5

AGRICULTURAL SPOKEN DIALOGUE SYSTEM

CONTENTS

OBJECTIVE OF THE CHAPTER

In this chapter we explain the architecture of proposed agricultural based spoken dialogue system. The contents of this chapter provide a blueprint of the target system that is the goal of this entire work. Once we fix on a good architecture for the implementation of an efficient spoken dialogue system, then we can move on to inspect each of the individual elements of the architecture one by one and discuss about what algorithms and techniques are suitable with regards to the Bodo language.

5.1 PROPOSED SYSTEM ARCHITECTURE

The proposed system mainly consist of Bluetooth enabled mobile for transferring utterance, Bluetooth based PC for receiving the signal, Asterisk [57] server for IVR module, Mysql sever to store the price , fertilizer and weather related information , Php-agi script to design the call flow. All these installation and configuration is done in Ubuntu 12.04. Open source speech recognition toolkit Sphinx and HTK is used to build the speech recognizer module and for analysing the collected corpus Pratt, wave-surfer and MatLAB are used. Following is the basic block of the proposed system.

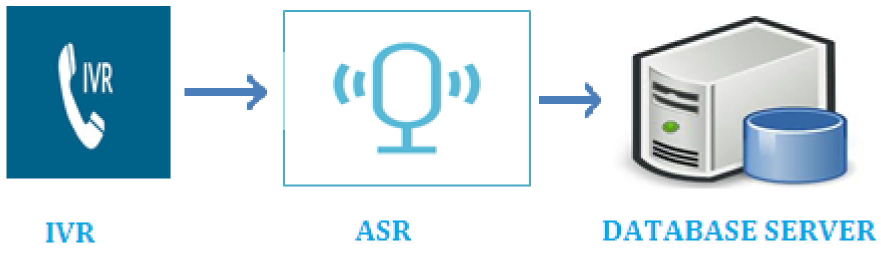


Figure 5.1 Block Diagram of Proposed System

a. IVR: IVR stands for Interactive Voice Response. It is a technology that allows computer to interact with humans through the use of voice and DTMF tones input via keypad. IVR systems can respond with prerecorded or dynamically generated audio to further direct users on how to proceed. IVR applications can be used to control almost any function where the interface can be broken down into a series of simple interactions. Asterisk AGI (Asterisk Gateway Interface) is being used for making IVR applications able to run in the voice-server. The interactive voice response system consists of a computer and application software running on the computer. It is connected with mobile by using Bluetooth as a medium of communication. The IVR module record the speech uttered by the user and store it into the computer. It also interacts with the computer to obtain necessary information.

Asterisk AGI is a software interface and communications protocol that allows an external, user-written program, launched from the Asterisk dial plan via pipes to control telephony operations on its associated control and voice channels. Also, there is a special class of PHP call PHPAGI , with the help of which, we can build IVR applications, which in turn can host other applications like speech recognition etc.

b. ASR ENGINE: The basic task of Automatic Speech Recognition (ASR) is to derive a sequence of words from a stream of acoustic information. Automatic recognition of telephonic voice queries requires a robust back-end ASR system. HTK [48], an open Source Speech Recognition Engine is used here which typically consists of Speech Feature Extraction module, Acoustic Model, Language Model, Decoder. In this module the input signal is captured and further analyzed. This is the very important module of this system. This module extracts different temporal features like Zero Crossing Rate (ZCR), Short Time Energy (STE) and Spectral features like formant analysis etc. The module perform the voiced, unvoiced, silence segmentation, voice activity detection using some predefined Knowledge Base (KB) and gives some decision about the correctness of the signal. This decides whether users need to re-record speech signal or not. If re-recording is not required then recorded speech signal go through ASR engine for decoding the signal.

c. DATABASE SERVER: This block contains the price, fertilizer and weather information about the crops. Based on the input supplied by ASR engine it retrieves the information available in MySQL database and supply it to the user using mobile network.

5.2 FUNCTIONALITY OF THE IVR BLOCK

In development of telephonic ASR system, Asterisk is used as an open source IVR Server which is running on Linux. It supports VoIP protocols like SIP, H.323;

interfaces with PSTN Channels, supports various PCI Cards, and also open source drivers and libraries are available [58]. There is a series of configuration files which controls Asterisk. It uses several directories on a Linux system to manage its various aspects, such as voicemail recordings, voice prompts, and configuration files. The necessary directories are:

- /etc/asterisk - Contains all of asterisk configuration files and logic information.
- /usr/lib/asterisk/modules - Contains all of asterisk’s loadable modules, operating asterisk functionality.
- /var/lib/asterisk/sounds - Contains all of asterisk’s sound files for playback and pre-loaded applications.
- /var/lib/asterisk/agi-bin - Contains all of asterisk’s AGI scripts and AGI logic.

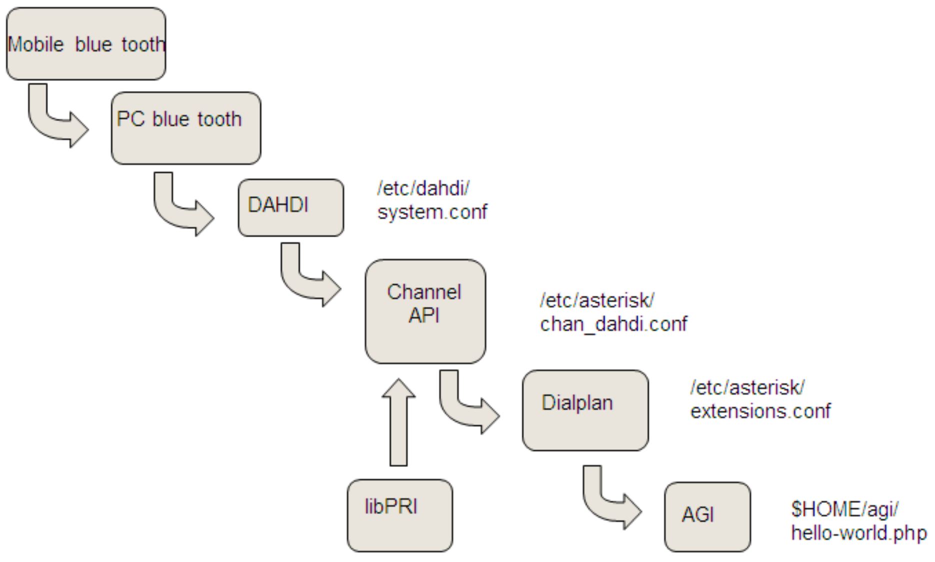
Asterisk Channel Driver to allow Bluetooth Cell/Mobile Phones to be used as FXO devices and Bluetooth Headsets as FXS devices. Asterisk is connected with network by Using Bluetooth enabled mobile network. In Asterisk the Chan-mobile supports FXS and FXO station interfaces for connecting wireless lines through a PC. The zttool program can be used to check the status of installed hardware in Asterisk [59]. The configuration information is stored in the following files:

- /etc/zaptel.conf: Configuration of hardware Interfaces
- /etc/asterisk/zapata.conf: Asterisk configuration to use hardware interfaces

The extensions.conf is the configuration file of Asterisk. It contains the "dial plan" of Asterisk, the master plan of control or execution flow for all of its operations. It controls, handled and routed the incoming and outgoing calls. Once the registered extension number is dialed the call is received at the server end. After receiving the call, asterisk server executes the dial plan and plays an audio file requiring the client to speak any input as prompted. The server recorded the user’s prompt in a wav file format for 3 sec and stored in a fixed location. In the next step of dial plan, a Php-Agi script is invoked with takes the audio file as input and process the call flow for further implementation. Finally the stored file is used as input for recognition in offline recognition mode, given in the shell script .The recognized word is stored in a text file (i.e. output.txt). To convert the call flow into dialogue manger we have used PHP-AGI script which is integrated with extention.conf file. Once the call landed to extention.conf file the script will activate and start working according to call flow

5.2.1 INTEGRATION OF DIALPLAN WITH RECOGNIZER

The configuration file "extensions.conf" contains the "dial plan" of Asterisk, the master plan of control or execution flow for all of its operations. Asterisk dial plan scan be developed for basic simple application to complex application for billing solutions, call centre solutions, IVR or any other custom application. Most of the computer telephony interaction systems like above can be auto mated by integrating ASR system with the asterisk through AGI programming along with rules definition in configuration file. Dial plan controls how incoming and outgoing calls are handled and routed**.** This file configures the behaviour and control of all connections through your PBX [60]. The content of "extensions.conf" is organized in sections, which can be either for static settings and definitions or for executable dial plan components in which case they are referred to as contexts. The settings sections are general and global and the names of contexts are entirely defined by the system administrator. Most of the computer telephony interaction systems can be automated by integrating ASR system with the asterisk through AGI programming along with rules definition in dial plan [61].



Mobile blue tooth
PC blue tooth
DAHDI
/etc/dahdi/ system.conf
Channel API
/etc/asterisk/ chan_dahdi.conf
Dialplan
/etc/asterisk/ extensions.conf
libPRI
AGI
$HOME/agi/ hello-world.php

Figure 5.2: Signal Flow in Asterisk

Configuration file for that is given below:

extensions.conf - The dialplan is divided in sections called contexts with more than one extension. Every extension has a priority and an application. With the help of contexts we can organize our dialplan. Our implemented dial plan:

```
[general]
static = yes
write protect = no
auto fall through = yes
clear global vars = no
priority jumping = no
[globals]
trunk_1 = Zap/g1
[voicemenu- custom]
Include =default
Comment =Welcome
alias_exten=1000
exten =10,1,Answer
exten =10,2,Playback(/var/lib/asterisk/message/input.wav)
exten =10, 3, Record (/var/lib/asterisk/scripts/rec/test: wav, 0, 3)
exten = 10, 4, AGI (recognize.agi)
extend =10, 5, ReadFile (ext_no=/var/lib/asterisk/scripts/rec/exten.txt, 3)
exten =10, 6, ReadFile (label_no=/var/lib/asterisk/scripts/rec/label.txt, 3)
extend =10, 7, GoTo (${ext_no}, ${label_no})
```

After registering, an appropriate extension number is dialled and a call is received at the server end. Asterisk server, upon receiving the call, executes the dial plan written specific to the called extension number and plays an audio file requiring the client to speak any input as prompted by the server which is further recorded in a wavfile format for 3 sec (specific to application) and stored in a fixed location. In the next step of dial plan, a shell script is invoked which externally provides output to be further used by Asterisk server as input to process user defined logic implementation. Finally the stored file is used as input for recognition in offline recognition mode, given in the shell script .The recognized word is stored in a text file (i.e. output.txt). The recognized word is processed and sent back to Asterisk server as input, which is stored in a variable and can be used for further processing according to the user or application requirements [63][64][65].

5.2.2 CALL FLOW

An Interactive Voice Response system call flow identifies all the resources available to callers, such as recorded announcements or prompts, access to self–service option, or live support. It also defines the specific and default actions that are will perform for each option selected. A call flow is a road map to how callers will be served from beginning to end. The objective of the call flow is to recognize the commodity name in the isolated word fashion; based on the recognized commodity name the respective information updated for that commodity is played out to the caller in the form of dialog. However, due to several reasons the recognition accuracy is not one hundred per cent. Therefore the call flow needs to take care of these errors into account and overcome the problems due to wrong recognition by ASR. Each node of the call flow relies on yes/no response from the user for prompted dialogs based on the recognized words to take the call flow decisions. Since, yes/no word recognition accuracy is good and it is reliable to take decisions at each of the nodal stages of the call flow.

5.3 CORPUS PREPARATION

As per requirement of our system we mainly required three types of data, commodity name, digit and yes/no. Our speech corpus consist of 31(25+4+2) unique Bodo words, out of which 25 commodity name, 4 digit (0, 1, 2, 3) and Yes/No word. There for we identify 100 different Bodo speaking people for different location and collect data in quite silent environment. We take 70 males speaker and 30 female speaker of different age group for data recording purpose. Finally we have collected 7171 words from 100 speakers which we are going to use speech recognizer. The data is recorded with the help of a unidirectional microphone using a recording tool wave surfer in .wav format. The speech data was digitized at 16 bits/sample at a sampling rate of 44 kHz. The following points were taken into account while collecting the data:

- **a.** speech data covering all dialects of BTAD is collected,
- **b.** multiple handsets and mobile services providers are used while collecting the data to capture possible variability,

c. male to female ratio of 1:1 is observed, and

d. The collected speech data is manually transcribed.

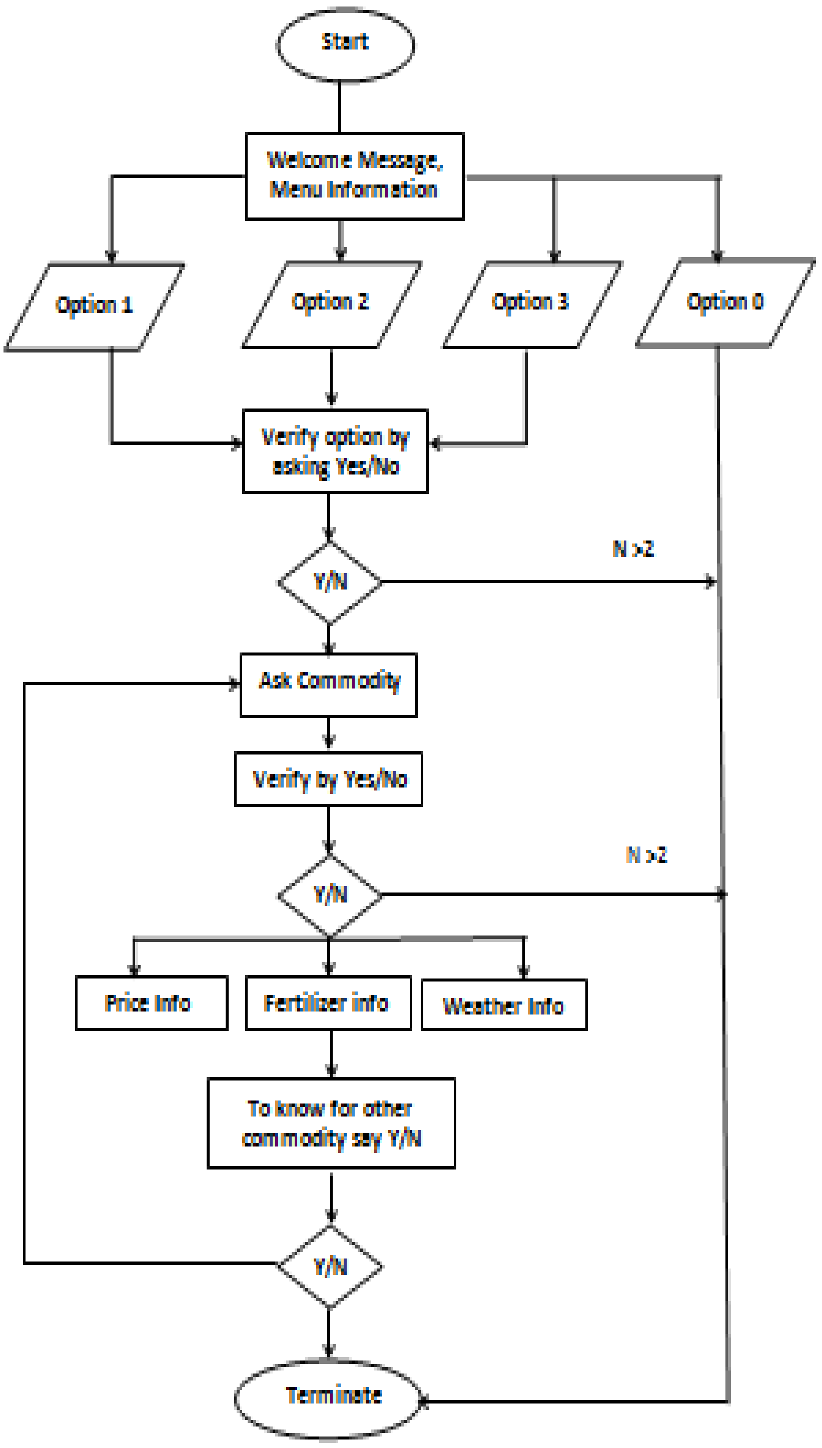


Figure: 5.3 Call Flow of the Proposed System

Following steps gives the complete IVRS flow of our proposed agro information system:

System: Welcome to Agricultural information system. To know the price of a commodity Say 1, to know about the fertilizer of the commodity say 2, to know the weather information about the commodity please say 3 and to end up the call please say Zero.

User: one

System: You have chosen one. To confirm it say yes or to deney say no.

User: yes

System: What is the commodity name for which you want to know the price?

User: Potato

System: You have said Potato. To confirm it say yes or to deny say no.

User: Yes.

System: The whole sale price is 1000 per quintal and the retail price is 12 per kg.

System: To know the price of another commodity please says yes or to go to main menu say no.

User: No

System: Welcome to Agricultural information system. To know the price of a commodity Say 1, to know about the fertilizer of the commodity say 2, to know the weather information about the commodity please say 3 and to end up the call please say Zero.

User: Two

System: You have chosen two. To confirm it say yes or to deny say no.

User: yes

System: What is the commodity name for which you want to know fertilizer information?

User: Potato

System: You have said Potato. To confirm it say yes or to deny say no.

User: Yes.

System: …………………………………………………………………..

System: To know the fertilizer information of another commodity please says yes or to go to main menu say no.

User: No

System: Welcome to Agricultural information system. To know the price of a commodity Say 1, to know about the fertilizer of the commodity say 2, to know the

weather information about the commodity please say 3 and to end up the call please say Zero.
User: three
System: You have chosen three. To confirm it say yes or to deny say no.
User: yes
System: What is the commodity name for which you want to know Weather information?
User: Potato
System: You have said Potato. To confirm it say yes or to deny say no.
User: Yes.
System: ……………………………………………….
System: To know the weather information of another commodity please says yes or to go to main menu say no.
User: No
System: Welcome to Agricultural information system. To know the price of a commodity Say 1, to know about the fertilizer of the commodity say 2, to know the weather information about the commodity please say 3 and to end up the call please say Zero.
User: Zero
System: Thank You.

Collected speech corpus is transcript at lexical level with marking of dis-fluencies. The training of these non-lexical sounds helps to recognize or ignore such noise during decoding process and makes the system robust. Major modules of ASR system are signal processing, training and testing of sub-word acoustic models. Short –time spectral characteristics and temporal features are computed in signal processing module. **Table 5.1** represents the distribution of totally collected speech data according to the variations of Speakers' Gender, Age, education qualification, Recording Handset model, Service provider and Environment in terms of percentage of occurrences in each criteria.

Table 5.1: Variation in collected speech data

Gender	Male	70
	Female	30
Age (in %)	Adult: 0-15	66
	Medium: 30-50	28
Qualification	Primary	46
	Secondary	38
	Post-Secondary	12
	others	4
Handset model in %	Nokia	46
	Samsung	20
	LG	5
	Reliance	2
	Sony	2
	Others	25
Service Provider in %	BSNL	4
	Reliance	29
	Aircel	23
	Airtel	50
Environment (in %)	Noise	2
	Clean	85
	Babble	9

5.4 CORPUS ANALYSIS

The intelligibility and naturalness of speech is affected in telephone speech loudness loss, circuit noise, side tone loudness loss, room noise, attenuation distortion, taker echo, listener echo, quantizing distortion, phase jitter etc. We also find different background noise in our corpus. To tag this noise we have designed one Semi-Automatic Transcription Tool. This tool has been designed for offline transcription of recorded speech data, such that all transcriptions during data collection can be checked, corrected and verified manually by human experts. Automatic conversion of text to phoneme (phonetic transcription) is necessary to create pronunciation lexicon which will help the ASR System training. At the time of transcription we have to give some transcription remark tags and also noise tags. It’s totally human driven task. The Grapheme to Phoneme (G2P) conversion in Bodo is based on orthographic rules, Parts of Speech (POS) information and semantics [65]. It is an important task for data transcription. Descriptions of different background noise and percentage of occurrence is explained in **Table 5.2**

Table 5.2: Different noise tags with % of occurrence

Tag	Explanation/examples	% of occurrence
<air>	Air flow	7.92
<animal>	Animal Sound	1.09
<Bang>	Sudden impulsive noise due to banging of door	0
<beep>	Telephone Beep sound	5.59
<bird>	Sound of Bird	5.22
<bn>	General Background Noise	17.91
<br>	Breath noise	0.4
<bs>	Background speech (babble>	15.9
<bsong>	Background song	0
<bins>	Background instrument	0
<burp>	Burp	0
<cough>	Cough	6.1
<cry>	Children cry	0
<ct>	Clearing of throat	0.03
<horn>	Horn noise of vehicles	1.02
<ht>	Hesitation	0
<laugh>	Laughter	8.08
<ln>	Line noise	6.06
<ls>	Lip smack	0
<ns>	Hiccups, yawns, grunts	0
<pau>	Pause or silence	7
<ring>	Phone ringing	0
<sneeze>	Sneeze	0

<sniff>	Sniff	0
<tc>	Tongue click	0
<vn>	Vehicle noise	7

Table 5.3 describes the different rejection tag which is occurred due to signal error. Speech files marker by other tag set may be accepted and considered at the time of ASR training.

Table 5. 3: Rejection tag sets

Transcription Remarks (WR)	**Description**
AMP_SOUND (Amplitude)	Amplitude will be modified fully or partially
CLN_SOUND (Clean)	Speech sound contain non-overlapping non speech event
CLP_SOUND (Clipped)	Speech sound clipped
CPC_SOUND (Channel problem considered)	Channel drop occurs randomly in silence region which have not affect speech region
CPR_SOUND (Channel Problem rejected)	Some word or phoneme dropped randomly
IMP_SOUND (Improper)	In this case the utterance is slightly different than prompt in phoneme level
NOS_SOUND (noise with speech)	Noise within speech
RSS_SOUND	Reasonable silence within speech
REJ_SOUND (Rejected)	In this case speech signal is wrongly select or may be too many noise or may be non-sense words, can’t be able to understand
SIL_SOUND (Silence)	No speech Utterance present
TRA_SOUND (Truncate Accept)	Truncation of not so significant amount (may be one or two phoneme) speech Utterance
TRR_SOUND (Truncate Accept)	Truncation of significant amount speech utterance
TDW-SOUND (Totally different Word)	In this case the utterance is totally different from the corresponding prompt

TRR_SOUND: If the actual speech signal is truncated with the beginning of the signal or ending of the signal than truncated rejection occur. This truncation loses the information of the actual speech degrades the performance of the decoder. This situation occurs due to non-co-operation of the user with the system and often speaks either before or after the specified time for recording. **Figure 5.4** shows time domain and also spectra domain view of TRR_SOUND problem.

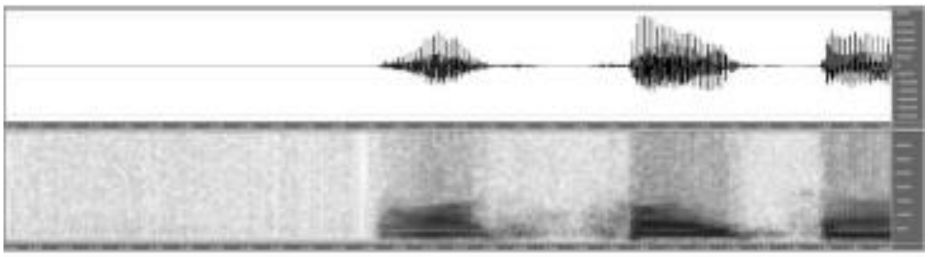


Figure 5.4 Sample speech file of TRR_SOUND

SIL_SOUND: In this kind of rejection no active speech is found in the entire wave file. This problem may arise due to network related issue or users' hesitation to speak. **Figure 5.5** shows time domain and also spectra domain view of SIL_SOUND problem.

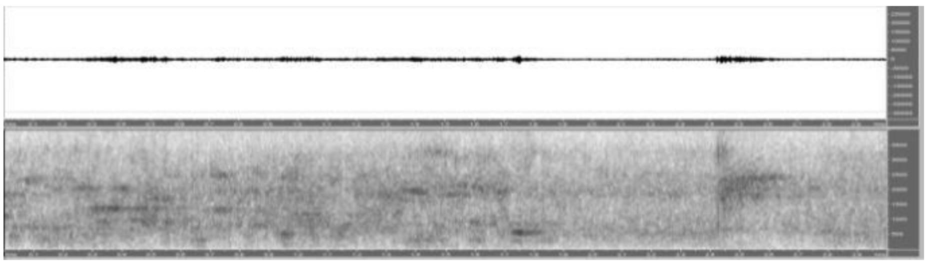


Figure 5.5: Sample speech file of SIL_SOUND

CPR_SOUND: During the transmission through telephone channel, sometimes we got silence in the active speech region. There for it loses the information of the actual signal totally or partially. **Figure 5.6** shows time domain and also spectra domain view of CPR_SOUND problem.

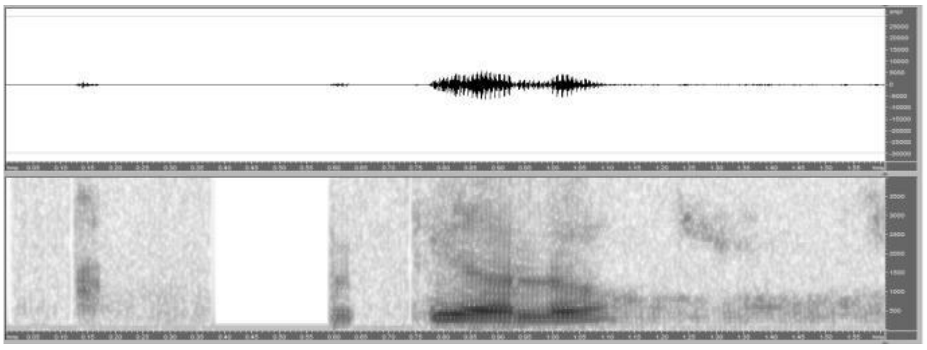


Figure 5.6 Sample speech files of CPR_SOUND

REJ_SOUND: These kinds of tag are very difficult to set because during conversation with the system the user spoke some unnecessary talks or nonsense words or may be too much cross talk etc. **Figure 5.7** shows time domain and also spectra domain view of REJ_SOUND problem. Where we have seen that few portions are speech segment and some portion are noisy. To tag the REJ_SOUND computationally is very big problem.

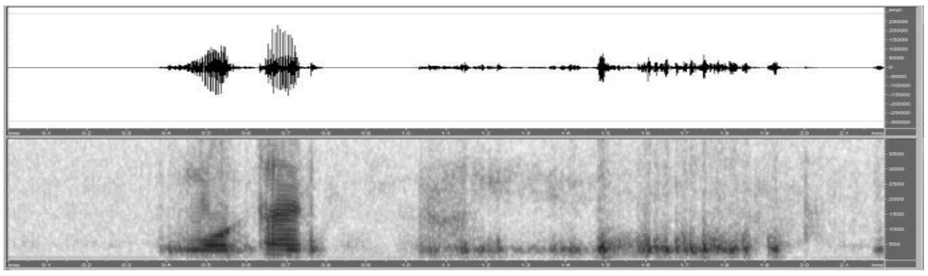


Figure 5.7: Sample speech file of REJ_SOUND

To solve this problem we analyze the STE, ZCR and Fo of collected speech corpus related to TRR_SOUND, SIL_SOUND and CPR_SOUND. **Figure 5.8** shows the different plots of these sounds.

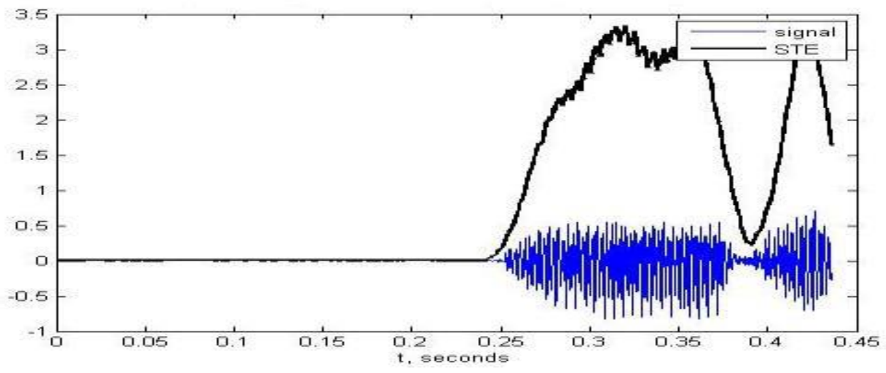


Figure: 5.8 (a) Signal vs. STE in TRR_SOUND

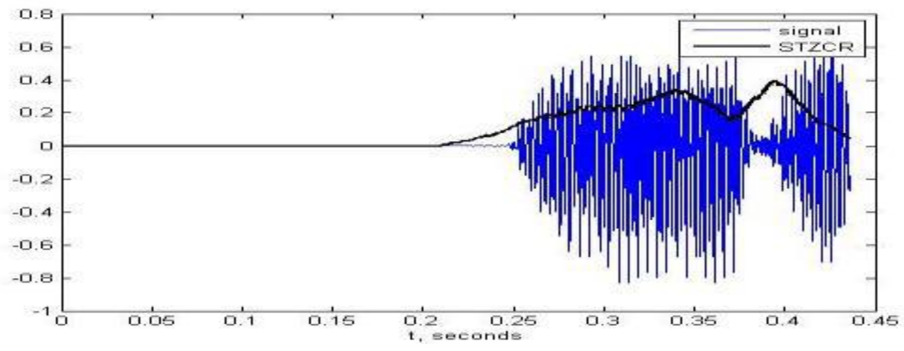


Figure 5.8 (b) Signal vs. ZCR in TRR_SOUND

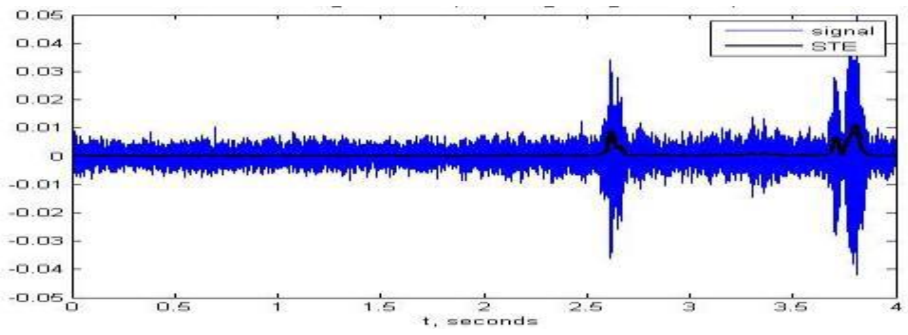


Figure 5.8(c) Signal vs. STE in SIL_SOUND

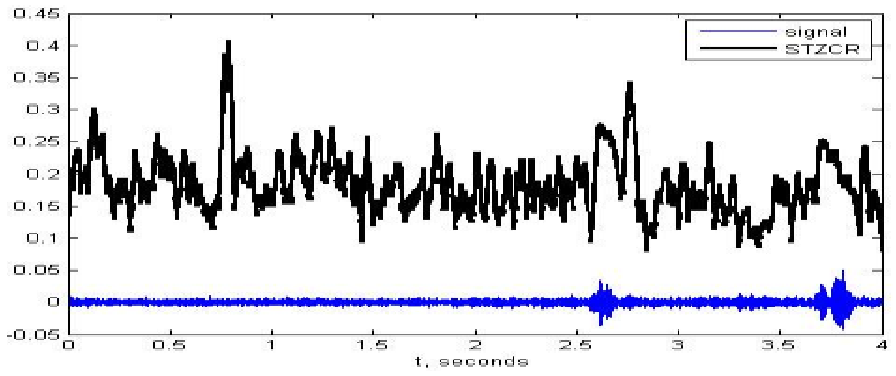


Figure 5.8 (d) Signal vs. ZCR in SIL_SOUND

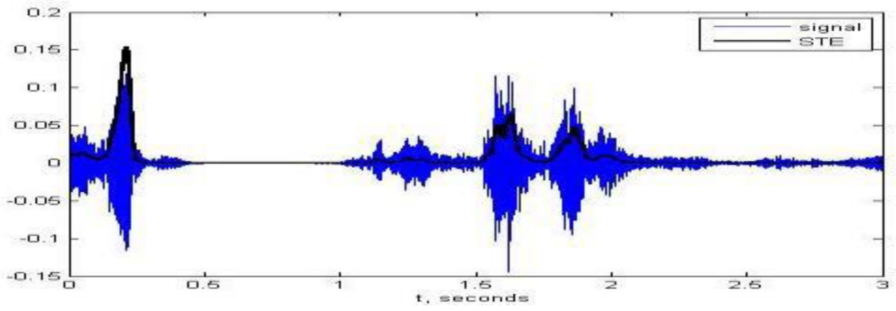


Figure 5.8 (e) Signal vs. ZCR in CPR_SOUND

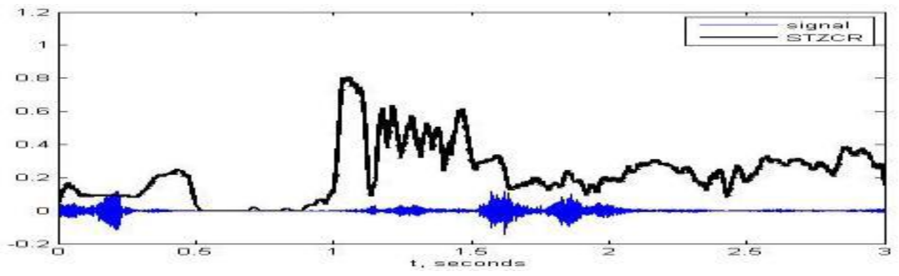


Figure 5.8: (f) Signal vs. in CPR_SOUND

The observations made from the above plots are given below:

- **a.** In case of TRR_SOUND i.e. the truncate problem there is a high STE and ZCR value at the beginning or ending of the file. It happens due the delay in speaking within specific time frame or sometimes starts so early.
- **b.** In case of SIL_SOUND it is observed that there is high ZCR or low STE value. This is because of channel noise or moistly unvoiced sounds.
- **c.** In case of CPR_SOUND is observed that ZCR and STE value fluctuate suddenly from high to low or from low to high .It happens because channel packets have been lost due to bad network or handset issue.
- **d.** Formant analysis also done for these rejection cases. But we observe that to tag SIL_SOUND is only possible with property. Others cases are not possible using formant analysis method.

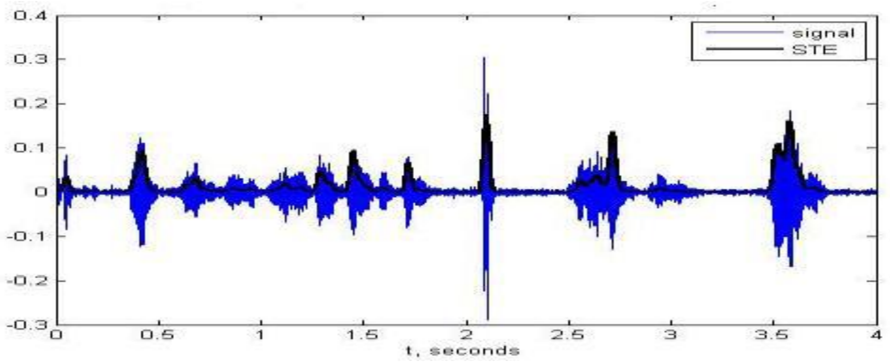


Figure 5.9(a)

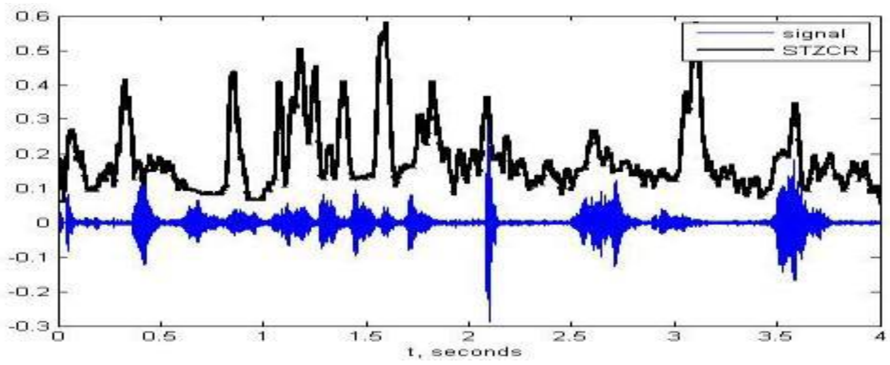


Figure 5.9(b)

Figure 5.9: REJ_SOUND: (a) Signal vs. STE (b) Signal vs. ZCR

Figure 5.9 shows another observation made for REJ_SOUND. It has been seen that because of the background noise there are many voiced and unvoiced regions are there many voiced and unvoiced regions are there. For this kind of speech signal it is very difficult to automatically extract the REJ_SOUND. When overall STE value is below expectation level of incoming speech signal then it is possible to mark REJ_SOUND.

The observations made from the above analysis we made the signal analysis module for building the recognition module of agricultural spoken dialogue system. **Table 5.4** shows result of automatic extraction of the rejection remarks except REJ_SOUND. To find out REJ_SOUND remarks system requires some manual intervention.

Table 5.4: Output of Signal Analysis Module and its decision

No. of incoming calls	No. of utterances	CPR_ SOUND	SIL_ SOUND	TRR_ SOUND	REJ_ SOUND
20	132	12	8	18	5

5.5 FEATURE EXTRACTION, TRAINING AND TESTING

For training acoustic models is necessary a set of feature files computed from the audio training data, one each for every recording in the training corpus. Each recording is transformed into a sequence of feature vectors consisting of the Mel-Frequency Cepstral Coefficients (MFCCs). Apart from the .wav files HTK 3 requires transcription File, Pronunciation dictionary, Filler dictionary for training of the system. So these files are created according to the required format.

A speech signal can be modeled as a series of piecewise stationary signals, which can be decoded to a set of one or more symbols. The continuous speech waveform is first preprocessed and converted into a sequence of discrete parameter vectors, called speech vectors. The role of the speech recognition system is to create a mapping between the speech vectors and the corresponding symbols. This system was trained using 5171 files of commodity and digit words collected from 100 speakers. The various parameters that are used for feature extraction are summarized in **Table 5.5.**

Table 5.5: MFCC Feature Extraction Parameters

Parameter	Default Value	Modified value
Pre emphasis coefficient	0.97	0.97
No of filters in filter bank	40	31
Lower cut –off freq	130 Hz	130
Upper cut-off freq	6800 Hz	3500
Sampling rate	8000 Hz	16000
Base feature Dim	13	13
Window length	0.00256 sec	0.00256 sec
Frame rate	100 per sec	100 per sec

The speech recognizer is developed using the open-source speech recognition toolkit HTK-3. The diagrammatic representation of the training and decoding modules are shown in **Figure 5.10** and **Figure 5.11** respectively. The tools HTKTrain-1.0 and HTK3.0.8 are used for training and decoding purposes respectively. MFCC features of 39 dimensions, comprising of 13 base features with its first and second derivatives, are used for speech parameterization [66-68].

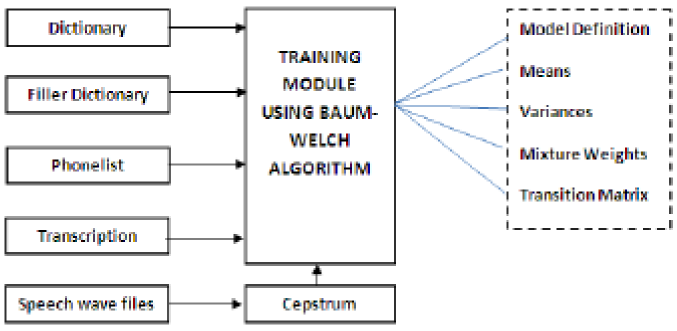


Figure 5.10: diagrammatic representation of ASR training module

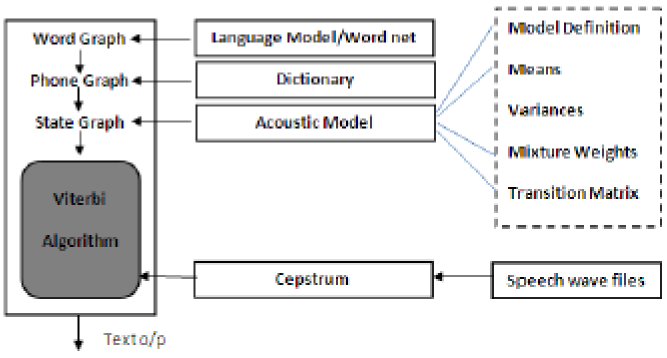


Figure 5.11: diagrammatic representation of ASR decoding module

During training time using HTK system verify several things in seven different phases. In first phase it checks dictionary and filler dictionary with phone list file. In the second phase it checks to make sure that there is not any duplicate entry in the dictionary. In the third step checks the general format and utterance length of the training file. Utterance length must be positive. The matching of total number of lines between the transcript file and the control file is done in the fourth step. In fifth steps it checks the amount of training data and verifies that the number of n_tied_states is reasonable for those training data or not. In the sixth step it verifies that all the transcript words are present in the dictionary or not. In the seventh step it verifies all the phone present in the transcript file is appear in the phone list or not. We have taken 6358 utterance words in the training phase and 1550 words in the testing phase. After completion of their training we have tested by new utterance words by the new input different Bodo speakers. For evaluating performance of speech system we have used the following equations where N is the number of words in test set, D is the number of deletions, S is number of substitutions and I is the number of insertions [6].

PC = (N − D − S)/N × 100 where PC gives word correction rate.

PA = (N − D − S − I)/N × 100 where PA gives word accuracy rate.

Word Error Rate (WER) = 100% − Percentage Accuracy (PA)

We have evaluated the performance by grouped all the item in one cluster and got WER as 83.29 in training mode and 79.12 % in testing mode.

Table 5.6: List of training word, occurrences in training set and % of accuracy

Entity Name	**Dictionary Word**	**Occ_ Count**	**Accuracy (%)**
माय	Mai	196	80
मायरं	Mairong	196	92
आलू	Aalu	198	77
सामब्रामगोजा	Saambraam- gwja	196	84
सामब्रामगूफूर	Saambraam- gufur	190	67
हाइजें	Haijeng	197	73
बानलुगोथां	Banlu-gwthang	196	98
आपेल	Aapel	198	84
बान्दाकबि	Bandakabi	203	100
फान्थाव	Phanthao	199	67
मुला	Mula	198	78
गाजर	Gajar	193	67
थाइबें	Thaibeng	199	76
थाइजौ	Thaijou	197	67
बेसर	Beshar	194	84
जिंखा	Zinkha	199	67
थालिर	Thalir	199	67
सुमफ्राम	Sumphram	200	83
तरमूज	Tarmuj	200	81
रैमालि	Raimali	199	79
खान्थाल	Khanthal	200	81
मौसूमि	Mausumi	198	89
नारेंखल	Narengkhal	193	91
खमला	Khamla	195	84
लतिखो	Lathikho	198	96
नई	Nai	198	97
से	Se	198	93
थम	Tham	198	95
नंगौ	Nongwo	324	97
नङा	Nonga	316	91

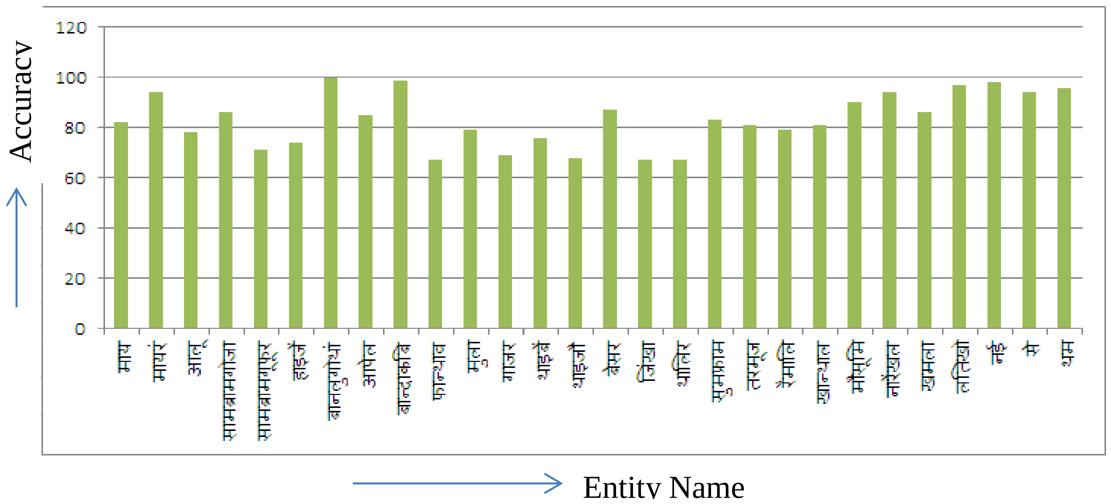


Figure 5.12: Percentage of Recognition Accuracy

5.6 RESULTS & DISCUSSION

The system performances in terms of word recognition accuracies with GMMs of different sizes and fixed number of tied states are given in **Table 5.7** Subsequently the number of tied states is also varied for 16 and 32 GMMs/state systems as given in **Table 5.8**. As the SDS system with 16 and 32 GMMs/state are found to give similar performances and hence we have used 16 GMMs/state in the deployed system.

Table 5.7: Word accuracies with diff GMM sizes and fixed no. of tied states

Decoder	**No. of GMM/state**				**Total Test files**
	4	**8**	**16**	**32**	
Commodity	66.73	68.45	62.69	62.63	1000
Bodo-digit	83.4	81.6	87.3	83.4	500
Yes/no	91.3	94.3	92.1	93.5	50

Table 5.8: Word accuracies with diff no .of tied states and 16 GMM vs 32 GMM

Decoder	**Sennon**								**Tot. Test files**
	1000		**1500**		**2000**		**2500**		
	16 GMM	32 GMM	16 GMM	32 GMM	16 GMM	32 GMM	16 GMM	32 GMM	
Commodity	70.23	69.5	71.34	62.8	72.38	72.8	72.69	72.6	1000

Bodo-Digit	87.3	82.1	84.6	80.7	86.3	80.2	86.2	82.2	500
Yes/No	93.1	89.2	93.2	88.3	89.3	91.3	93.4	92.4	50

SUMMARY

In this chapter we propose the model for building agricultural spoken dialouge system in Bodo language. The propose model consist of IVR, ASR and DATABASE server work. For building this model open source tool kit htk, asterisk, php agi , mysql is used. A user friendly call flow is design to get the price, fertilizer and weather information for a particular commodity. The make the system robust we propose some tagging mechanism for input speech. This tagging mechanism helps to error handing of the system. In truncate problem there is a high STE and ZCR value at the beginning or ending of the file. It happens due the delay in speaking within specific time frame or sometimes starts so early. In case of silence there is high ZCR or low STE value. This is because of channel noise or moistly unvoiced sounds. In case of clipped sound ZCR and STE value fluctuate suddenly from high to low or from low to high .It happens because channel packets have been lost due to bad network or handset issue. The trainer and the decoder configuration files have several parameters. These parameters can be tested to improve the efficiency of the speech recognition system.