Implementation of Word Level Speech Recognition System for Punjabi Language

Shama Mittal
Student, CSED
Thapar University, Patiala
(Punjab), India

Rupinderdeep Kaur
Lecturer, CSED
Thapar University, Patiala
(Punjab), India

ABSTRACT
In this paper the implementation of the word level speech recognition system for Punjabi language is explained because it is a highly prosodic language. Here HTK Toolkit along with Julius Toolkit is used. First step is data collection and two hours data is collected in read speech mode. Second step is data preparation, in which hmmlist, grammar and dictionary files are created. Once the data is prepared, 75% and 25% of data is used for training and testing respectively. The experimental results show that the accuracy of the system comes out to be 57.54%

Keywords
Automatic Speech Recognition (ASR), Hidden Markov Toolkit (HTK), Julius, Punjabi

1. INTRODUCTION
The ability of human beings to create communication between machines and computers with the help of keyboards or some other external devices is more cumbersome and slow too. A speech can be considered as a significant component to make this communication intelligible and rapid [1]. The process of recognizing and translating the spoken words, using the technologies which uses computerized devices, such as smart technologies and robotics, is known as Automatic Speech Recognition (ASR).

In an Automatic Speech Recognition system, an articulation of signals of speech is taken as an input and is converted into as much as promising to the text sequence of the data spoken by the user.

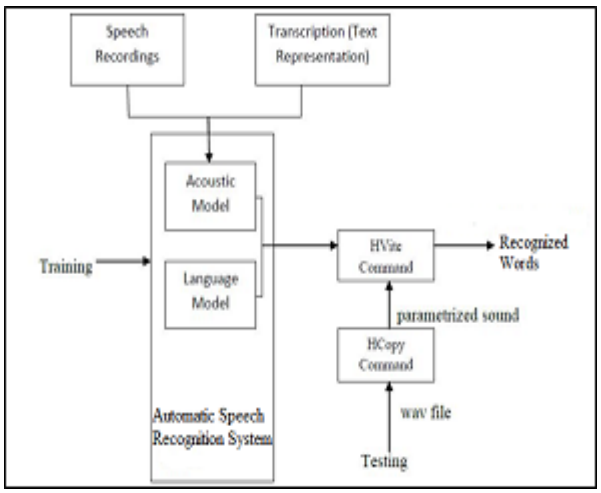

Figure 1. General frame work for ASR

The general frame work used in speech-to-text recognition system is shown in figure 1. Main problem in development of ASR system is because of the variety of the speaking style of different human beings and along with this environmental disturbances are the main reason too [10].

So the main objective of ASR system is to revolutionize the signals based on speech into the signals based on text which are not dependent on device, surroundings or the speaker in an efficient and more accurate way as possible.

2. HTK TOOLKIT
In order to introduce the HTK tools, the best way is to follow the steps for processing which are involved in creating the continuous speech recognizer based on sub-words [2]. As shown in figure 2, four main phases are involved in this procedure:

- Data Preparation
- Data Training
- Data Testing
- Data Analysis

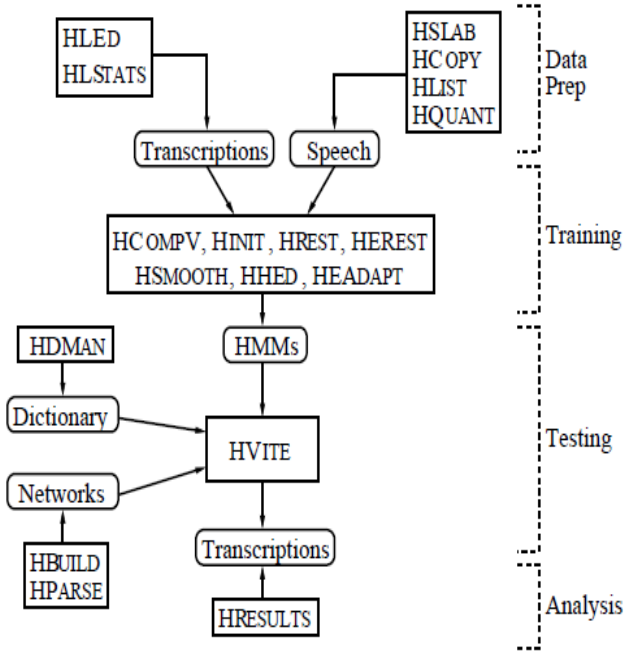

Figure 2: HTK Processing Stages

2.1 Data Preparation Tools
A bunch of speech files and the corresponding transcription files are needed to create a set of HMMs. Before it is used for training, it is necessary to convert it into a convenient parametric form. Thus, any corresponding transcription file need to be transformed such that appropriate format is created

and the recommended phone or word labels can be used properly.

2.2 Training Tools

In second step of system development, prototype definition is written for each HMM, so that the required topology for each HMM can be defined. The only purpose of defining a prototype is to enumerate the overall characteristics and topology of HMM. Here, no actual parameters are evaluated.

The transitions came out from any state, made equally likely with the help of these probabilities.

The actual training process takes place in stages. First of all, an initial set of modules is created. The same can be used as a bootstrap data if some of the speech data is achievable for which sub-word's (*i.e.* phone) boundaries have been marked. In such cases, *HInit* and *HRest* tools use fully labelled bootstrap data to give provision for an isolated style training. The training data is uniformly segmented on the first cycle. Here, corresponding data segments are matched with each model state, so that, means and variances can be computed. Gaussian models mixtures are trained to use a mutated version of k-means clustering. Successively, on the second cycle, the uniform segmentation has to be replaced by Viterbi alignment [14]. First of all, *HInit* computes the initial parameters and then through *HRest* further re-estimation is done. Now, segmental k-means procedure is replaced with Baum-Welch re-estimation [15] procedure using once again the same bootstrap data. The brief introduction of the HTK commands used in this work are explained in Table 1 below.

Table 1. Brief Summary of HTK Commands

Sr. No	Command	Phase	Description
1	HLEd	Data Preparation	Label editor designed for making the appropriate changes to the label files and store results in a single Master Label File (MLF).
2	HCopy	Data Preparation	Main objective is to copy the data of one or many source files and sends the data in to an output file.
3	HInit	Data Training	Reads all of the data used in bootstrap training and all the required phones examples are cut out.
4	HCompV	Data Training	When all the phone models discovered, are identically initialized to have the state variances and means equal to global variances and means.
5	HDMan	Recognition	Used for preparing a pronunciation dictionary from different sources. From the given scrip file a list of editing commands are read out and the results of modified and joined copies of one or more dictionaries came out as outputs.
6	HVite	Recognition	Used to evaluate the lattices and models using language.
7	HParse	Analysis	Word level lattice files are generated using a syntax description text file which contains a bunch of modified rules based on the extended Backus-Naur Form (EBNF).
8	HResults	Analysis	In order to assess the performance of the recognizer based on HMM, pre transcribe some test sentences which are pre-recorded. And then, match the output obtained from recogniser with the actual reference transcriptions.

2.3 Recognition Tools

HVite is an HTK recognition tool that allows the evaluation using language lattices and models. A tool named *HLRecsore*, used for generating the lattices with the help of *HVite* (or *HDecode*) to be manipulated. *HDecode* is to be distributed under the more restrictive license agreement.

2.4 Analysis Tools

As soon as the recogniser based on HMM is built, it's compulsory to assess its performance. Mainly, this is carried out by transcribing some of pre-recorded sentences used for testing and then match the outcome generated by the recogniser with the original transcriptions. For this purpose, *HResults* tool is used which will adjust both the transcriptions using dynamic programming, and thus counting of replaced, inserted and deleted errors is done. *HResults* provides a speaker-by-speaker breakdowns, transcriptions aligned timely and confusion matrices for the sake of global performance measures.

3. JULIUS

Julius is a large vocabulary continuous speech recognition (LVCSR) engine. It is a high performance decoder software which is used by speech related researchers and developers [3]. It incorporates major search techniques like tree lexicon, enveloped beam search, Gaussian selection, Gaussian pruning etc. It is not only used for searching process, but also used for modularizing the model structures in a way that independent model structures are obtained. Along with this various HMM types such as tied-mixture models and shared-state triphones are supported too and with any number of phones, mixtures or states.

The main platform for working of Julius is Linux and other Unix Operating Systems. It can also perform well on Windows Operating system.

4. LITERATURE SURVEY

Based on the parameters like language, tool used and year of publication, a review of around 50 research papers has been done. Out of which 11 selected papers which are most appropriate with the proposed work, and are summarized in Table 2.

Table 2. Brief Summary of different authors for speech-to-text system

Sr. No	Author Name	Year	Tools/ Technique Used	Language	Further Work
1	Azmi et *al. [4]*	2008	HTK(Hidden Markov Tool Kit) tool	Arabic	• Comparison of mono-phone and tri-phone • MFCC features extracted
2	R.Kumar *et al. [5]*	2008	HMM(Hidden Markov Models) and DTW(Dynamic Time Wrap) techniques	Punjabi	• Recognize speaker independent isolated words • Comparison of performance of speaker dependent with speaker independent using small vocabulary
3	R.Kumar *et al. [6]*	2011	Vector Quantization and DTW technique	Punjabi	• Developed a speaker independent isolated word ASR(Automatic Speech Recognition)
4	K.Kumar *et al. [7]*	2012	HTK	Hindi	• Recognize connected words with a vocabulary of 102 words
5	Al-Qatab *et al. [8]*	2010	HTK	Arabic	• Recognize isolated words • Recognize continuous speech • Both with a vocabulary of 33 words
6	S.Mandal *et al. [9]*	2010	SPHINX3	Bengali	• E-mail application • Developed for visually challenged people
7	Lee *et al. [11]*	1989	SPHINX	English	• HMM & LPC as feature vectors • Phone concatenate to form word, word concatenate to form large sentence
8	Woodland *et al [12]*	1995	HTK	English	• MFCC as a feature extraction technique
9	Choudhary *et al. [13]*	2013	HMM	Hindi	• Recognize isolated and connected words • System was trained for hundred different isolated words • Each word is uttered ten times
10	Lee *et al. [14]*	1989	HMM	English	• Novel smoothing algorithm was used • Smooth out the HMM output parameters
11	Ming et. al. [16]	1998	Bayesian Principle for triphones	---	• Tri-phone models are built using models of less context dependency

5. PROPOSED WORK

This paper aims to discuss the performance of a Punjabi (Gurmukhi script) isolated automated transcription consisting of 266 word vocabulary and developed to work in both speaker dependent and speaker independent real time environments.

Here in this work, 2 hours of data is collected in read speech mode. The data is collected from 5 females and 2 males. First of all, the long duration input is divided into 4ms to 5ms duration wave files using Wave Surfer software. After dividing the long duration sound file into short wave files transcription of data is done using the standard IPA chart. And the whole data is transcribed by 5 different transcribers using their transcription patterns. Once the transcription of whole

data is done then preparation, training and testing of data is done, so that the person came to know how accurately the system is able to identify the words.

Step 1: Data Preparation

- Save whole transcription (IPA) in transcription_all.txt file and put all the .wav files separately in a single folder.
- Create a new file list.txt in which all the IPA and their corresponding ASCII are written.
- Now need to install HTK and Julius in Ubuntu OS, for this just copy the binary files in usr/local/bin
- HTK takes input only in ASCII form not in IPA form, so next step is to convert IPA to ASCII i.e. need to convert all the transcription written in transcription_all.txt into ASCII.
- For IPA to ASCII conversion follow the following steps:
    - Copy only the transcription part from transcription_all.txt file to another file i.e.trans_all.txt.
    - At the beginning of transcription of each wav file introduce '%' symbol and at the end of each transcription introduce '@' symbol.
    - Introduce single space after each character using command on terminal:-
    - sed 's/./& /g' trans_all.txt> trans_space.txt
    - Introduce file name of their corresponding transcription in trans-space.txt
    - Replace '%' symbol with .lab
    - Run convert.sh script on terminal. This script will convert all IPA in trans_space.txt to their corresponding ASCII as given in list.txt and save it another file trans_ascii.txt.
    - Now find $ in trans_ascii.txt and replace it with /n and @ with /n
    - Now find & in trans_ascii.txt and replace it with /n so that in each line only a single word is present.
- Now check the file for all symbols, if any symbol not converted then convert it manually.
- Now need to create two mlf files. For this:
    - Introduce #!MLF!# at the beginning in trans_ascii.txt and rename it as train.mlf. This file is for training purpose.
    - Now create another file test.mlf and cut the last (say 25% of the total files) from train.mlf and save it in test.mlf with #!MLF!# at the beginning. This file is for testing purpose.

Thus transcription of all wav files are converted into ASCII form to be given as input to HTK toolkit.

- Copy all the ASCII symbols in two columns and name the file as dict.txt
- Write all the symbols in a separate file and symbols should be in concatenated form and name the file as gram.txt
- Now copy all the unique ASCII codes in single row and save it as hmmlist.txt
- Now create a file target.list which contains the path of all wav (training + testing) files and correspondingly path of .mfc files that are to be generated
- Now copy the path of .mfc files for only those wav files which are listed in train.mlf and name the file as train.list
- Now copy the path of .mfc files for only those wav files which are listed in test.mlf and name the file as test.list

Step 2: Data Training

- Components required for word level training:-
    - A file consisting of path of MFCC features of all wav files used for training (say, train.list).
    - List of Models (say, hmmlist.txt).
    - Configuration file (say, analysis.conf).
    - Prototype Model (say, proto.txt).
    - Mean and Variance Model (say, hmmdefs and macros).
- Create .mfc files using target.list with the help of HCopy command on terminal as:-

    HCopy -T 1 -C analysis.conf -S target.list
- transcription_all.txt file is given to the transcription2mlf.jl script, to generate word.mlf file. word.mlf can be created by executing following command on terminal:

    julia transcription2mlf.jl transcription_all.txt word.mlf
- Now, this word.mlf is used to create train.mlf file.
- At the time of training the system, for each ASCII character 10 mixtures will generate from hmm0 to hmm9. These mixtures are generated using proto.txt. and hmmlist
- Now, train.mlf along with mktri.led (script of Julia language) is used to create triphone list (say, triphones1) and an mlf file containing these triphones instead of single phones.
- For each ASCII character 6 new mixtures will generate from hmm10 to hmm15 using triphone and wintri,mlf. These mixtures are for training the system for word level recognition.
- The triphone list and wintri.mlf are also used for generating dictionary for word level and tiedlidst (used further for testing).

Training is completed after performing all these steps.

Step 3: Data Testing

- Components required for testing are:-
    - List of Models (say, tiedlist).
    - Pronunciation dictionary (say, tri_dict.txt).
    - Configuration file (analysis_train.conf).
    - Testing file (say test.mlf)
    - A file consisting of path of MFCC features of wav files that are to be recognized (say, test.list).
    - Grammar (say, gram.txt).

- Master Macro File in which mixtures of each state of each HMM model are computed (say, hmmdef and macros).

- Wdnet.txt is created using tri_dict.txt and tiedlist.
- Now these files are used and with the help of HVite command recout_test.mlf is created.
- At the time of testing, HResults command is used to compare recout_test.mlf (system generated) file with test.mlf (manual) and accordingly results are displayed on the terminal in the form of % accuracy and %correctness.

Testing is completed after performing all these steps.

An automated transcription process is done for read speech mode, and from the whole data 75% of data is used for training and 25% for testing purpose.

6. PUNJABI LANGUAGE TRANSCRIPTION

The visual representation of speech sounds (phones) is called the phonetic transcription. IPA (International Phonetic Alphabet) is the most common type of phonetic transcription. Suppose there is a word 'ਮੁੱਖ ਗੱਲ' in Punjabi, this can be transcribed as 'mʊkkʰ gəll' using the IPA chart. Phonetic transcription deals with the sound of phones used in words, i.e., it tells us about the pronunciation of the words. Another advantage of transcribing words using IPA chart, is that computer can be made understand utterances of the language on which the person is working as all the languages can be transcribed into IPA format.

Transcription can be done at phoneme level, word level or at syllable level. In this paper, main focus has been the transcription at phoneme level. Some of the examples of phonetic transcription are given below in Table 3.

Table 3. Symbolic Form Representation of Spoken Utterances

Punjabi	Transcription
ਸਾਰਾ ਮਨੁੱਖੀ ਪਰੀਵਾਰ ਆਪਕੀ ਮਹਮਿਾ	sara mənʊkʰi pərɪvar apki məhɪma
ਜੀ ਆਇਆਂ ਨੂੰ	ʤɜɔ ɑɑeːiaŋ ŋʊʰ
ਸੱਜੇ ਹਥ ਮੁੜ ਜਾਣਾ	səʤʤe həʈʰ mʊɳ ʤaɳa

7. EXPERIMENTAL RESULTS

Equations (1), (2) and (3) are used for analyzing the results:

$$\text{Accuracy} = ((H - I) \div N) \times 100 \quad (1)$$

$$\text{Correctness} = (H \div N) \times 100 \quad (2)$$

$$N = H + D + S \quad (3)$$

Where, N is total no. of phones, H is correctly detected phones, D is the number of deleted phones, I is the number of inserted phones i.e. external insertion of phones, and S is the substituted phones.

Table 4. Result for word level recognition

Training Data (in Minutes)	Testing Data (in Minutes)	Accuracy (in %)	Correctness (in %)
138.20	48.13	57.54	65.36

Table 4 represent the results for word level recognition system which include the data used for training and testing as well as the accuracy and correctness of the system.

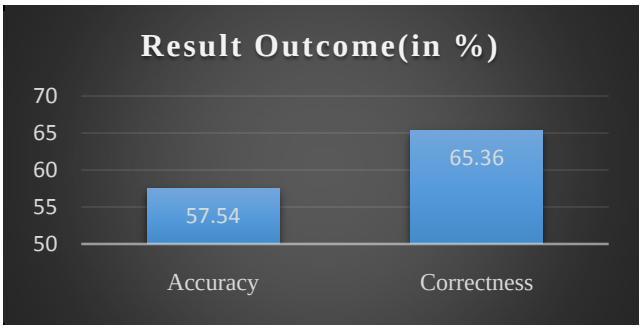

Figure 3: Results for word level recognition

Figure 3 represents the result for word level recognition system for Punjabi language and it shows that the system is 57.54 % accurate and 65.36% correctly recognize the system.

8. CONCLUSION AND FUTURE SCOPE

In this research work, the speech to text recognition system with Punjabi language shows the useful results. The speech is transcribed to ASCII characters using IPA chart. This research work is for recognizing the text word by word. The previous research work done in this field is only up to phone level recognition and more over less work is done for Punjabi language too. This work can also be performed for other regional and foreign languages.

9. FUTURE SCOPE

The same work can also be done with lecture speech mode and conversational speech mode. Moreover to increase the accuracy and correctness in case of read speech mode more data can be collected so that the system can be trained with large amount of data. This work can also be performed by using Audacity tool for recording the sounds. Using Audacity run time sound input can be given to system during testing.

10. REFERENCES

[1] HTK "Hidden Markov Model Toolkit", available at "http://htk.eng.cam.ac.uk", 2012.

[2] L.R. Rabiner , "A Tutorial on Hidden Markov Models and Selected Applications in Speech Recognition", Proc. of the IEEE Vol. 77, Issue 2,pp. 257–286,1989.

[3] "Introduction of Julius," http://julius.osdn.jp/en_index.php

[4] Azmi, M., Tolba, H., Mahdy, S., and Fashal, M. (2008). "Syllable-based automatic Arabic speech recognition", Proceedings of the 7th WSEAS International Conference on Signal Processing, Robotics and Automation, World Scientific and Engineering Academy and Society, WSEAS, 246-250.

[5] R. Kumar "Comparison of HMM and DTW for Isolated Word Recognition of Punjabi Language" In Proceedings

of Progress in Pattern Recognition, Image Analysis, Computer Vision, and Applications, Sao Paulo, Brazil. Vol. 6419 of Lecture Notes in Computer Science (LNCS), pp. 244– 252, Springer Verlag, November 8-11, 2010.

[6] R. Kumar and M. Singh, "Spoken isolated Word Recognition of Punjabi Language Using dynamic time Warping Technique" Demo in Proceedings of Information System for Indian Languages, Punjabi University, Patiala, India, March 9 - 11, 2011. Vol. 139 of Communication in Computer and Information Science (CCIS), Page 301, Springer Verlag.

[7] K. Kumar, R. K. Aggarwal, and A. Jain "A Hindi speech recognition system for connected words using HTK" International Journal Computational Systems Engineering, Vol. 1, No. 1, 2012

[8] B. A. Q. Al-Qatab and R. N. Ainon, "Arabic Speech Recognition Using Hidden Markov Model Toolkit (HTK)", Paper presented at International Symposium in Information Technology (ITSim). Kuala Lumpur, June 15-17, 2010.

[9] Mandal, S., Das, D., Mitra, P.: SHRUTI-II: A Verncular Speech Recognition System in Bengali and an Application.

[10] S. Young, "Hidden Markov Model Toolkit: Design and Philosophy," CUED/F-INENG/TR.152, Cambridge University Engineering Department, Sept. 1994.

[11] Lee, K. F., Hon, H. W., Hwang, M. Y., and Mahajan, S. (1989), "The SPHINX speech recognition system", Proceedings of the IEEE International Conference in Acoustics, Speech and Signal Processing.

[12] Woodland, P. C., Leggetter, C. J., Odell, J. J., Valtchev, V., and Young, S. J. (1995). "The HTK large vocabulary speech recognition system", IEEE International Conference on Acoustics, Speech, and Signal Processing, ICASSP, 1, 73-76.

[13] Choudhary, A., Chauhan, M. R., and Gupta, M. G. (2013). "Automatic Speech Recognition System for Isolated and Connected Words of Hindi Language By Using Hidden Markov Model Toolkit (HTK)".

[14] Lee, K. F., and Hon, H. W. (1989). "Speaker-independent phone recognition using hidden Markov models", IEEE Transactions on Acoustics, Speech and Signal Processing, 37(11), 1641-1648.

[15] Steve Young, Gunnar Evermann, Mark Gales. HTK Book (for HTK version 3.4). England, Cambridge University of Engineering Department, 2006.

[16] Ming, J., and Smith, F. J. (1998). "Improved phone recognition using Bayesian tri phone models", Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing, 1, 409-412