S	
E1	
E2	
For Office Use Only	

Masters Project Final Report

(MCS)

2019

Project Title	Speech recognition for Tamil
Student Name	N.Suvethan
Registration No. & Index No.	Reg No : 2017/MCS/081 Index No: 17440811
Supervisor's Name	Dr. A.R.Weerasinghe

For Office Use ONLY

Declaration

The thesis is my original work and has not been submitted previously for a degree at this or any other university/institute.

To the best of my knowledge it does not contain any material published or written by another person, except as acknowledged in the text.

Student Name: N. Suvethan

Registration Number: 2017MCS081

Index Number: 17440811

____

Signature: Date:

This is to certify that this thesis is based on the work of

Mr./Ms.

Under my supervision. The thesis has been prepared according to the format stipulated and is of acceptable standard.

Certified by:

Supervisor Name: Dr. A.R.Weerasinghe

____

Signature: Date:

Speech recognition for Tamil

A dissertation submitted for the Degree of Master of Computer Science

N.Suvethan
University of Colombo School of Computing

2020

Table of Contents

List of Figures

List of Abbreviations

ML – Machine Learning

AI – Artificial Intelligence

ASR – Automatic Speech Recognition

MFCC - Mel-Frequency Cepstral Coefficients

CMVN - Cepstral Mean and Variance Normalization

JFA - Joint Factor Analysis

GMM - Gaussian Mixture Model

DNN - Deep neural networks

Chapter 1

Introduction

1.1 Introduction

Mobile based applications have been heavily used in our day to day life to accomplish things in modern world. Input data is the main source for these applications to process the things that user intended. These data are in various different formats. Some may be text, pictures, video, audio etc. Among these formats of data, textual, voice/audio and video data are being the most widely used format in the world.

Now a days, voice data is becoming important source for mobile applications in the fast moving environment. Most of the people are using their voice to interact with applications to get the things done than typing things which will take more time and sometimes if you are in a rush you don’t like to spend time on texting for your purpose. When it comes to text, sometimes it needs formatting, filters and classifications. Although, voice is heavily used it is also limited certain languages and also contains locality barriers. In this research it is being looked for an efficient research papers to model voice data for Tamil speaking people who are living in suburbs.

1.2 Problem domain

This study is correlated with few major areas in modern computer science. Machine learning based techniques for deep speech learning and feature extraction and acoustic modelling of data using Kaldi toolkit are some of them. Under this background section it is explained in brief what are these fields in order to compose the background of the quest which have been carried out.

1.2.1 Machine Learning

It is correct if Machine Learning (ML) is stated as the brain science of modern computer science. This is a sub branch which comes under the tree of Artificial Intelligence (AI).

Simply ML is the study about the pattern description and prediction with statistical based algorithms. These algorithms are able to act automatically without any human interference. In many cases these algorithms have three phases. Implementing, training and testing are those three phases. After trained an algorithm with proper data they are able to find some unseen information from newest data which haven’t met the algorithm before. So these algorithms do act as filters which can be used to separate and extract information from data. Therefore ML can be replaced the position of the newest computer brain science as it use some basic instruction or prior experiences which have met earlier, to gain and extract some unseen information from new data. ML is not a field that stand by itself, it has many overlaps with statistics and base of AI. ML algorithms are applied in many areas today, such as Face recognition, Fraud detection,
Task scheduling, Elevator scheduling, online recommendation and etc.

1.2.2 Kaldi toolkit

Kaldi is an open source toolkit made for dealing with speech data. It is mostly used for speech recognition but also for other tasks such as speaker recognition and speaker diarisation. The toolkit is developed 7 years ago still constantly updated and further developed by a large community. Kaldi is widely adopted both in Academia (400+ citations in 2015) and industry.

Kaldi is written mainly in C/C++, but the toolkit is wrapped with Bash and Python scripts. For basic usage this wrapping spares the need to get in too deep in the source code.

1.3 Research problem

Computer software basically has two types of requirements in an application domain. Those are functional and non-functional requirements. Both are equally important to be satisfied in industry. Functional requirements tells how and what the software contains of. In other words, it describes features and functions provided by software. Non-functional requirements which are known as quality requirements describes the constraints and conditions of the software design and implementation.

In this technology era, software development industry has vast growth in terms of many softwares spreading across many industry domains. In other word, this potentiality has moved to usage of computer software into large scale, mission and life critical applications in high stake industries such as medicine, nuclear power, defence, etc. It involves higher trade-off between cost and the quality of the output that the software can produce. Hence, it can be identified quality is one of the important factor in many areas and many industries around the world.

Software quality has been described by many characteristic aspects. There are number of software quality metrics used to evaluate the qualities of the software such as Scalability, Security, Reliability, Usability, etc [1]. Among all these quality metrics, software usability has been described as one of the important quality attribute concerned and specially, software localization is the key to success in an international market [2]. Since, software localization limitation can cause of huge problem in terms of users who uses the software application which indirectly makes the application producer or provider to lose billion dollars of money and business [3].

Localization refers to the adaptation of a product, application or document content to meet the language, cultural and other requirements of a specific target market (a locale). In every business today has different levels of dependability on the

internet and computer software, it has increasingly critical to address localization needs [4].
The problem of this research is that suburb people who does not know any language other than their mother tongue faces lots of problem when it comes to communication. People faces lots of difficulties and as a result they lose lots of important things in their day to day life. This research mainly focused on Tamil speaking community to avoid the void they have in communication.

1.4 Motivation & Contribution

There is an ongoing research on speech to text in Sinhala which gave the idea to develop this for Tamil speaking community, specially the people in suburbs who only speaks Tamil. The main problem of this research is collecting the dataset and its transcript where voice dataset consists of colloquial Tamil. This research is focusing on developing an application to recognize Tamil speech to Tamil text and convert output text to English speech using Google APIs to break the language barrier when using software applications. The following components have been addressed in this research.

1. Finding speech corpus data set which is a combination of recorded speech and their corresponding transcriptions to recognize the speech.
2. Creating good training data set for the research.
3. Apply machine learning techniques to identify the speech.
4. Produce the most accurate translated speech to end user.

1.5 Scope of the study

1. Voice samples will be collected for Tamil speech covering all the diversed areas from different age groups and different gender.
2. Collected voice samples will be checked one by one and all the inconsistent samples will be removed.
3. All the properly sorted voice samples then pre-processed and trained through machine learning techniques.
4. A software application will be developed to pass the Tamil speech as input and using the training dataset then the output Tamil text will be recognized accordingly.

Chapter 02

Literature Review

2.1 Introduction

In this chapter, literature of the related previous works are compared and analyzed in order to get more understanding on current outcomes, future research works and the improvements to be done on the related domain. This study associated with the research field of linguistics and machine learning. Thus, this chapter intends to explore theoretical fundamentals of the techniques which has been applied in this research work.

As per the reviewed literature, many author's work on linguistics has been focused on modeling speech recognition engine and translating it to text. Further, most of the other related works has been focused on providing web API to access speech recognition engine and display the translated text to different places. Many research studies have been made on deep speech research area and noticed still end to end speech recognition is only available for English, Spanish, Mandarin and Chinese. There are few samples of end to end speech recognition from English to Spanish and vice versa [4].

2.1.1 Direct speech-to-speech translation with a sequence-to-sequence model

This research is written by Ye Jia, Ron J. Weiss, Fadi Biadsy, Wolfgang Macherey, Melvin Johnson, Zhifeng Chen and Yonghi Wu where they presented an attention-based sequence to sequence neural network which can directly translate speech from one language into another language [5]. Moreover in their research the network is trained end to end and learning to map speech spectrograms into target spectrograms in another language corresponding to the translated content.

2.1.1.1 Model architecture

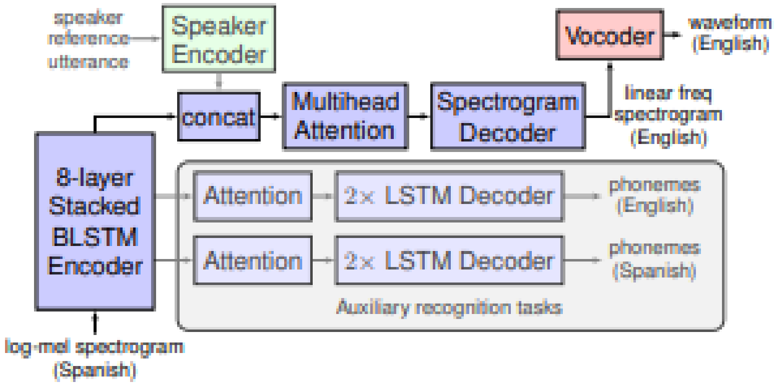


Figure 1- Direct speech-to-speech translation with a sequence-to-sequence model

Above architecture produces English speech (top right) from Spanish speech (bottom left), and a non-compulsory speaker reference utterance (top left) which is only used for voice transferal experiments in Section 3.4. The speech model is multitask trained to forecast source and target phoneme transcripts as well, however these additional tasks are not used during inference. Optional components are drawn in light colors.

This study experimented two types of Spanish-to-English translation datasets. They are,

- Conversational Spanish-to-English
- Fisher Spanish-to-English

Evaluation has been done for speech to speech by computing Bilingual Evaluation Understudy (BLEU) marks as an objective of speech intelligibility and translation quality by using a pre-trained ASR system to identify the generated speech and comparing the results to ground truth translations.

The limitation of this research is that it provides good quality but when it comes to performance it is not good as a standard cascade of Speech to Text (ST) and Text to Speech (TS) models.

This research study helps to use the models and apply it to do Tamil speech to text conversion and since the quality of the result is high even though it has performance issues.

2.1.2 Speech Recognition System for Sinhala

This is an ongoing research in UCSC on Sinhala speech to text which made think about taking this research to another step by directly translating the Tamil speech to text [6]. The main objective of this project is to enhance human computer interaction by developing Automatic Speech Recognition (ASR) engine for Sinhala that allows people in the society to use digital devices without language barrier.

An acoustic model is used in Automatic Speech Recognition (ASR) to represent the connection between an audio signal and phonemes (or other linguistic units) that make up speech. The model is learned from audio datasets and its annotated transcripts. Such datasets are created by taking voice recordings of human speech and their transcriptions and then compiling them into statistical representations of the sounds which make up words. Sequence of sounds and words are modelled by the language model to predict the most likely spoken word. The language model also used to recognize the words and phrases that sound similar.

2.1.2.1 ASR Building Process & Recognition Process

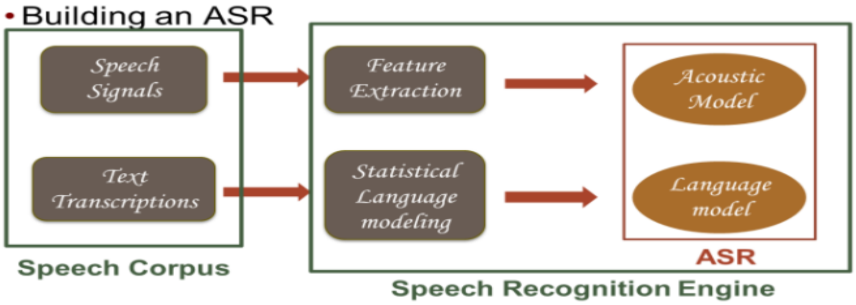


Figure 2 - Building ASR

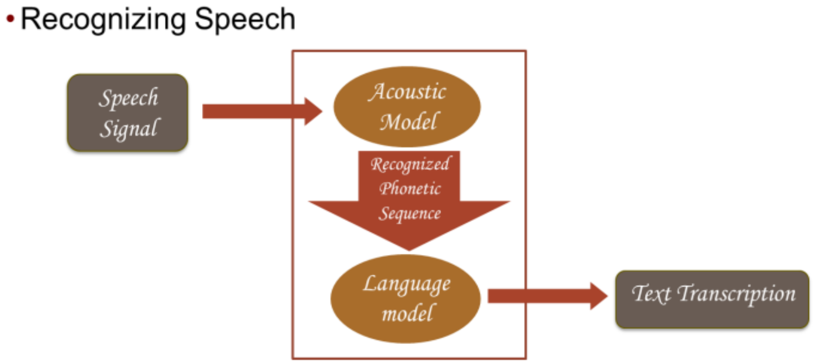


Figure 3- Recognizing Speech

The most difficult issue is to build an ASR system from scratch as it needs good training data sets to give quality results which can be used in software applications. The type of data set which is used for recognizing speech is called Speech Corpus. A speech corpus is a combination of recorded speech (*Acoustic data*) and their corresponding transcriptions (*Labels*). Labeling the acoustic data appropriately is the hardest part in building process.

2.1.3 Automatic English speech to Tamil speech translation system

J.Poornakala and A.Maheshwari has written a research paper on automatic English speech to Tamil speech translation and they proposed a solution by only keeping both English and Tamil words in a database and converted the speech to text queried through database for matches [7]. If any matches found, then the system returns the matched word to screen.

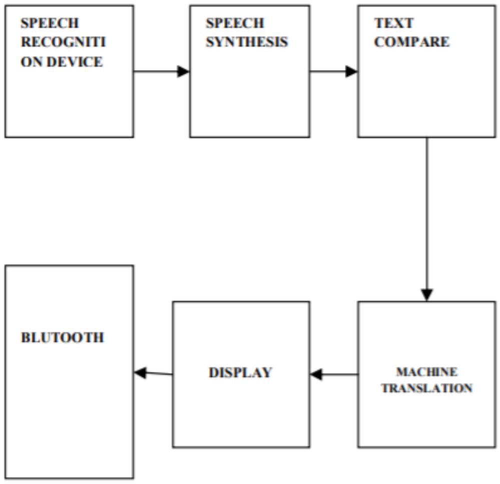


Figure 4 - Automatic English speech to Tamil speech model

Speech recognition system that recognizes the English speech from speech recognition device and then translated into English text using speech synthesis. Thereafter English text is converted into text format for comparison with the words stored in the database if it matches with the Tamil text stored in the database then, English text is converted into Tamil text with the help of the machine translation system and finally the text will be displayed on the screen. Bluetooth is used to send transferred data forward.

One of the foremost limitations of this research paper, the accuracy level of speech recognition is very low and can be improved by applying machine learning techniques without directly keeping everything in database.

This research work helped me to get an overall picture of how to divide building translation process and convert the translated text to display outside using Bluetooth.

2.1.4 Online speech translation system for Tamil

Maadhavaraj A, Shiva Kumar HR and A G Ramakrishnan have done a research on online speech translation for tamil in recent past. The purpose of this research to help users who wish to learn English through Tamil [8].

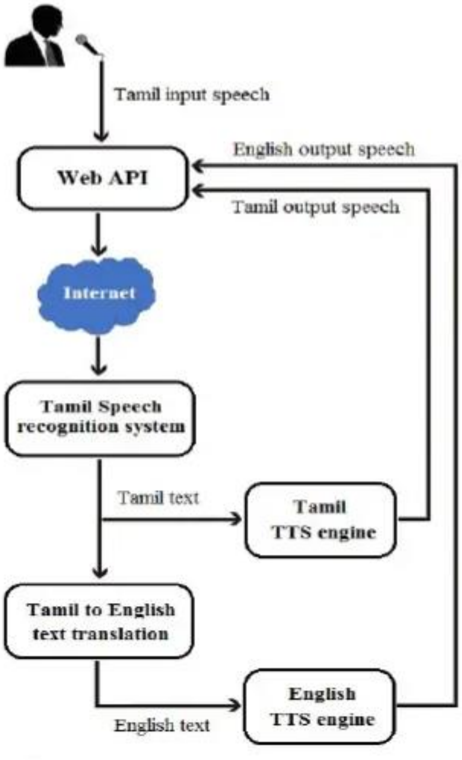


Figure 5- Online speech translation system flow

They have developed Web API which interacts with the Tamil speech recognition system and then recognized text will be translated using Google's cloud translation API and finally translated English text is passed to English text to speech library to output English speech.

The main limitation of this research is that it is mainly focusing on proper pronunciation, therefore when it comes to accuracy of translation is very low in noisy environment and people with different accents.

Chapter 3

Methodology

3.1 Introduction

The hypothesis of the research work is mainly based on the speech recognition that there are clearly identifiable Tamil text for the Tamil speech. Thus, this work is focusing on collecting all the possible speech scenarios from different gender and age groups to translate Tamil speech to Tamil text with maximum accuracy and then translate the output text to English speech using Google APIs.

The datasets required for this research work is acquired by manually collected voice data sets with transcripts.

3.2 Dataset Creation

3.2.1 Voice Dataset

Voice samples are collected from 100 speakers of different ages, genders, and from different areas of the country. The speakers are prompted to read out a pre-selected set of sentences which will cover most of the speech sounds in the Tamil language.

3.2.2 Preprocessing and Feature Extraction

Most of the models that works with audio data deals with some pixel-based rendition of that data. We need to use features that provides good representation of data. The features we need to look for needs to be good at following things.

1. Recognizing the sound of human speech.
2. Removal of any unnecessary noise

For many years there were several attempts to make those features and today **MFCCs** which are widely used in the industry.

3.2.2.1 MFCC

MFCC has become almost a standard in the industry since it was invented in the 80s by Davis and Mermelstein. In simple words, MFCCs are only taking the sounds that are best heard by our ears into account [9].

In Kaldi we use two additional features:

1. **CMVNs** which are used for better normalization of the MFCCs
2. **I-Vectors** are used for better understanding of the variances inside the domain. I-Vectors are based on the same concepts of Joint Factor Analysis (JFA), but are more appropriate for understanding both channel and speaker variances.

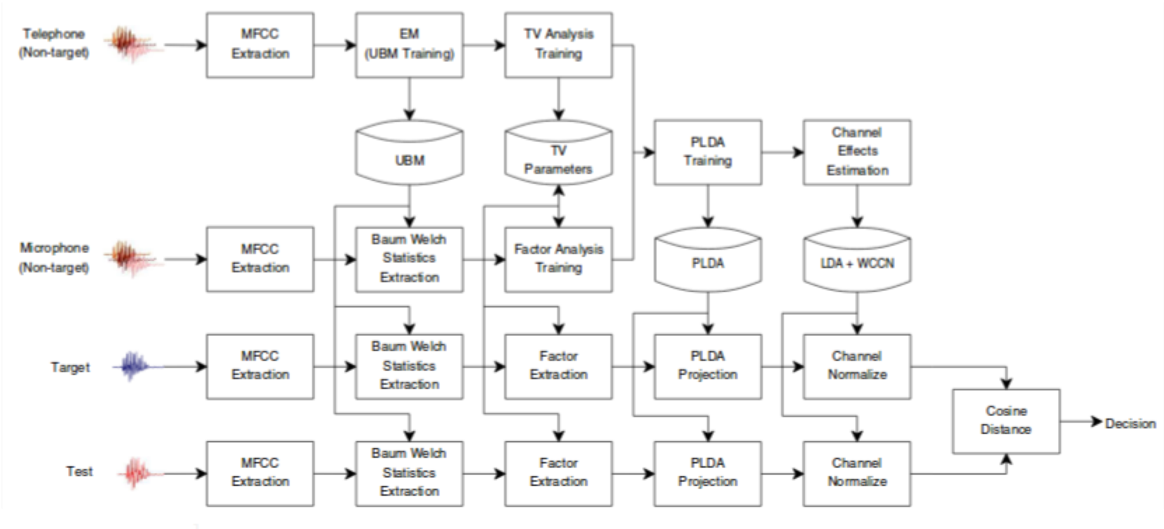


Figure 6 - The Process of using I-Vectors

In summary, **MFCC** and **CMVN** are used for indicating the **content** of each audio utterance and **I-Vectors** are used for indicating the **style** of each audio utterance or speaker.

3.2.3 Dataset modelling

Kaldi’s model can be divided into two main components. The first part is the **Acoustic Model**, which used to be a Gaussian Mixture Model (GMM) but now it was widely replaced by Deep neural networks (DNN). That model will transcribe the audio features that we created into some sequence of context-dependent phonemes (in Kaldi dialect we represent them by numbers).

The second part is the **Decoding Graph**, which takes the phonemes and turns them into lattices. A lattice is a representation of the alternative word-sequences that are likely for a particular audio part. This is generally the output that you want to get in a speech recognition system. The decoding graph takes into account the grammar of your data, as well as the distribution and probabilities of contiguous specific words (n-grams).

3.2.4 Training Process

As a whole, this is the trickiest part. In Kaldi you’ll need to order your transcribed audio data in a particular order to train that is described in depth in the Kaldi documentation [10].

After organizing your data, you’ll need a representation of each word to the phonemes that create them. This representation will be termed “dictionary” and it will decide the outputs of the acoustic model.
Once you have both of those things at hand, you can start training your model. The different training steps you can use are named in Kaldi dialect “recipes”. The most widely used recipe is WSJ recipe. In most of the recipes we are starting with aligning the phonemes into the audio sound with GMM. This basic step (named “alignment”) helps us to decide on what sequence we want our DNN to spit out later.

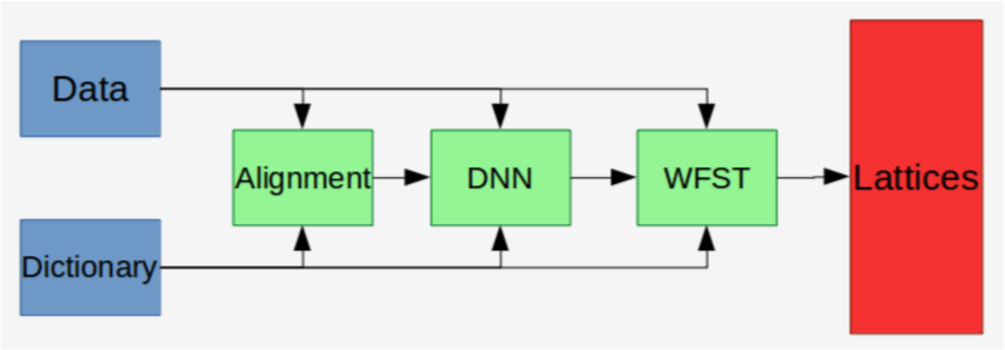


Figure 7 - Training process

Once the alignment is done we will create the DNN that will generates the **Acoustic Model**, and we will train it to match the alignment output. After creating the acoustic model we can train the WFST to transform the DNN output into the wanted lattices.

3.3 Solutions for Infrastructure and Platform

This section explains the reasons for selecting particular infrastructure for the study of the quest. In general it is known that ML algorithms, libraries and tools require high performing hardware utilities for the calculation tasks. The reasons for these decisions are further clarified in sections below.

3.3.1 PC Hardware

This is one of the options for infrastructure and platform from the available two. Thus as explained in previous section the best solution available for the infrastructure is to use the normal commodity personal computer which was in use. The hardware availability of the device is well enough to meet the requirements which needed for study. It was Intel(R) Core(TM) i7-8550U CPU @ 1.80 GHz – 1.99GHz Processor with 8GB memory which is running windows 10 version operating system.

3.4 Evaluation

A working software to recognize end to end speech is the primary outcome of this research work and they uses trained datasets for speech recognition. The test data set will contain all diversed speech scenarios with different gender and age groups. Accuracy of the speech recognition will be tested and compared to produce the final evaluation results.

3.4.1 Project Plan and Timeline

Task	Month								
Project Design, Proposal and Scope	■								
Problem definition, goals and objectives identification	■								
Literature Survey	■	■	■	■	■	■	■	■	
Collection of data set and training the datasets to apply it for speech recognition.			■	■	■				
Analysis and validations on results obtained					■	■	■		
Development of software application for Tamil speech to text recognition with trained data sets & evaluation							■	■	
Project Presentations and Viva									■
Documentation	■	■	■	■	■	■	■	■	■
Supervisor discussions and progress reviews	■	■	■	■	■	■	■	■	■

3.4.2 List of Deliverables

1. Dissertation about the research work.
2. Collect voice dataset from different gender and age groups.
3. Fine-tune and remove the inconsistent data from dataset.
4. Trained the fine-tuned datasets using machine learning techniques.
5. Application to recognize speech to text and convert text to English speech using Google API

Chapter 04

Evaluation

4.1 Approach

I’m applied experimental based evaluation by experimenting results in following ways.

1. “Male” only dataset modelling, training and testing.
2. “Female” only dataset modelling, training and testing.
3. “Mixed” dataset modelling, training and testing.

I collected voice data sets from 6 individuals 3 males and 3 females and I gave 50 sentences to them and asked to speak. I recorded those things as m4a format and later converted it to wave format with sample rate 8000 and 16 bit PCM.

4.2 Tools

4.2.1 Kaldi toolkit

Kaldi is an open source toolkit made for dealing with speech data. It is mostly used for recognizing speech but also used for other tasks such as speaker identification and speaker diarisation. The toolkit is developed 7 years ago still continuously updated and further developed by a large community. Kaldi is widely adopted both in Academia (400+ citations in 2015) and industry.

Kaldi is written mainly in C/C++, but the toolkit is wrapped with Bash and Python scripts. For basic usage this wrapping spares the need to get in too deep in the source code.

4.2.2 Kaldi Gstreamer server

This is a real-time full-duplex speech recognition server, based on the Kaldi toolkit and the GStreamer framework and implemented in Python.

Features

1. Full duplex communication based on web sockets: speech goes in, partial hypotheses come out (think of Android's voice typing)
2. Very scalable: the server consists of a master component and workers; one worker is needed per concurrent recognition session; workers can be started and stopped independently of the master on remote machines
3. Can do speech segmentation, i.e., a long speech signal is broken into shorter segments based on silences
4. Supports arbitrarily long speech input (e.g., you can stream live speech into it)
5. Supports Kaldi's GMM and "online DNN" models
6. Supports rescoring of the recognition lattice with a large language model
7. Supports persisting the acoustic model adaptation state between requests
8. Supports unlimited set of audio codecs (actually only those supported by GStreamer)
9. Supports rewriting raw recognition results using external programs (can be used for converting words to numbers, etc)
10. Python, Java, Javascript, Haskell clients are available

Chapter 05

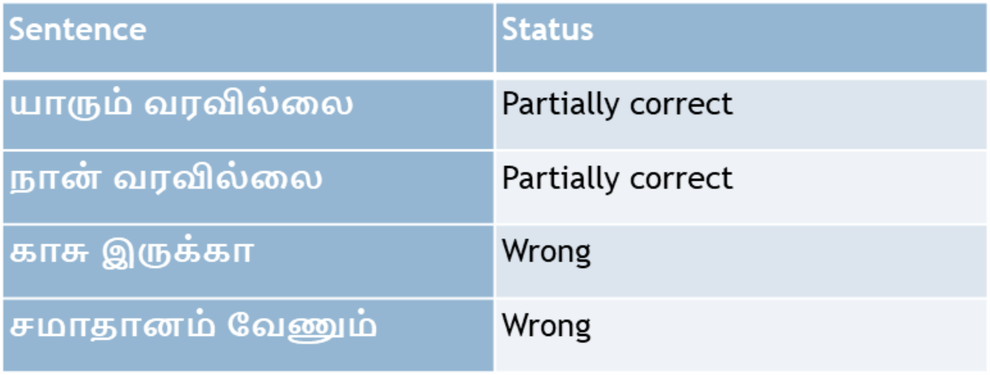
Experiment

5.1 Testing

I have done experiment with male and female test data with following trained datasets.

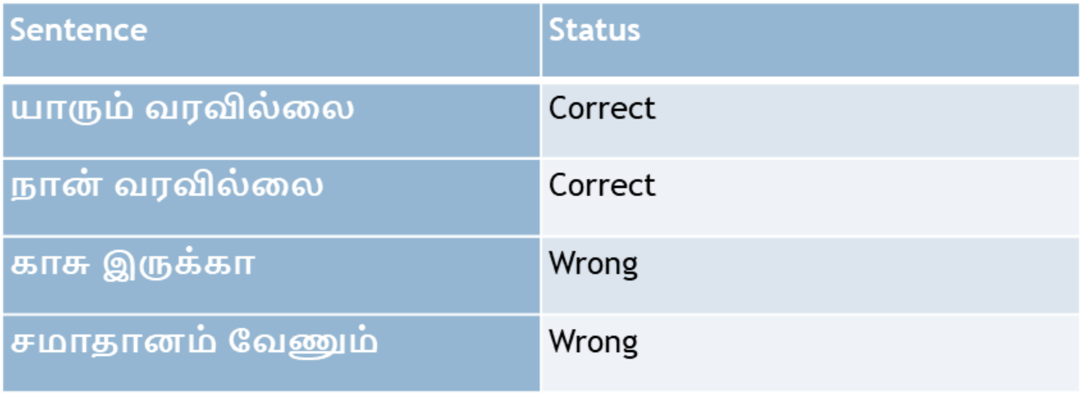
1. Female only trained dataset.
2. Male only trained dataset.
3. Mixed trained dataset.

5.1.1 Male only trained dataset with tri2b



Sentence	Status
யாரும் வரவில்லை	Partially correct
நான் வரவில்லை	Partially correct
காசு இருக்கா	Wrong
சமாதானம் வேணும்	Wrong

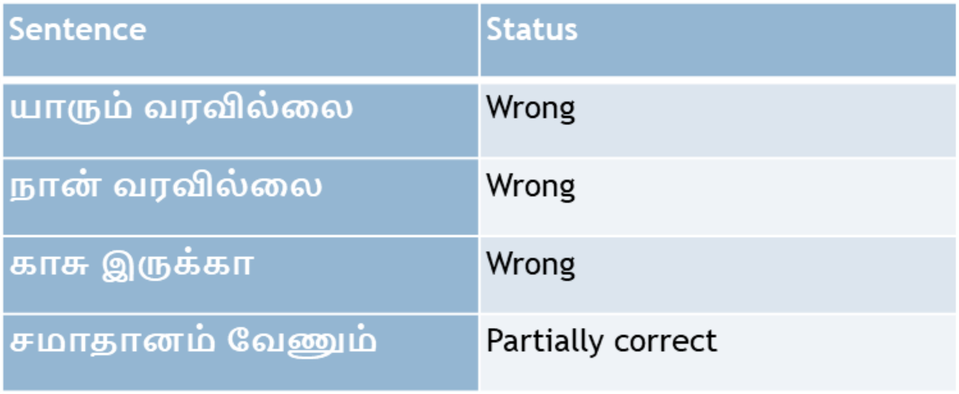
Figure 8 - Male test data for male only dataset tri2b



Sentence	Status
யாரும் வரவில்லை	Correct
நான் வரவில்லை	Correct
காசு இருக்கா	Wrong
சமாதானம் வேணும்	Wrong

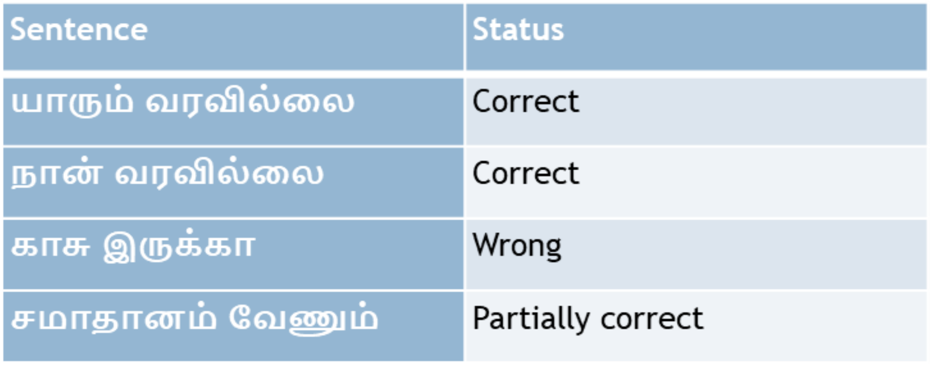
Figure 9 - Female test data for male only dataset tri2b

5.1.2 Female only trained dataset with tri2b



Sentence	Status
யாரும் வரவில்லை	Wrong
நான் வரவில்லை	Wrong
காசு இருக்கா	Wrong
சமாதானம் வேணும்	Partially correct

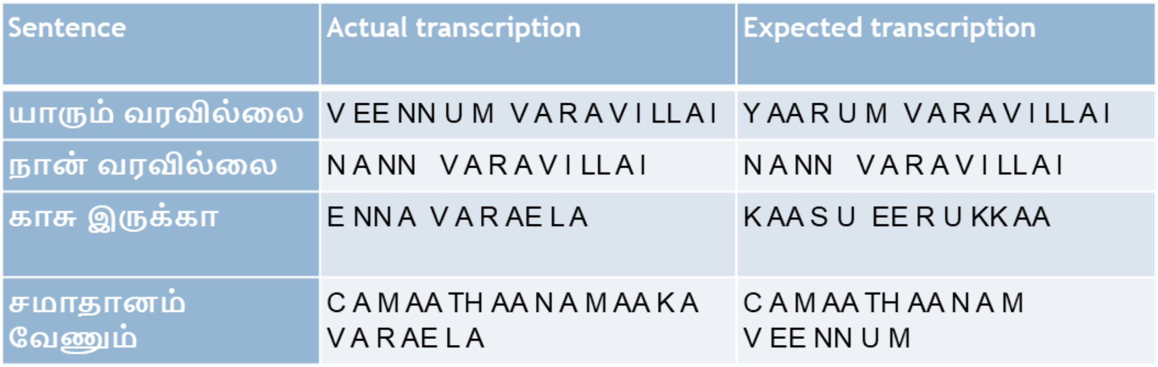
Figure 10 - Male test data for female only dataset tri2b



Sentence	Status
யாரும் வரவில்லை	Correct
நான் வரவில்லை	Correct
காசு இருக்கா	Wrong
சமாதானம் வேணும்	Partially correct

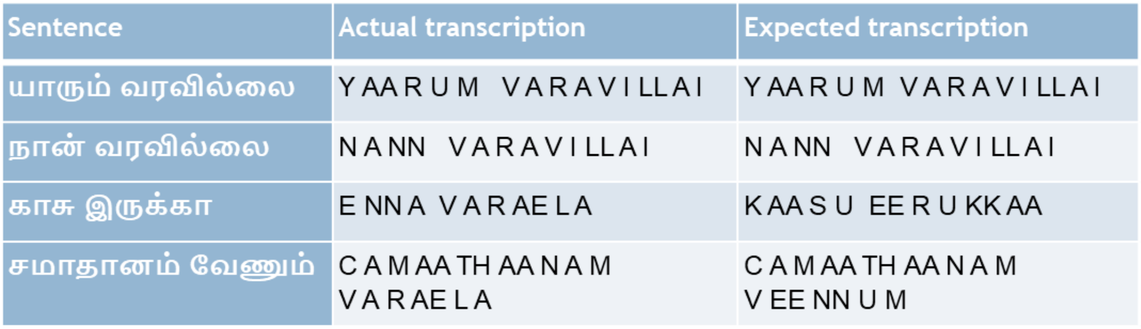
Figure 11 - female test data for female only dataset tri2b

5.1.3 Mixed (male and female) trained dataset with tri2b



Sentence	Actual transcription	Expected transcription
யாரும் வரவில்லை	VEENNUM VARAVILLAI	YAARUM VARAVILLAI
நான் வரவில்லை	NANN VARAVILLAI	NANN VARAVILLAI
காசு இருக்கா	ENNA VARAELA	KAASU EERUKKAA
சமாதானம் வேணும்	CAMAATHAANAMAAKA VARAELA	CAMAATHAANAM VEENNUM

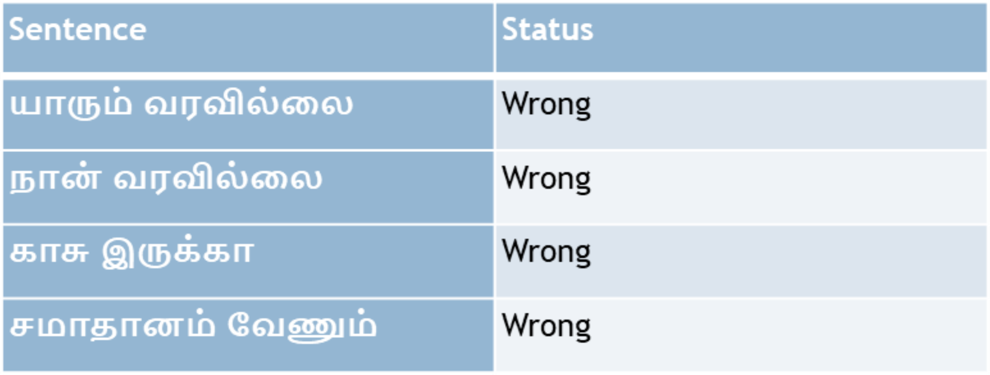
Figure 12 - male test data for mixed dataset tri2b



Sentence	Actual transcription	Expected transcription
யாரும் வரவில்லை	YAARUM VARAVILLAI	YAARUM VARAVILLAI
நான் வரவில்லை	NANN VARAVILLAI	NANN VARAVILLAI
காசு இருக்கா	ENNA VARAELA	KAASU EERUKKAA
சமாதானம் வேணும்	CAMAATHAANAM VARAELA	CAMAATHAANAM VEENNUM

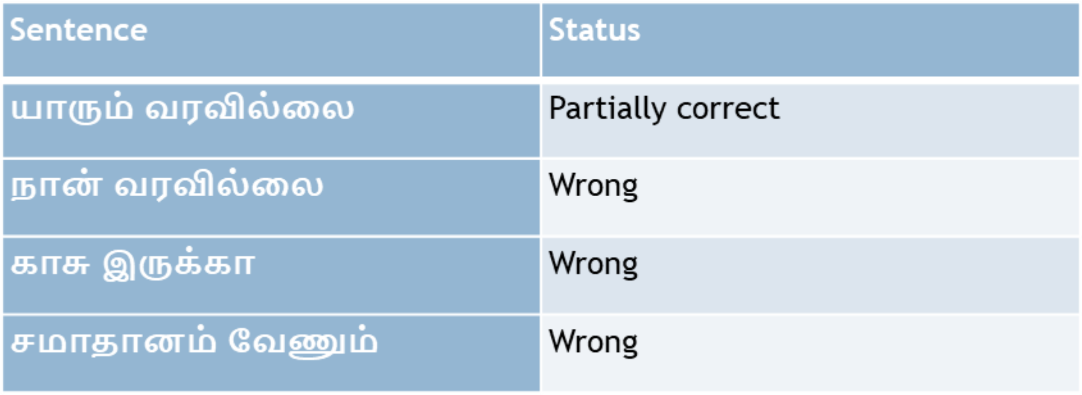
Figure 13 - female test data for mixed dataset tri2b

5.1.4 Male only trained dataset with tri3b



Sentence	Status
யாரும் வரவில்லை	Wrong
நான் வரவில்லை	Wrong
காசு இருக்கா	Wrong
சமாதானம் வேணும்	Wrong

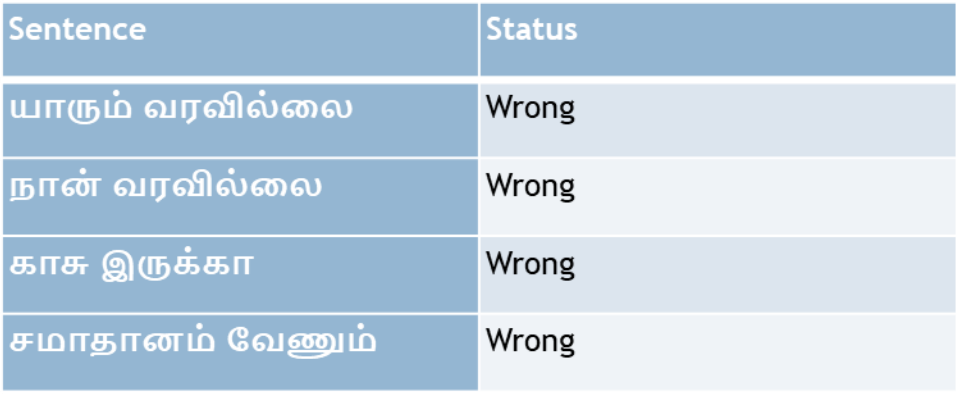
Figure 14 - male test data with male only dataset tri3b



Sentence	Status
யாரும் வரவில்லை	Partially correct
நான் வரவில்லை	Wrong
காசு இருக்கா	Wrong
சமாதானம் வேணும்	Wrong

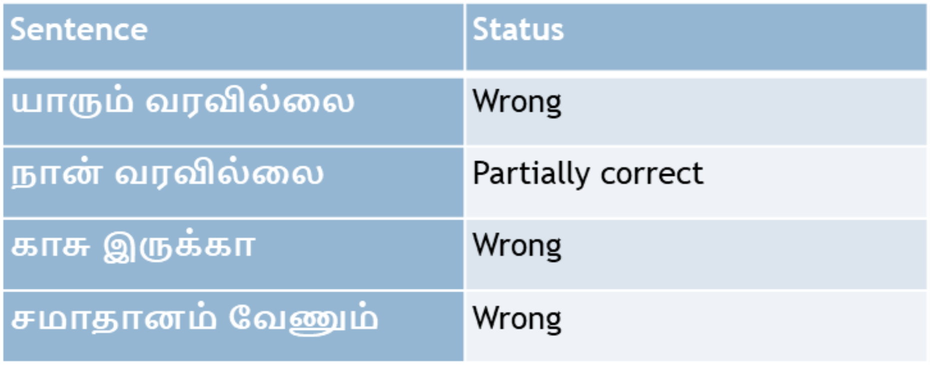
Figure 15 - female test data with male only dataset tri3b

5.1.5 Female only trained dataset with tri3b



Sentence	Status
யாரும் வரவில்லை	Wrong
நான் வரவில்லை	Wrong
காசு இருக்கா	Wrong
சமாதானம் வேணும்	Wrong

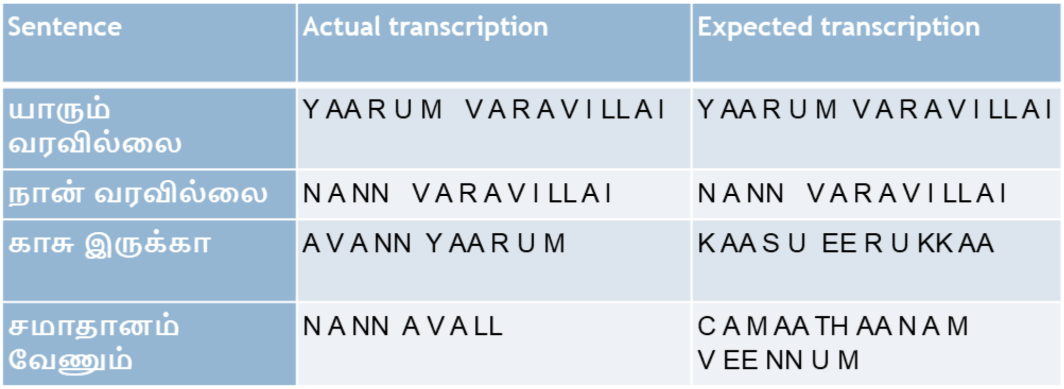
Figure 16 - male test data with female only dataset tri3b



Sentence	Status
யாரும் வரவில்லை	Wrong
நான் வரவில்லை	Partially correct
காசு இருக்கா	Wrong
சமாதானம் வேணும்	Wrong

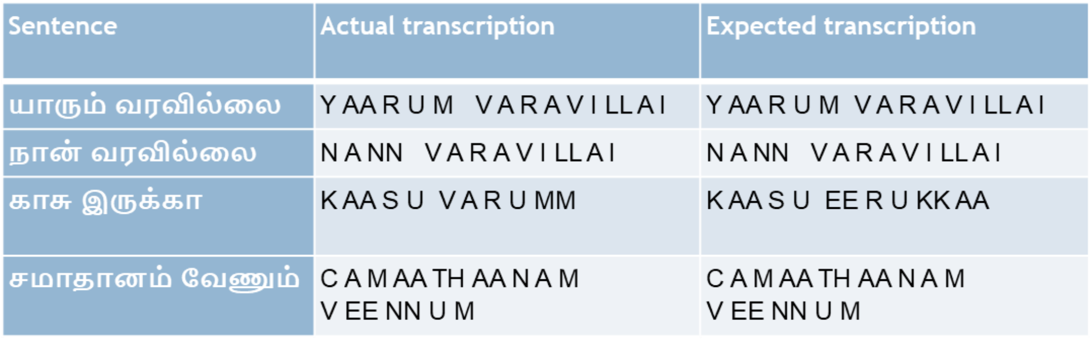
Figure 17 - female test data with female only dataset tri3b

5.1.6 Mixed (male and female) trained dataset with tri3b



Sentence	Actual transcription	Expected transcription
யாரும் வரவில்லை	YAARUM VARAVILLAI	YAARUM VARAVILLAI
நான் வரவில்லை	NANN VARAVILLAI	NANN VARAVILLAI
காசு இருக்கா	AVANN YAARUM	KAASU EERUKKAA
சமாதானம் வேணும்	NANN AVALL	CAMAATHAANAM VEENNUM

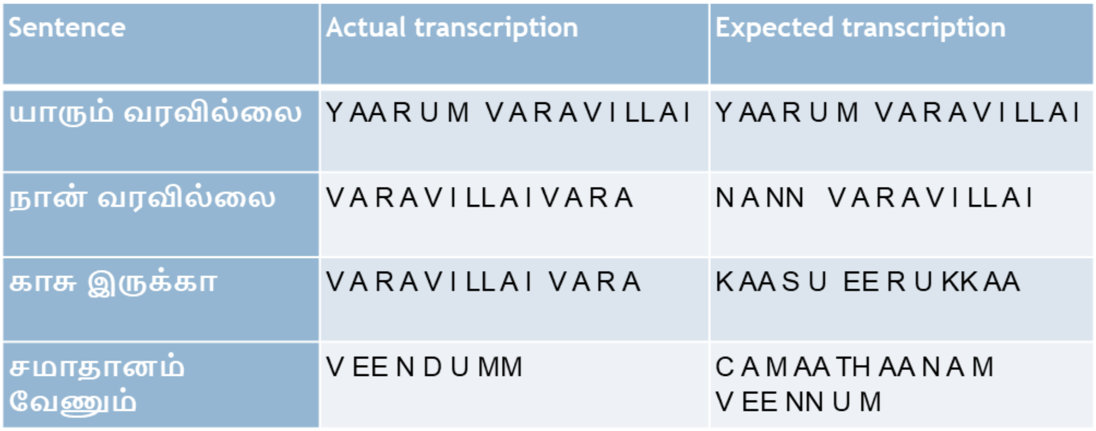
Figure 18 - male test data with mixed dataset tri3b



Sentence	Actual transcription	Expected transcription
யாரும் வரவில்லை	YAARUM VARAVILLAI	YAARUM VARAVILLAI
நான் வரவில்லை	NANN VARAVILLAI	NANN VARAVILLAI
காசு இருக்கா	KAASU VARUMM	KAASU EERUKKAA
சமாதானம் வேணும்	CAMAATHAANAM VEENNUM	CAMAATHAANAM VEENNUM

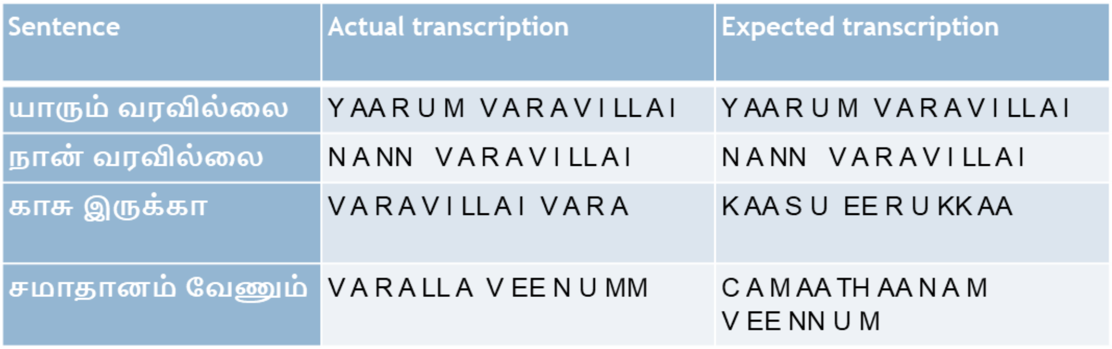
Figure 19 - female test data with mixed dataset tri3b

5.1.7 Mixed (male and female) trained dataset with tri4b



Sentence	Actual transcription	Expected transcription
யாரும் வரவில்லை	YAARUM VARAVILLAI	YAARUM VARAVILLAI
நான் வரவில்லை	VARAVILLAIVARA	NANN VARAVILLAI
காசு இருக்கா	VARAVILLAI VARA	KAASU EERUKKAA
சமாதானம் வேணும்	VEENDUMM	CAMAATHAANAM VEENNUM

Figure 20 - male test data with mixed dataset tri4b



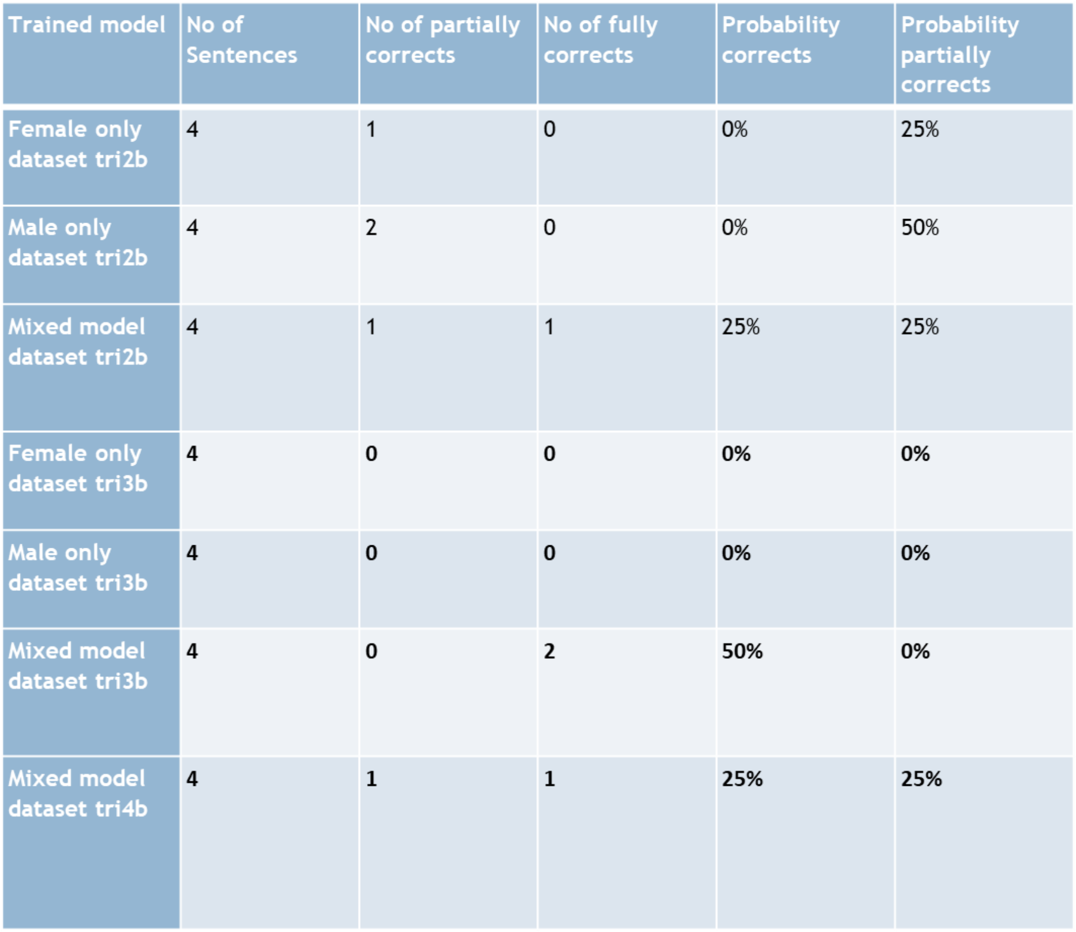
Sentence	Actual transcription	Expected transcription
யாரும் வரவில்லை	YAARUM VARAVILLAI	YAARUM VARAVILLAI
நான் வரவில்லை	NANN VARAVILLAI	NANN VARAVILLAI
காசு இருக்கா	VARAVILLAI VARA	KAASU EERUKKAA
சமாதானம் வேணும்	VARALLA VEENUMM	CAMAATHAANAM VEENNUM

Figure 21 - female test data with mixed dataset tri4b

5.2 Text data filtration

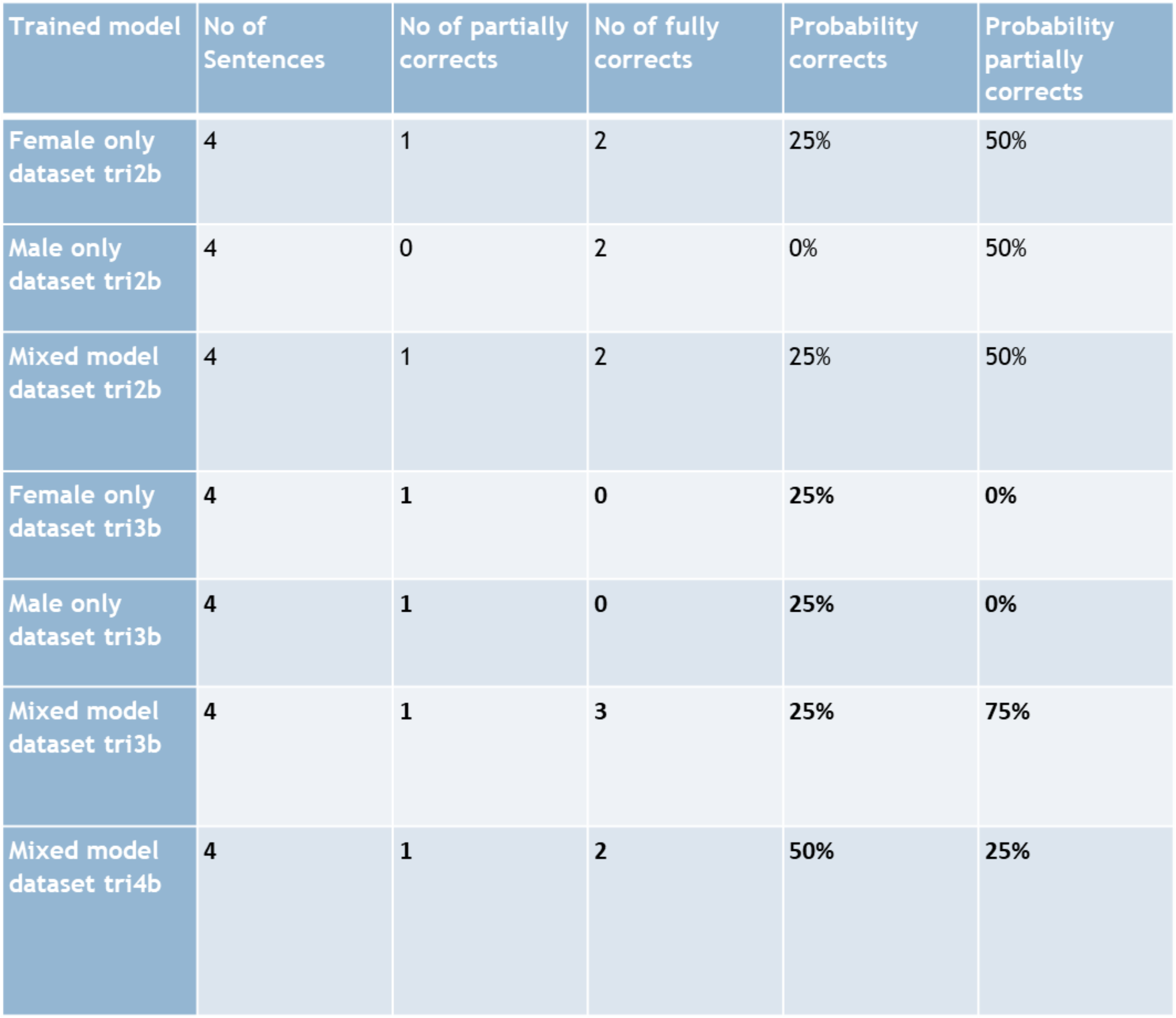
I have filtered and created following list to choose the best training dataset for male and female.

5.2.1 Male test data



Trained model	No of Sentences	No of partially corrects	No of fully corrects	Probability corrects	Probability partially corrects
Female only dataset tri2b	4	1	0	0%	25%
Male only dataset tri2b	4	2	0	0%	50%
Mixed model dataset tri2b	4	1	1	25%	25%
Female only dataset tri3b	4	0	0	0%	0%
Male only dataset tri3b	4	0	0	0%	0%
Mixed model dataset tri3b	4	0	2	50%	0%
Mixed model dataset tri4b	4	1	1	25%	25%

Figure 22 - male test data summary



Trained model	No of Sentences	No of partially corrects	No of fully corrects	Probability corrects	Probability partially corrects
Female only dataset tri2b	4	1	2	25%	50%
Male only dataset tri2b	4	0	2	0%	50%
Mixed model dataset tri2b	4	1	2	25%	50%
Female only dataset tri3b	4	1	0	25%	0%
Male only dataset tri3b	4	1	0	25%	0%
Mixed model dataset tri3b	4	1	3	25%	75%
Mixed model dataset tri4b	4	1	2	50%	25%

Figure 23 - female test data summary

5.3 Important factors which affects testing datasets

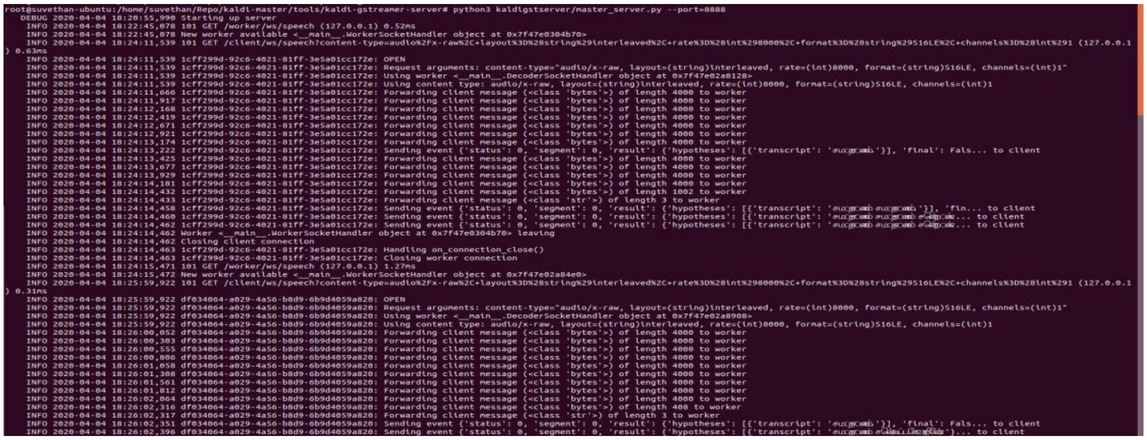
- Format of the voice data.
- Sampling rate of voice data.
- Number of voice data.
- Quality of the voice data.

5.4 Conclusion

From the result, I recommend mixed trained data model with tri2b for female testing when test data is less than 100, mixed trained data model with tri4b for female testing when test data is more than 100 and mixed trained data model with tri4b for male testing as it has lots of correctly identified sentences. This recommendation can be differ with the factors I mentioned above.

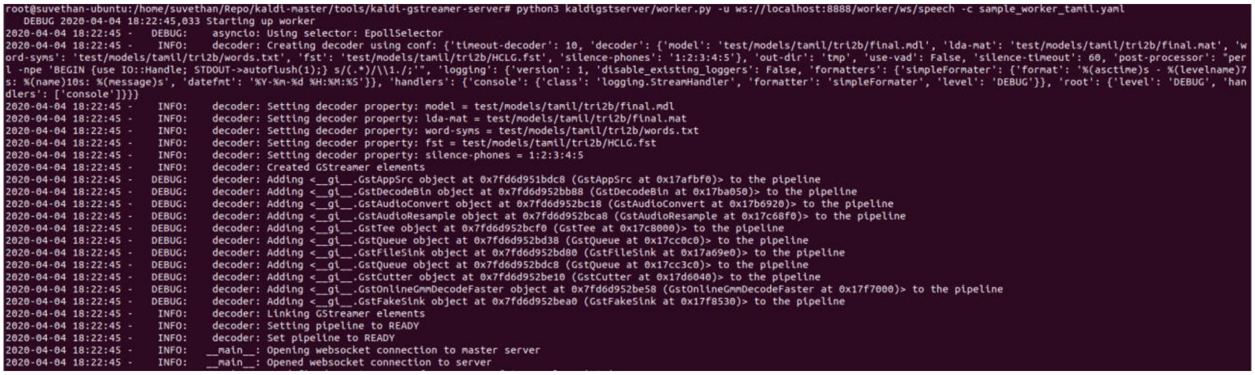
5.5 Sample screenshots of kaldi gstreamer

5.5.1 Master and worker server



```
root@suvethan-ubuntu:/home/suvethan/Repo/kaldi-master/tools/kaldi-gstreamer-server# python3 kaldigstserver/master_server.py --port=8888
   DEBUG 2020-04-04 18:20:55,990 Starting up server
    INFO 2020-04-04 18:22:45,078 101 GET /worker/ws/speech (127.0.0.1) 0.52ms
    INFO 2020-04-04 18:22:45,078 New worker available <__main__.WorkerSocketHandler object at 0x7f47e0304b70>
    INFO 2020-04-04 18:24:11,539 101 GET /client/ws/speech?content-type=audio%2Fx-raw%2C+layout%3D%28string%29interleaved%2C+rate%3D%28int%298000%2C+format%3D%28string%29S16LE%2C+channels%3D%28int%291 (127.0.0.1
) 0.63ms
    INFO 2020-04-04 18:24:11,539 1cff299d-92c6-4021-81ff-3e5a01cc172e: OPEN
    INFO 2020-04-04 18:24:11,539 1cff299d-92c6-4021-81ff-3e5a01cc172e: Request arguments: content-type="audio/x-raw, layout=(string)interleaved, rate=(int)8000, format=(string)S16LE, channels=(int)1"
    INFO 2020-04-04 18:24:11,539 1cff299d-92c6-4021-81ff-3e5a01cc172e: Using worker <__main__.DecoderSocketHandler object at 0x7f47e02a8128>
    INFO 2020-04-04 18:24:11,539 1cff299d-92c6-4021-81ff-3e5a01cc172e: Using content type: audio/x-raw, layout=(string)interleaved, rate=(int)8000, format=(string)S16LE, channels=(int)1
    INFO 2020-04-04 18:24:11,666 1cff299d-92c6-4021-81ff-3e5a01cc172e: Forwarding client message (<class 'bytes'>) of length 4000 to worker
    INFO 2020-04-04 18:24:11,917 1cff299d-92c6-4021-81ff-3e5a01cc172e: Forwarding client message (<class 'bytes'>) of length 4000 to worker
    INFO 2020-04-04 18:24:12,168 1cff299d-92c6-4021-81ff-3e5a01cc172e: Forwarding client message (<class 'bytes'>) of length 4000 to worker
    INFO 2020-04-04 18:24:12,419 1cff299d-92c6-4021-81ff-3e5a01cc172e: Forwarding client message (<class 'bytes'>) of length 4000 to worker
    INFO 2020-04-04 18:24:12,671 1cff299d-92c6-4021-81ff-3e5a01cc172e: Forwarding client message (<class 'bytes'>) of length 4000 to worker
    INFO 2020-04-04 18:24:12,921 1cff299d-92c6-4021-81ff-3e5a01cc172e: Forwarding client message (<class 'bytes'>) of length 4000 to worker
    INFO 2020-04-04 18:24:13,174 1cff299d-92c6-4021-81ff-3e5a01cc172e: Forwarding client message (<class 'bytes'>) of length 4000 to worker
    INFO 2020-04-04 18:24:13,222 1cff299d-92c6-4021-81ff-3e5a01cc172e: Sending event {'status': 0, 'segment': 0, 'result': {'hypotheses': [{'transcript': 'சமூகம்'}], 'final': Fals... to client
    INFO 2020-04-04 18:24:13,425 1cff299d-92c6-4021-81ff-3e5a01cc172e: Forwarding client message (<class 'bytes'>) of length 4000 to worker
    INFO 2020-04-04 18:24:13,677 1cff299d-92c6-4021-81ff-3e5a01cc172e: Forwarding client message (<class 'bytes'>) of length 4000 to worker
    INFO 2020-04-04 18:24:13,929 1cff299d-92c6-4021-81ff-3e5a01cc172e: Forwarding client message (<class 'bytes'>) of length 4000 to worker
    INFO 2020-04-04 18:24:14,181 1cff299d-92c6-4021-81ff-3e5a01cc172e: Forwarding client message (<class 'bytes'>) of length 4000 to worker
    INFO 2020-04-04 18:24:14,432 1cff299d-92c6-4021-81ff-3e5a01cc172e: Forwarding client message (<class 'bytes'>) of length 1002 to worker
    INFO 2020-04-04 18:24:14,433 1cff299d-92c6-4021-81ff-3e5a01cc172e: Forwarding client message (<class 'str'>) of length 3 to worker
    INFO 2020-04-04 18:24:14,458 1cff299d-92c6-4021-81ff-3e5a01cc172e: Sending event {'status': 0, 'segment': 0, 'result': {'hypotheses': [{'transcript': 'சமூகம் சமூகம்'}], 'fin... to client
    INFO 2020-04-04 18:24:14,460 1cff299d-92c6-4021-81ff-3e5a01cc172e: Sending event {'status': 0, 'segment': 0, 'result': {'hypotheses': [{'transcript': 'சமூகம் சமூகம் [illegible]... to client
    INFO 2020-04-04 18:24:14,462 1cff299d-92c6-4021-81ff-3e5a01cc172e: Sending event {'status': 0, 'segment': 0, 'result': {'hypotheses': [{'transcript': 'சமூகம் சமூகம் [illegible]... to client
    INFO 2020-04-04 18:24:14,462 Worker <__main__.WorkerSocketHandler object at 0x7f47e0304b70> leaving
    INFO 2020-04-04 18:24:14,462 Closing client connection
    INFO 2020-04-04 18:24:14,463 1cff299d-92c6-4021-81ff-3e5a01cc172e: Handling on_connection_close()
    INFO 2020-04-04 18:24:14,463 1cff299d-92c6-4021-81ff-3e5a01cc172e: Closing worker connection
    INFO 2020-04-04 18:24:15,471 101 GET /worker/ws/speech (127.0.0.1) 1.27ms
    INFO 2020-04-04 18:24:15,472 New worker available <__main__.WorkerSocketHandler object at 0x7f47e02a84e0>
    INFO 2020-04-04 18:25:59,922 101 GET /client/ws/speech?content-type=audio%2Fx-raw%2C+layout%3D%28string%29interleaved%2C+rate%3D%28int%298000%2C+format%3D%28string%29S16LE%2C+channels%3D%28int%291 (127.0.0.1
) 0.31ms
    INFO 2020-04-04 18:25:59,922 df034064-a029-4a56-b8d9-6b9d4059a820: OPEN
    INFO 2020-04-04 18:25:59,922 df034064-a029-4a56-b8d9-6b9d4059a820: Request arguments: content-type="audio/x-raw, layout=(string)interleaved, rate=(int)8000, format=(string)S16LE, channels=(int)1"
    INFO 2020-04-04 18:25:59,922 df034064-a029-4a56-b8d9-6b9d4059a820: Using worker <__main__.DecoderSocketHandler object at 0x7f47e02a8908>
    INFO 2020-04-04 18:25:59,922 df034064-a029-4a56-b8d9-6b9d4059a820: Using content type: audio/x-raw, layout=(string)interleaved, rate=(int)8000, format=(string)S16LE, channels=(int)1
    INFO 2020-04-04 18:26:00,052 df034064-a029-4a56-b8d9-6b9d4059a820: Forwarding client message (<class 'bytes'>) of length 4000 to worker
    INFO 2020-04-04 18:26:00,303 df034064-a029-4a56-b8d9-6b9d4059a820: Forwarding client message (<class 'bytes'>) of length 4000 to worker
    INFO 2020-04-04 18:26:00,555 df034064-a029-4a56-b8d9-6b9d4059a820: Forwarding client message (<class 'bytes'>) of length 4000 to worker
    INFO 2020-04-04 18:26:00,806 df034064-a029-4a56-b8d9-6b9d4059a820: Forwarding client message (<class 'bytes'>) of length 4000 to worker
    INFO 2020-04-04 18:26:01,058 df034064-a029-4a56-b8d9-6b9d4059a820: Forwarding client message (<class 'bytes'>) of length 4000 to worker
    INFO 2020-04-04 18:26:01,308 df034064-a029-4a56-b8d9-6b9d4059a820: Forwarding client message (<class 'bytes'>) of length 4000 to worker
    INFO 2020-04-04 18:26:01,561 df034064-a029-4a56-b8d9-6b9d4059a820: Forwarding client message (<class 'bytes'>) of length 4000 to worker
    INFO 2020-04-04 18:26:01,812 df034064-a029-4a56-b8d9-6b9d4059a820: Forwarding client message (<class 'bytes'>) of length 4000 to worker
    INFO 2020-04-04 18:26:02,064 df034064-a029-4a56-b8d9-6b9d4059a820: Forwarding client message (<class 'bytes'>) of length 4000 to worker
    INFO 2020-04-04 18:26:02,316 df034064-a029-4a56-b8d9-6b9d4059a820: Forwarding client message (<class 'bytes'>) of length 408 to worker
    INFO 2020-04-04 18:26:02,317 df034064-a029-4a56-b8d9-6b9d4059a820: Forwarding client message (<class 'str'>) of length 3 to worker
    INFO 2020-04-04 18:26:02,351 df034064-a029-4a56-b8d9-6b9d4059a820: Sending event {'status': 0, 'segment': 0, 'result': {'hypotheses': [{'transcript': 'சமூகம்'}], 'final': Fals... to client
    INFO 2020-04-04 18:26:02,396 df034064-a029-4a56-b8d9-6b9d4059a820: Sending event {'status': 0, 'segment': 0, 'result': {'hypotheses': [{'transcript': 'சமூகம் [illegible]'}... to client
```

Figure 25 - Master server

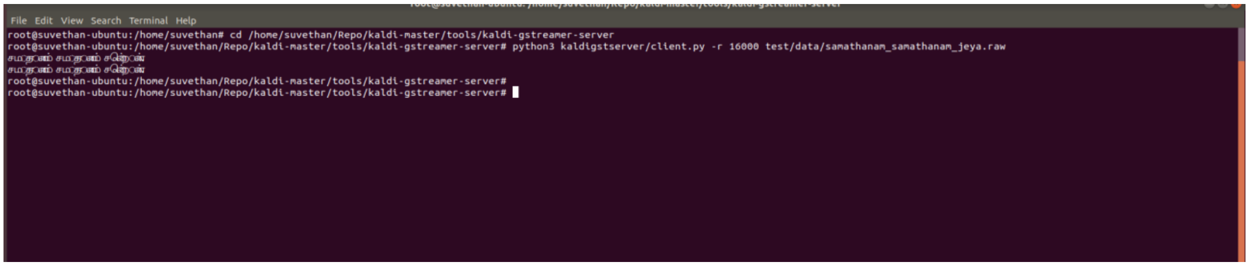


```
root@suvethan-ubuntu:/home/suvethan/Repo/kaldi-master/tools/kaldi-gstreamer-server# python3 kaldigstserver/worker.py -u ws://localhost:8888/worker/ws/speech -c sample_worker_tamil.yaml
   DEBUG 2020-04-04 18:22:45,033 Starting up worker
2020-04-04 18:22:45 -   DEBUG:    asyncio: Using selector: EpollSelector
2020-04-04 18:22:45 -    INFO:    decoder: Creating decoder using conf: {'timeout-decoder': 10, 'decoder': {'model': 'test/models/tamil/tri2b/final.mdl', 'lda-mat': 'test/models/tamil/tri2b/final.mat', 'w
ord-syms': 'test/models/tamil/tri2b/words.txt', 'fst': 'test/models/tamil/tri2b/HCLG.fst', 'silence-phones': '1:2:3:4:5'}, 'out-dir': 'tmp', 'use-vad': False, 'silence-timeout': 60, 'post-processor': "per
l -npe 'BEGIN {use IO::Handle; STDOUT->autoflush(1);} s/(.*)/\\1./;'", 'logging': {'version': 1, 'disable_existing_loggers': False, 'formatters': {'simpleFormater': {'format': '%(asctime)s - %(levelname)7
s: %(name)10s: %(message)s', 'datefmt': '%Y-%m-%d %H:%M:%S'}}, 'handlers': {'console': {'class': 'logging.StreamHandler', 'formatter': 'simpleFormater', 'level': 'DEBUG'}}, 'root': {'level': 'DEBUG', 'han
dlers': ['console']}}}
2020-04-04 18:22:45 -    INFO:    decoder: Setting decoder property: model = test/models/tamil/tri2b/final.mdl
2020-04-04 18:22:45 -    INFO:    decoder: Setting decoder property: lda-mat = test/models/tamil/tri2b/final.mat
2020-04-04 18:22:45 -    INFO:    decoder: Setting decoder property: word-syms = test/models/tamil/tri2b/words.txt
2020-04-04 18:22:45 -    INFO:    decoder: Setting decoder property: fst = test/models/tamil/tri2b/HCLG.fst
2020-04-04 18:22:45 -    INFO:    decoder: Setting decoder property: silence-phones = 1:2:3:4:5
2020-04-04 18:22:45 -    INFO:    decoder: Created GStreamer elements
2020-04-04 18:22:45 -   DEBUG:    decoder: Adding <__gi__.GstAppSrc object at 0x7fd6d951bdc8 (GstAppSrc at 0x17afbf0)> to the pipeline
2020-04-04 18:22:45 -   DEBUG:    decoder: Adding <__gi__.GstDecodeBin object at 0x7fd6d952bb88 (GstDecodeBin at 0x17ba050)> to the pipeline
2020-04-04 18:22:45 -   DEBUG:    decoder: Adding <__gi__.GstAudioConvert object at 0x7fd6d952bc18 (GstAudioConvert at 0x17b6920)> to the pipeline
2020-04-04 18:22:45 -   DEBUG:    decoder: Adding <__gi__.GstAudioResample object at 0x7fd6d952bca8 (GstAudioResample at 0x17c68f0)> to the pipeline
2020-04-04 18:22:45 -   DEBUG:    decoder: Adding <__gi__.GstTee object at 0x7fd6d952bcf0 (GstTee at 0x17c8000)> to the pipeline
2020-04-04 18:22:45 -   DEBUG:    decoder: Adding <__gi__.GstQueue object at 0x7fd6d952bd38 (GstQueue at 0x17cc0c0)> to the pipeline
2020-04-04 18:22:45 -   DEBUG:    decoder: Adding <__gi__.GstFileSink object at 0x7fd6d952bd80 (GstFileSink at 0x17a69e0)> to the pipeline
2020-04-04 18:22:45 -   DEBUG:    decoder: Adding <__gi__.GstQueue object at 0x7fd6d952bdc8 (GstQueue at 0x17cc3c0)> to the pipeline
2020-04-04 18:22:45 -   DEBUG:    decoder: Adding <__gi__.GstCutter object at 0x7fd6d952be10 (GstCutter at 0x17d6040)> to the pipeline
2020-04-04 18:22:45 -   DEBUG:    decoder: Adding <__gi__.GstOnlineGmmDecodeFaster object at 0x7fd6d952be58 (GstOnlineGmmDecodeFaster at 0x17f7000)> to the pipeline
2020-04-04 18:22:45 -   DEBUG:    decoder: Adding <__gi__.GstFakeSink object at 0x7fd6d952bea0 (GstFakeSink at 0x17f8530)> to the pipeline
2020-04-04 18:22:45 -    INFO:    decoder: Linking GStreamer elements
2020-04-04 18:22:45 -    INFO:    decoder: Setting pipeline to READY
2020-04-04 18:22:45 -    INFO:    decoder: Set pipeline to READY
2020-04-04 18:22:45 -    INFO:  __main__: Opening websocket connection to master server
2020-04-04 18:22:45 -    INFO:  __main__: Opened websocket connection to server
```

Figure 24 - Worker server

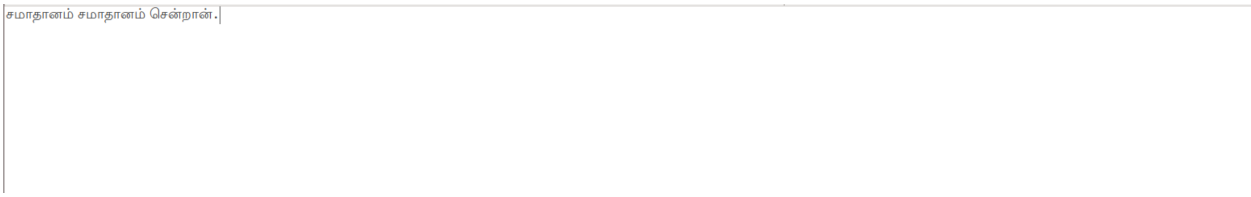
5.5.2 Input and output of the speech

Client.py file has been used to pass recorded speech in raw format which communicates with worker and returns text for the speech provided. Following are the screenshots of those with approximate match and exact match.


root@suvethan-ubuntu: /home/suvethan/Repo/kaldi-master/tools/kaldi-gstreamer-server
File Edit View Search Terminal Help

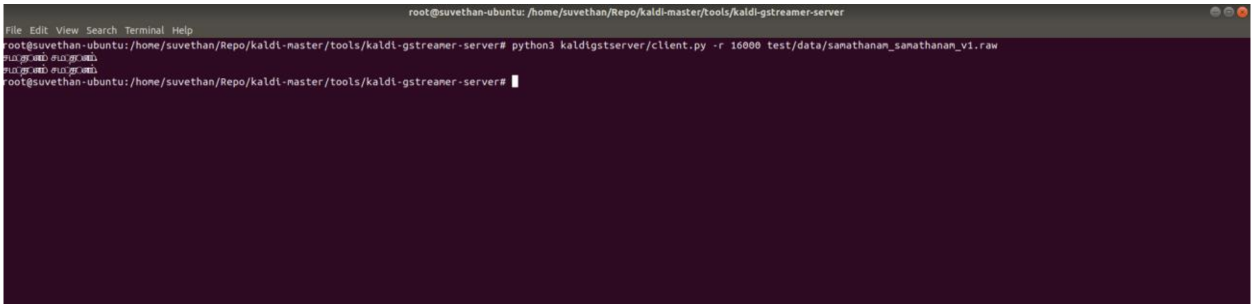
```
root@suvethan-ubuntu:/home/suvethan# cd /home/suvethan/Repo/kaldi-master/tools/kaldi-gstreamer-server
root@suvethan-ubuntu:/home/suvethan/Repo/kaldi-master/tools/kaldi-gstreamer-server# python3 kaldigstserver/client.py -r 16000 test/data/samathanam_samathanam_jeya.raw
சமாதானம் சமாதானம் சென்றான்
சமாதானம் சமாதானம் சென்றான்
root@suvethan-ubuntu:/home/suvethan/Repo/kaldi-master/tools/kaldi-gstreamer-server#
root@suvethan-ubuntu:/home/suvethan/Repo/kaldi-master/tools/kaldi-gstreamer-server# 
```

Figure 27 - Client file with approximate match



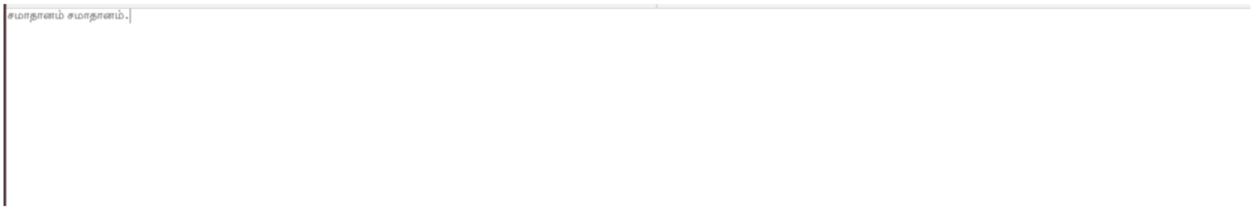
சமாதானம் சமாதானம் சென்றான்.

Figure 26 - Approximate match copied to text file since terminal does not decode Tamil properly


root@suvethan-ubuntu: /home/suvethan/Repo/kaldi-master/tools/kaldi-gstreamer-server
File Edit View Search Terminal Help

```
root@suvethan-ubuntu:/home/suvethan/Repo/kaldi-master/tools/kaldi-gstreamer-server# python3 kaldigstserver/client.py -r 16000 test/data/samathanam_samathanam_v1.raw
சமாதானம் சமாதானம்.
சமாதானம் சமாதானம்.
root@suvethan-ubuntu:/home/suvethan/Repo/kaldi-master/tools/kaldi-gstreamer-server# 
```

Figure 28 - Client file with exact match



சமாதானம் சமாதானம்.

Figure 29 - Exact match copied to text file since terminal does not decode Tamil properly

List of References

1. Software Localization Is Key to Success In International Markets. In: Net Transl. https://www.net-translators.com/knowledge-center/articles/localization-the-key-to-success-in-the-international-market/. Accessed 16 Jul 2019

2. (2016) The Right Localization Strategy for Your Business. In: PhraseApp Blog. https://phraseapp.com/blog/posts/how-important-is-localization-for-your-business/. Accessed 16 Jul 2019

3. (2017) Software Localization - Why It's Important for your Business. In: Transl. Hum. Blog. https://www.translatebyhumans.com/blog/understanding-software-localization/. Accessed 16 Jul 2019

4. Speech-to-speech translation. https://google-research.github.io/lingvo-lab/translatotron/. Accessed 16 Jul 2019

5. Jia Y, Weiss RJ, Biadsy F, et al (2019) Direct speech-to-speech translation with a sequence-to-sequence model. ArXiv190406037 Cs Eess

6. Speech Recognition System for Sinhala. In: UCSC. https://ucsc.cmb.ac.lk/speech-recognition-system-sinhala/. Accessed 16 Jul 2019

7. Poornakala J, Maheshwari A, Student PG (2007) Automatic Speech-Speech Translation System from English to Tamil Language. 5:5

8. R. A. Ganesan and H. R. ShivaKumar, "Online speech translation system for Tamil."

9. "Practical Cryptography." [Online]. Available: http://www.practicalcryptography.com/miscellaneous/machine-learning/guide-mel-frequency-cepstral-coefficients-mfccs/. Accessed: 24-Jan-2020.

10. "Kaldi: Data preparation." [Online]. Available: http://kaldi-asr.org/doc/data_prep.html. [Accessed: 24-Jan-2020].