VOCAL CODE

[1] Gyan Prakash, [2]Aditya Bapat, [3]Ajay Shewale, [3]Sona R Pawara
[1,2,3] Student, [4]Professor
[1,2,3] Department of Computer
[1,2,3] Sinhgad Academy of Engineering, Pune, India

Abstract : Software development by means of programming languages involve keyboard for input. All programming languages are mostly text-oriented. This text familiarizes nature of programming languages is an obstacle to persons suffering from arms disability. Someone having dazzling mind and potential for programming skills, but suffering from arm injuries or being disabled could not become a programmer. To be a good developer it is important to remember the syntax and keywords of a programming language.

In our project, we propose a methodology for Java programming language where a programmer will speak and code in Java will be written accordingly.

Keywords **-** Vocal code, repetitive strain injuries (RSI), HMM, speech recognition, Natural Language Processing.**.**

I. INTRODUCTION

Vocal Code supports Java programming language. The project implementation consists mainly conversion of speech to text using voice recognition algorithms and then that text for coding purposes for them who are suffering from repetitive strain injuries (RSI) and related disabilities that make typing difficult or impossible. It is done by adding commands (methods, classes, conditional constructs, loop templates, etc.) and their spoken formats to the Vocal code. And we are using HMM, NLP for speech recognition and MFCC for feature extraction. Wherever possible, we have kept the spoken formats for Java language consistent with formats in Java language. Speech recognition software has helped many people who have difficulty in typing. Commercial speech recognition programs serve the purpose of dictation of text in a natural language, such as English. Programming languages were never supposed to be Vocal and have a lot of punctuation, unusual spelling and capitalization ("println", "compute Length"), and non-standard symbols (++, --, && etc.). Due to this standard speech recognition software cannot be simply used to write programs. Java speech support has the latent to greatly help the programmers who have difficulty using their hands. Java is one of the most popular programming languages. Hence Introducing speech typing for Java Language will be beneficial for all programmers

II. LITERACTURE SURVEY

Soon Suck Jarng in [1] discussed HMM voice recognition algorithm is explained and the importance of voice information DB is revealed for better improvement of voice recognition rate.

Stephen C. Arnold in [2] Vocal Programming. A system that enables a person to program without typing is needed because of the high incidents of repetitive stress injuries among people who program. This paper presents a design for a system that generates environments that enables people to program by voice and a method of determining if the system is successful. It also shows how this generator can be used to support entering data and writing XML documents.

Lawrence R. Rabiner in [3] discussed A tutorial on Hidden Markov Models and Selected Applications in Speech Recognition. In this paper they attempt to carefully and mathematically review the theoretical aspects of this type of statistical modelling and show how they have been applied to selected problems in machine recognition of speech.

Hyan-Soo Bae, Ho-Jin Lee, Suk-Gyu Lee in [4] discussed Voice Recogniton based on adaptive MFCC and deep learning.In this paper, they propose an enhanced voice recognition method using Adaptive MFCC and Deep Learning. To improve the voice recognition rate, it is important to extract the audio data from original signal. However the existing Algorithms which is used to remove the noise of particular band deteriorate the audio signal.

Yen-Huann Goh, Paramesran Raveendran in [5] discussed HMM-based speech recognition using adaptive framing. A common approach in mapping a signal to discrete events is to define a set of symbols that correspond to useful acoustic features of the signal over a short constant time interval. This paper proposes a hidden Markov models (HMM) based speech recognition by using cepstrum feature of the signal over adaptive time interval.

Mark Gales and Steve Weng in [6] discussed The Application of Hidden Markov Models in Speech Recognition. Hidden Markov Models (HMMs) provide a simple and effective framework for modelling time-varying spectral vector sequences. As a consequence, almost all present day large vocabulary continuous speech recognition (LVCSR) systems are based on HMMs.

G.S. Ng ; S.S. Erdogan ; W.N. Pan [7] Artificial Neural Networks (ANNs) have been used to perform classification for Automatic Speech Recognition (ASR). This paper proposes a new neural network, the Contenders' Network (CN) which requires little initial knowledge of the classification problem and lesser neurons than other ANNs.

III. PROPOSED SYSTEM

A Programming surroundings can create annoying barriers for the rising numbers of software developers that suffer from repetitive strain injuries (RSI) and related disabilities that make the typing difficult or impossible. Not only is the software development process contain somewhat text rigorous activities like program composition, editing, and navigation, but the tools used for programming are also operated textually.
The main purpose of the project is to reduce the efforts to write long codes and reduce the physical problems associated with it and also to make it possible for more and more people to write codes.

Our system is mainly divided into three Modules as shown in Figure 1.

- GUI(Graphical User Interface)
- Speech To Text Convertor
- Code Generator

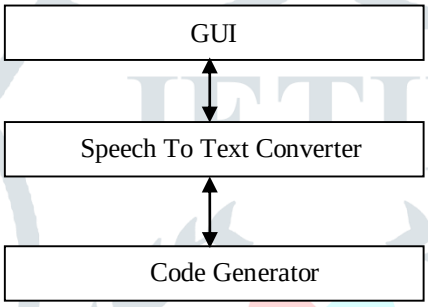

Figure.1:Modules for Vocal coding

let's have a brief overview of the high level working of our system, before getting into the details of the system. The GUI module provides an interface for users to interact with the system. Speech to text converter is responsible for converting each listened word to text. A user will speak code using microphone and text converter will convert this spoken code to text. For this converted text Java syntax and semantics are applied to this text to generate standardized Java code. That .java file is then compiled.

IV. MACHINE LEARNING APPROACHES

Machine learning is the field in which models are constructed from the training data (input data) for decision making. It makes use of different types of algorithms that can learn from and make predictions on data. The models created are data-driven and not follow the instructions which are strictly .For applying machine learning we have collected 2000 voice recordings from both male and female to create dataset for creating models, training models and testing models. Different types of machine learning models used are:

1. HMM(Hidden Markov Model)

HMM (Hidden Markov Model) recognition algorithm, for every new input voice signal, voice feature parameters are generated which are used in the learning process to create a new HMM model. So with each new HMM model created for every word, during the testing phase, all these models are compared with the test word to find out the matching voice sample.

2. Neural Networks

We will build a deep neural network that functions as part of an end-to-end automatic speech recognition (ASR) pipeline! The completed pipeline will accept raw audio as input and return a predicted transcription of the spoken language.

STEP 1: is a pre-processing step that converts raw audio to one of two feature representations that are commonly used for ASR.

STEP 2: is an acoustic model which accepts audio features as input and returns a probability distribution over all potential transcriptions. After learning about the basic types of neural networks that are often used for acoustic modeling, we will engage in our own investigations, to design our own acoustic model!

STEP 3: in the pipeline it takes the output from the acoustic model and returns a predicted transcription.

V. METHODOLOGY

STEP 1: Acoustic Features for Speech Recognition

For this project, we won't use the raw audio waveform as input to our model. Instead, we provide code that first performs a pre-processing step to convert the raw audio to a feature representation that has historically proven successful for ASR models. Wer acoustic model will accept the feature representation as input.

In this project, we will explore two possible feature representations. After completing the project, if we'd like to read more about deep learning architectures that can accept raw audio input, we are encouraged to explore this research paper.

Spectrograms

The first option for an audio feature representation is the spectrogram. In order to complete this project, we will not need to dig deeply into the details of how a spectrogram is calculated; but, if we are curious, the code for calculating the spectrogram was borrowed from this repository. The implementation appears in the utils.py file in our repository.

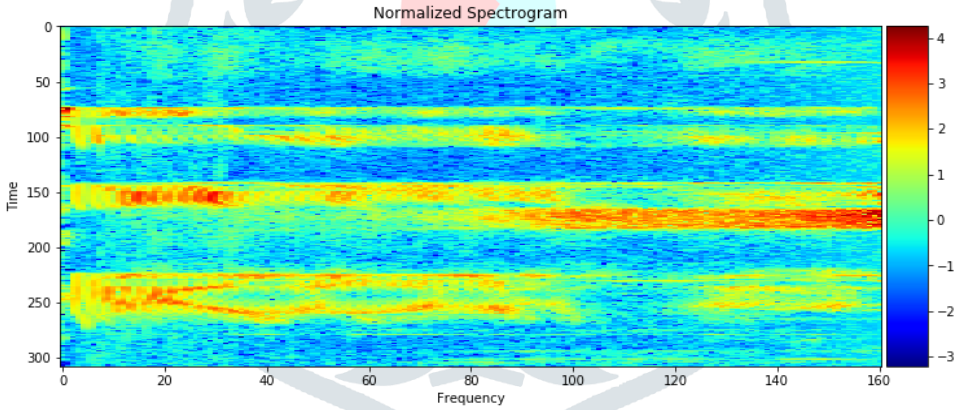

Figure 2:Spectogram

The code that we give returns the spectrogram as a 2D tensor, where the first (vertical) dimension indexes time, and the second (horizontal) dimension indexes frequency. To speed the convergence of algorithm, we have also normalized the spectrogram. (We can see this quickly in the visualization below by noting that the mean value hovers around zero, and most entries in the tensor assume values close to zero.)

Mel-Frequency Cepstral Coefficients (MFCCs)

The second option for an audio feature representation is MFCCs. We do not need to dig deeply into the details of how MFCCs are calculated, but if we would like more information, we are welcome to peruse the documentation of the python_speech_features Python package. Just as with the spectrogram features, the MFCCs are normalized in the supplied code.

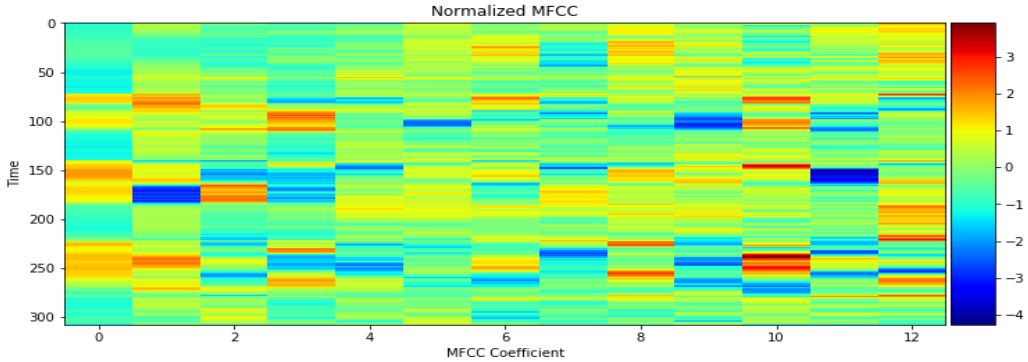

Figure 3:MFCCs

The main idea behind MFCC features is the same as spectrogram features: at each time window, the MFCC feature yields a feature vector that characterizes the sound within the window. Note that the MFCC feature is much lower-dimensional than the spectrogram feature, which could help an acoustic model to avoid over fitting to the training dataset.

STEP 2: Deep Neural Networks for Acoustic Modeling

In this section, we will experiment with various neural network architectures for acoustic modelling.we will begin by training five relatively simple architectures. Model 0 is provided for we. we will write code to implement Models 1, 2, 3, and 4. If we would like to experiment further, then create and train more models under the Models 5+ heading.

All models will be specified in the sample_models.py file. After importing the sample models module, we will train architectures in the notebook.

After experimenting with the five simple architectures, we will have the opportunity to compare their performance. Based on our findings, we will construct a deeper architecture that is designed to outperform all of the shallow models.

For our convenience, we have designed the notebook so that each model can be specified and trained on separate occasions. That is, say we decide to take a break from the notebook after training Model 1. Then, we need not re-execute all prior code cells in the notebook before training Model 2. We need only re-execute the code cell below, that is marked with run this code cell if we are resuming the notebook after a break, before transitioning to the code cells corresponding to Model 2.

Model 0: Given model - It was the only model trained on MFCC features. This model was the worst of them all according to the Training and validation accuracy. This model is very shellow as well and for the complicated task such as speech recognition we should be using deeper models.

Model 1: Fully connected layer after rnn: This model is a small upgrade on the model 0, in the model 0 we use outputs from RNN and from those we are calculating loss and probs for the spoken words. This is not good in practices as we have seen. The idea of the Model 1 is to add a fully connected layer between RNN part and output probs. In this way we are getting more information which we use to get better predictions. This add-on really helped in the matter of decreasing loss. As we can see on the left graph, model immediately started with loss of about 300. But this model didn't improve over time at both training loss and validation loss.

Model 2: CNN model: In this model we have added CNN layer at the beginning (before RNN layer). This way we are analysing Spectrogram or MFCC features with convolutional kernels. With this strategy we are able to get only important features and give them to the RNN part of the network. This strategy showed the best results of all models tested. Training loss kept decreasing over time and also validation loss decreased a lot. This results helped me in deciding what layers/architecture to use in the Final model.

Model 3: Deep RNN Network: This model is made to have many RNN layers. In the testing (above) I have used 2 RNN layers + fully connected layer after the last RNN layer. This model showed the same training results as our Model 1. At the beginning of the training process it started to decrease loss and after 2.5 epochs it just stopped and kept that amount of loss until the end of the training. Validation loss stayed constant throughout whole training process.

Model 4: Bidirectional RNN layers: This model uses Bidir_Rnn layers, which is logical use in this use case. With using Bidir_rnn we are able to analyse input signal from the beginning to the end and from the end to the beginning. This model showed similar results to models 1 and 3.

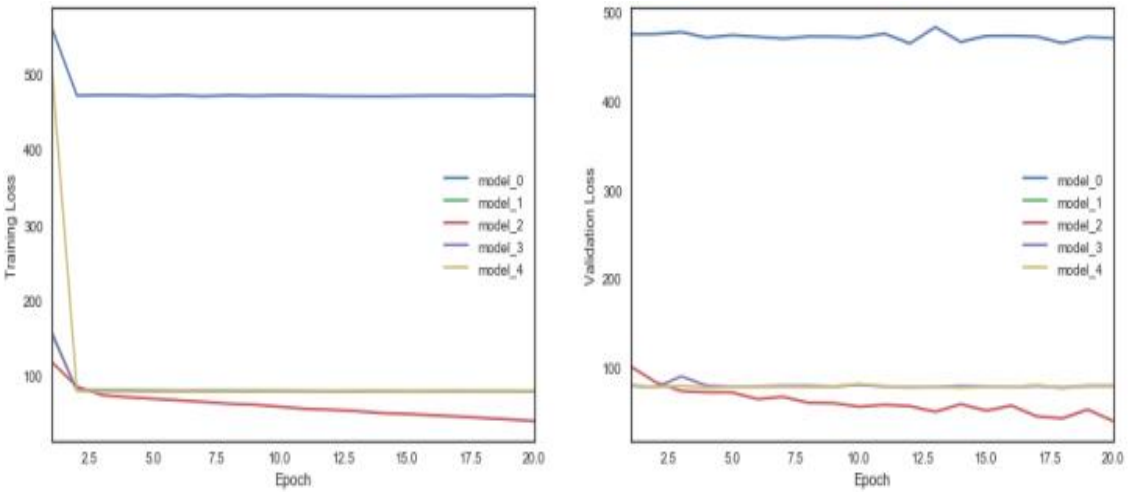

Figure 4:comparison between models

After comparison model 2 will be selected as the best model for training and validation as it has lowest training loss and validation loss among all the models. The model may change if any best model is generated after passing new data sets for training.

VI. ARCHICTURE

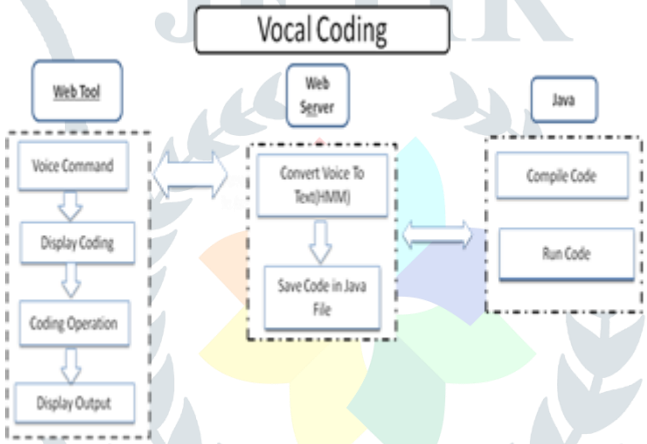

Figure 5: Architecture

From the architecture, it can be seen that through the web tool, user passes their voice command using mic which is recorded and are forwarded to the server where conversion of voice to text and then text to .java file takes place. Which is later pass for compilation and the compiled code is displayed on web tool as well its output. Such systems provide an alternative way for the evaluation of computer programs.

VII. CONCLUSION

This paper provides possible approaches that are best suitable to develop a speech recognition system to learn programming concepts for people suffering from repetitive strain injuries (RSI) and related disabilities that make typing difficult or impossible . Over the last few decades, advancements in web phone technologies have reached an impressive level. With the advancement of web technology, the society has changed drastically. This research provides a solution to these problems by developing the system "Vocal Code" to take them forward. This system mainly focused on developing software that acts as an assistant, especially for RSI and related disabilities people. “Vocal Code” is an intelligent system, based on speech recognition and user interaction system. Capturing and recognizing the given sound input is one of the most important features. Moreover, also providing the ability to use anywhere anytime without hard efforts. The ultimate goal of this research is to make provision for people who are unable to type code, solve their programming learning related issues for empowering the differently able. The application that building may contain a limited number of datasets with the sounds and words and also hope to get accuracy more than 40%. Accuracy of voice decides the accuracy of code. As this system developed for RSI affected and handicapped people as a web application, this will be a big help to them. "Vocal Code" would be a baseline for future weng inventors to use for their researchers. By using this framework as a building block, they can continue building what they want on top of this. This will be an open door for future developers as a platform for voice recognition.

VIII. ACKNOWLEDGMENT

We take this opportunity to thank all the people involved in the making of this paper. We want to especially thank our respected guide, Prof. Sona R Pawara for her guidance and encouragement, which has assisted us to achieve our goal. Her valuable advice has helped us throughout. Our Head of Department Prof. B.B Gite and Project Coordinator Mr. Santosh Shelke has also been very helpful and we are grateful for the support he provided us with. Last but not the least we would like to convey our gratitude to all the teaching and non-teaching staff members of our department, our friends and families for their valuable suggestions and support.

REFERENCES

[1] Soon Suck Jarng :The HMM voice recognition algorithm . DOC: 26-29 April 2011,IEEE, ISSN: 2162-9048
[2] Cristian IONIŢA"Building Domain Specific Languages for Voice Recognition Applications",Revista Informatica Economică nr. 2(46)/2008.
[3] Lawrence R. Rabiner, IEEE 'A Tutorial On Hidden Markov Model And Selected Applications In Speech Recognition, Proceedings Of The IEEE, Vol. 77, No. 2, February 1989.
[4] Hyan-Soo Bae, Ho-Jin Lee, Suk-Gyu Lee, Voice Recogniton based on adaptive MFCC and deep learning.2016 IEEE 11th Conference on Industrial Electronics and Applications (ICIEA)
[5] Yen-Huann Goh, Paramesran Raveendran, HMM-based speech recognition using adaptive framing ,IEEE ISBN: 978-1-4244-4546-2. Date of Conference: 23-26 Jan. 2009
[6] Foundations and TrendsR in Signal Processing Vol. 1, No. 3 (2007) 195–304 c 2008 M. Gales and S. Weng DOI: 10.1561/2000000004
[7] G.S. Ng ; S.S. Erdogan ; W.N. Pan, ANN used for voice recognition.**IEEE**, **DOI:** 10.1109/SICON.1993.515791

BIOGRAPHY

Gyan Prakash
Pursuing Bachelor of Engineering (B.E) in Computer from Sinhgad Academy of Engineering, Savitribai Phule Pune University (S.P.P.U)
gyan3233@gmail.com.

Aditya Bapat
Pursuing Bachelor of Engineering (B.E) in Computer from Sinhgad Academy of Engineering, Savitribai Phule Pune University (S.P.P.U)
bapatadi@gmail.com

Ajay Shewale
Pursuing Bachelor of Engineering (B.E) in Computer from Sinhgad Academy of Engineering, Savitribai Phule Pune University (S.P.P.U)
ajayshewale21@gmail.com

Prof. Sona R Pawara
Prof. in Sinhgad Academy of Engineering, Savitribai Phule Pune University (S.P.P.U)
sona.g18@gmail.com