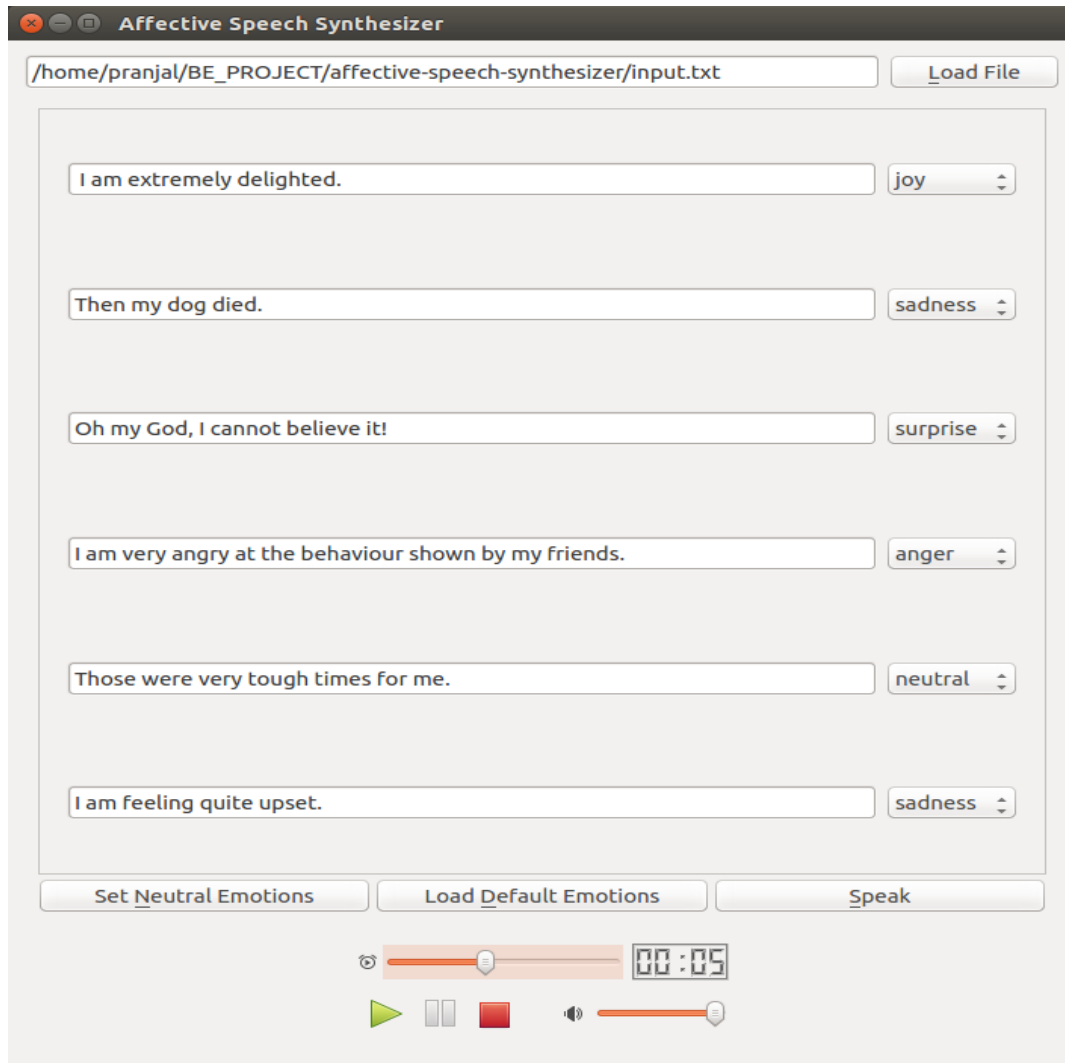Project GUI after corrective actions suggested by project guide:

Figure E.4: Improved GUI with native look-and-feel

Assignment 4

E.1 Title

Test tool selection and testing of various test cases for the project performed and generate various testing result charts, graphs etc. including reliability testing.

E.2 Type of Testing Used

Since our project focuses mainly on two important tasks Classification of sentences based on emotions using machine learning (A) and conversion of monotonic speech into expressive speech (B), we have analyzed different testing strategies based on the aforementioned tasks.

E.2.1 Unit testing

This testing was very valuable for our project since we were able to independently test the emotion identification module and the speech synthesis module with good results. We essentially decoupled the two subsystems and tested them.

E.2.1.1 Classification module

This subsystem was tested by providing some sentences rich with five emotions - joy, sadness, anger, surprise and neutral. With an accuracy of about 80%, it produced the results accordingly. It labelled more sentences as neutral because the database was biased toward it.

E.2.1.2 Speech synthesis module

For voice synthesis and digital signal processing unit, same sentences were given as input. The affective audio generated by the unit was perceived to contain the emotion tagged as reported by the test users. The sentences which depicted anger as the emotion, were spoken a bit fast by the system.

E.2.2 Integration testing

Having tested the individual subsystems, the emotion prediction and voice synthesis modules were then integrated. A sentence was passed to the input

function of the system. The emotion classifier correctly identified the emotion tag of the sentence and then using this emotion tag, the affective audio was generated.

E.2.2.1 Multithreading fix

It was discovered during testing that a single MaryTTS engine does not handle simultaneous requests and is not thread safe. Therefore, we spawned multiple instances of the engine and then fired simultaneous requests to them.

E.2.3 Validation testing

Since the ideas presented through this project are still in research phase, a usable product is a feat that cannot be achieved provided the current technology in this field. Still, we tried to test this software, targeting the audio book publishers as our client.

E.2.3.1 Emotional speech as output

This software provides the affective audio for emotions - joy, sadness, anger, surprise and neutral for the input provided by the user.

E.2.3.2 Input characteristics

Since majority of text is either available as .txt or as .pdf, this software has the ability to process both of them.

E.2.4 Performance and load testing

We used load testing to test the system under various loads. The workstation on which testing was performed consisted of a 2.5 GHz quad-core machine with 8 GB of main memory running Ubuntu operating system.

E.2.5 GUI testing

We made a very simple GUI in the beginning but soon found out that it is not user friendly and does not have a native-look-and-feel. Then we made another GUI which solved the problems of the previous one and offered better functionality in terms of audio playing and emotion selection.

E.2.5.1 Emotion selection

A drop down menu was associated with each and every sentence that represented the emotion of the sentence. The user has the ability to change this according to his or her preference.

E.2.5.2 Audio playback

A media player like functionality was added to the GUI so that the user can play, pause, stop, seek and change the volume of the audio being played.

E.2.5.3 Scrollable interface

If the number of sentences exceeded the minimum which can fit on the screen, the windows automatically provides scroll bars that can be used to navigate between the sentences.

E.3 Test Cases and Results

E.3.1 Unit testing

E.3.1.1 Classifier module

No.	Input	Expected Output	Actual Output
1.	I am very delighted	joy	joy
2.	My dog died	sadness	sadness
3.	Oh my God, I got that!	surprise	surprise
4.	I am very angry at that behaviour	anger	anger
5.	Those were tough times for me	neutral	neutral
6.	I am not happy	sadness	sadness
7.	I am not very sad	neutral	neutral

Table E.1: Test cases for classifier module

E.3.1.2 Speech synthesis module

No.	Input emotion	Expected Output	Actual Output
1.	joy	speaks in an overall pleasant way	as expected
2.	sadness	the pitch drops and tempo decreases	as expected
3.	surprise	the pitch increases, some words stressed	as expected
4.	anger	speaks a bit fast	speaks very fast
5.	neutral	speaks normally	as expected

Table E.2: Test cases for speech synthesis module

E.3.2 Integration testing

No.	Input	Expected Output	Actual Output
1.	affective sentence	emotional speech generated	as expected
2.	neutral sentence	neutral speech generated	as expected
3.	text file with various sentences	proper affective audio generated corresponding to emotions	as expected
4.	pdf file with various sentences	proper affective audio generated corresponding to emotions	as expected

Table E.3: Test cases for integration testing

E.3.3 Validation testing

No.	Feature	Implemented	Details
1.	produces affective voice on emotional text input	yes	can process 5 emotions
2.	handles text files	yes	can process .txt and .pdf
3.	automatically tags sentences with emotions	yes	currently 80% accurate
4.	support for other languages	no	future scope, data set limitations
5.	support for other accents	yes	training data to be provided

Table E.4: Test cases for validation testing

E.3.4 Performance and load testing

No.	Input	Time taken to execute
1.	5-10 sentences.	2-3 seconds.
2.	1 page.	About 15 seconds.
3.	10-15 pages.	About 3 minutes.
4.	A novel of about 300 pages.	About 1 hour 5 minutes.
5.	3-4 instances of our software running at the same time.	Sometimes system gets hung up.

Table E.5: Test cases for load testing

E.3.5 GUI testing

No.	Case	Expected Output	Actual Output
1.	loading files	.txt and .pdf files loaded	as expected
2.	changing emotion of a sentence	drop down updated	as expected
3.	sentences overflow the viewport	proper scroll bars are added accrdingly	as expected
4.	audio playing capability	play, pause, stop, seek, volume	as expected
5.	performing computational intensive task	show progress to user	no progress shown

Table E.6: Test cases for GUI testing

Emotion Classification testing

Input	Expected Output	Actual Output
I am very delighted	joy	joy
My dog died	sadness	sadness
Oh my God, I got that!	surprise	surprise
I am very angry at that behaviour	anger	anger
Those were tough times for me	neutral	neutral
I am not happy	sadness	sadness
I am not very sad	neutral	neutral

Testing output audio quality:
For voice synthesis and digital signal processing unit, same sentences were given as input. The affective audio generated by the unit was

perceived to contain the emotion tagged as reported by the test users.

Reliability Testing:
Load testing - We used load testing to test the system under various loads. The workstation o which testing was performed consisted of a 2.5 GHz quad-core machine with 8 GB of main memory and Ubuntu platform.

To this system varying loads were provided to check performance of the system.

Input	Time taken to execute
5-10 sentences.	2-3 seconds.
1 page.	About 15 seconds.
10-15 pages.	About 3 minutes.
A novel of about 300 pages.	About 1 hour 5 minutes.
3-4 instances of our software running at the same time.	Sometimes system gets hung up.

ANNEXURE F

INFORMATION OF PROJECT GROUP MEMBERS

Name : Pranjal Bhor

Date of Birth : Oct. 24th, 1994

Gender : Male

Permanent Address : Samartha housing society, shirsath mala, Savedi, Ahmednagar.

E-Mail : psmlbhor@gmail.com

Mobile/Contact No. : +918698652268

Placement Details : Veritas Software

Name : Vaibhav Chaudhari

Date of Birth : April 24th, 1995

Gender : Male

Permanent Address : "Mruduprakash", Plot No. 19, Samartha Colony, near M.J. College,Jalgaon.

E-Mail : vcvaibhav14@gmail.com

Mobile/Contact No. : +919403934320

Placement Details : Tech Racers

Name : Prathamesh Dharangutte

Date of Birth : May 18th, 1995

Gender : Male

Permanent Address : 13/357/1, Sona clinic, behind D-Mart, Datar mala, Kadapure Tal, Ichalkaranji.

E-Mail : pratham.d192@gmail.com

Mobile/Contact No. : +917588263102

Placement Details : HSBC

Name : Murtaza Raja

Date of Birth : Jan. 15th, 1995

Gender : Male

Permanent Address : 102/A zainy bldg, shehabi colony, rustampura, nr. police chowky, Surat 395002

E-Mail : murt.raja@gmail.com

Mobile/Contact No. : +918412806426