Available online at www.sciencedirect.com

SciVerse ScienceDirect

Procedia Engineering 30 (2012) 402 – 409

Procedia Engineering

www.elsevier.com/locate/procedia

International Conference on Communication Technology and System Design 2011

Error Minimization in Phoneme based Automated Speech Recognition for Similar Sounding Phonemes

Karan Gangaputra, a*

National Institute of Technology, Surathkal, Karnataka-570025 Indib

Abstract

The technology of ASR (Automated Speech Recognition) has been quite successful with the use of Hidden Markov Model (HHM) with the aid of probabilistic and the best path methods. The words with limited vocabulary content can easily be modeled and trained. The modules with a large dictionary has to be modeled with context independent *phones* joining together to form a word. The major problem lies in recognizing the word with similar sounding phonemes.

This paper aims in minimizing the error with similar sounding phonemes by using the viterbi algorithm for each similar syllable in the backward direction.

Keywords: Automated Speech recognition; Phoneme based correction; Error minimization;

1. Introduction

Speech recognition has evolved as one of the leading fields in computer science considering the recent trends. The telecommunication industry greatly relies on it to automate most of its services by replacing manual operators, enhancing the features of voice to text message, aiding autism patients by replacing manual operators and so on.

The mistakes in identifying similar sounding phones (like the' a' series (a, j, k) 'e' series like (b, c, d. e, g, p, v) etc) sans considering the number of times system has been trained with that alphabet. When we consider a word as a whole the system constructs the sound as a whole. In a speech recognition system

* Karan gangaputra. Tel.: +91-8880207075.
E-mail address: karan.gangaputra@gmail.com.

1877-7058
doi:10.1016/j.proeng.2012.01.878

with a large vocabulary system the word are formed by joining the phonemes e.g. Obama is analyzed as O-ba-ma.

Consider a sample grammar consisting of the similar sounding phonemes (o, ba, ma, sa, ka, cha, ra, la, a). Due to similarity, the system assumes most of these phones as a single phone; let's call this phone as a super phone. Iteratively removing the super phones we get a new super phone, lets name it with number of iteration, e.g.: 1-superphone, 2-superphone.Now even if we form a dictionary consisting of a word made of higher order super phones then all the other permutation of the similar sounding phones would be recognized as the former.

Speech recognition systems generally assume that the speech signal is a realization of some message encoded as a sequence of one or more symbols. The Hidden Markov Models (HMM) forms the basic building blocks of the speech recognition system. HMM is given a keyword or multiple instances of the keyword and a stream of utterance which is basically the search space where it has to locate the keywords if present. One way to identify keyword in the utterance is to pass the utterance through the HMM based system and get transcription for the utterance. Viterbi algorithm is useful in identifying the best path a signal can take in a HMM. Viterbi uses 27 dynamic programming to reduce search space. In this paper we tend to apply Viterbi in the proper direction until the similar sounding posteriority is removed.

2. Related work

Lidia Mangu et al [4], proposed high sentence posterior probability using Maximum a-posteriority Probability (MAP) approach. Later Steve Young et al [3], modified to word posterior probability to reduce the word error rate using the N-best rescoring. This approach failed in yielding the accuracy in recognizing the accuracy for similar sounding sounds, because the fault occurs in the posterior of the syllable and the following method will give rise to overlapping probabilities with very less differtiating boundaries. The pruning of the similar sounding posterior was also suggested at the sentence lattice level, but no standard threshold was formed to exercise this process.

3. Overview of speech recognition system

As shown in Fig 1, the general speech recognition processes the input speech signal and quantifies it into a known standard valuation system (in this case MFCC). Mel frequency cepstral coefficient (MFCC) are most commonly used quantifying feature set in speech recognition. Speech signal is divided into short frames over the range it can be considered stationary. Fourier transform of each frame signal is obtained. Mel scaling is applied to the Fourier coefficients using triangular overlapping windows. These filters are linearly spaced for lower frequency, while log spaced filters at higher frequency. Logs of the powers are computed at each of the mel frequency. Discrete cosine transform (DCT) is computed of the mel log power. MFCCs are the amplitudes of resulting spectrum .The following standard is traced with trained standard where the probable match is maximum. The trained data standard is set by a combination of emission probability of the phone signal and transmission from one phone to other [3].

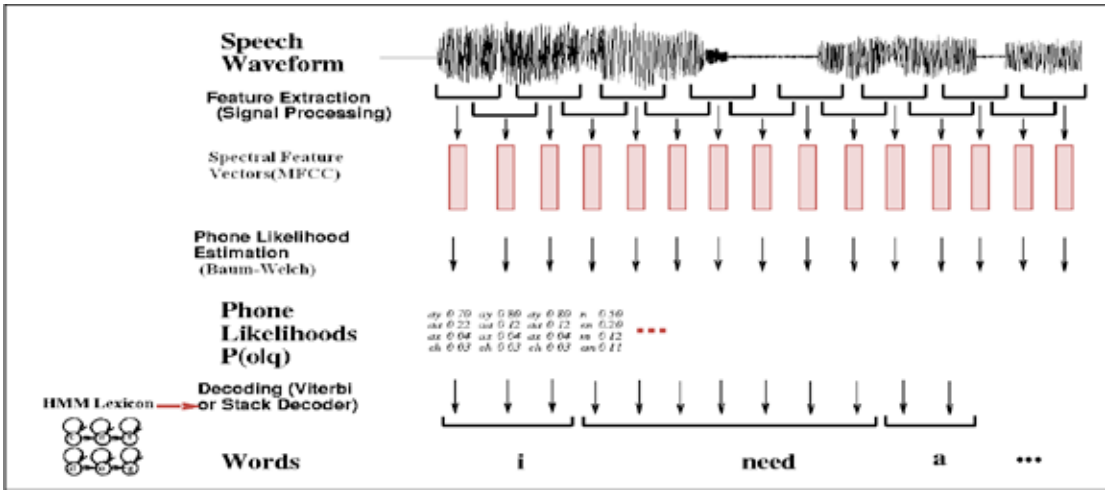

Fig. 1. Architecture of Speech recognition

General speech recognition requires a large set of sound data for both testing and training. The sound data is converted to feature vectors to establish parameters (generally mfcc (mel frequency cepstral coefficient**)** are used). The data is trained using the Baum-welch re-estimation to set the trained standard.

Viterbi algorithm is used to map the given quantified signal with established trained standard [1]. In this algorithm for the first observation sequence we find out probability of a state being the start state. His is done by taking the product of initial probability and the observation probability for the state. For every other observation all the states try to find the predecessor such that the probability of the predecessor multiplied by the transition probability from the predecessor to itself is maximized.

4. Proposed method

The error rate in the similar sounding phonemes can be reduced by generating an emission standard (using the Baum Welch algorithm) of the last frame and mapping it in the backward direction using the Viterbi algorithm until the similar phone is repeated. The process can be used for the other phoneme frame iteratively, to give a better judgment factor.

4.1. Algorithm to use the reverse Viterbi algorithm as proposed

Step 1: Take the input testing word ‘W’ for recognition.

Step 2: Reduce the frame size ‘F’=100ns using Hcopy. (Hcopy is used to copy the input files to an output file and to convert the parameters ‘on the fly’) [3]

Step 3: Separate all the phones ‘p’ in their respective states.

Step 4: a. Let the initial emission ‘e’ of the end of state1 be α_1.

b. for p=1 to p=n (no of phones)S Verify if any other phone has a similar emission ‘e’ in the same position. // Lets call the phones having similar emissions in the same position as ‘S’.c. Apply Viterbi in the backward direction to all the phones having the similar emission. (S). Apply Viterbi for all S d.

Count the no of frames (N) with similar transitions and emissions of each phones. N=N+S

Step 5: Iterate the above process for the other states (state 2 to state n).

Step 6: Repeat step 2 to step 5 for all the Test phonemes.

Step 7: Apply the general Viterbi by avoiding those frames.

4.2. Flowchart to use the reverse Viterbi algorithm as propose

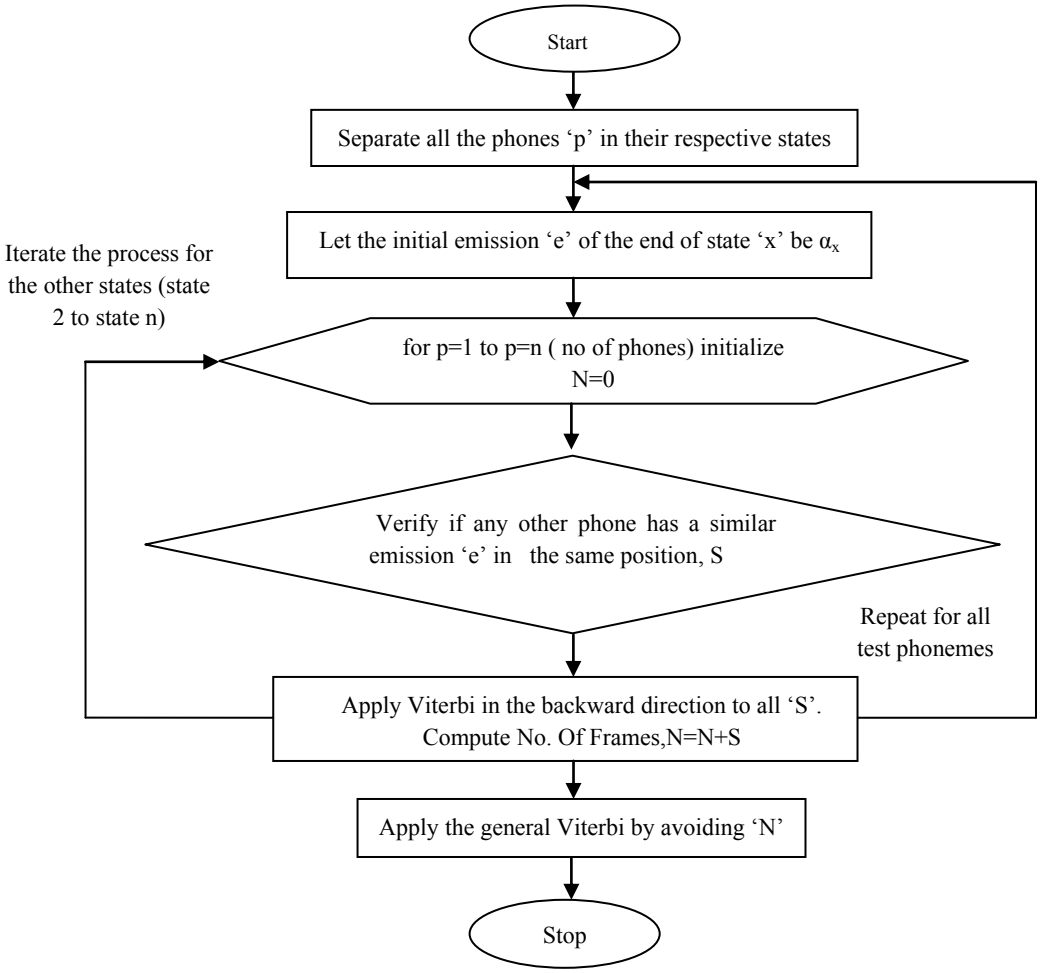

Fig. 2. Flowchart for the proposed method

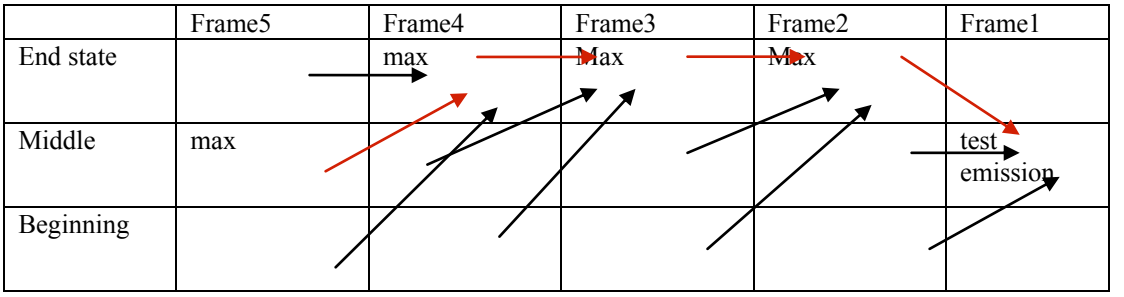

Fig 3. Applicability of the proposed method

5. Implementation and result analysis

HTK-(HMM toolkit) was used for this purpose. It makes matured HMM implementation. The license does not allow re distribution of the code. The HTK consists of various tools which enables data preparation and to use the algorithms in the general method [3]. Fig 4 shows different tools and the stages in HTK as it accepts the input and till it gets the output. The following tools were used:

HLEd: to convert the word mlf files to phone mlf files. This program is a simple editor for manipulating label files. HLEd works by reading in a list of *editing* commands from an edit script file and then makes an edited copy of one or more label files. For multiple level files, edit commands are applied to the *current level* which is initially the first (i.e. 1). Other levels may be edited by moving to the required level using the ML Move Level command.

HCopy: to copy the input files to an output file and to convert the parameters 'on the fly'. This program will copy one or more data files to a designated output file, optionally converting the data into a parameterized form. While the source files can be in any supported format, the output format is always HTK. By default, the whole of the source file is copied to the target but options exist to only copy a specified segment. Hence, this program is used to convert data files in other formats to the HTK format, to concatenate or segment data files, and to parameterize the result.

HCompV: to compute the GMM with global mean and variances. It is primarily used to initialize the parameters of a HMM such that all component means and all covariance's are set equal to the global data mean and covariance.

HInit: to initialize data with K- means. HInit is used to provide initial estimates for the parameters of a single HMM using a set of observation sequences. It works by repeatedly using Viterbi alignment to segment the training observations and then recomputing the parameters by pooling the vectors in each segment. HInit can be used to provide initial estimates of whole word models in which case the observation sequences are realizations of the corresponding vocabulary word. Alternatively, HInit can be used to generate initial estimates of *seed* HMMs for phoneme-based speech recognition.

HRest: to execute Baum-Welch re estimations. HRest performs basic Baum-Welch re-estimation of the parameters of a single HMM using a set of observation sequences. HRest can be used for normal isolated word training in which the observation sequences are realizations of the corresponding vocabulary word.

HParse:To create the word network from the grammar. The HParse program generates word level lattice files (for use with e.g. HVite) from a text file syntax description containing a set of rewrite rules based on extended Backus-Naur Form (EBNF). The EBNF rules are used to generate an internal representation of the corresponding finite-state
network where HParse network nodes represent the words in the network, and are connected via sets of links. This HParse network is then converted to HTK V2 word level lattice.

HVite: To run Viterbi recognition algorithm. HVite is a general-purpose Viterbi word recognizer. It will match a speech file against a network of HMMs and output a transcription for each. When performing N-best recognition a word level lattice containing multiple hypotheses can also be produced.

HResults: is the HTK performance analysis tool. It reads in a set of label files (typically output from a recognition tool such as HVite) and compares them with the corresponding reference transcription files.

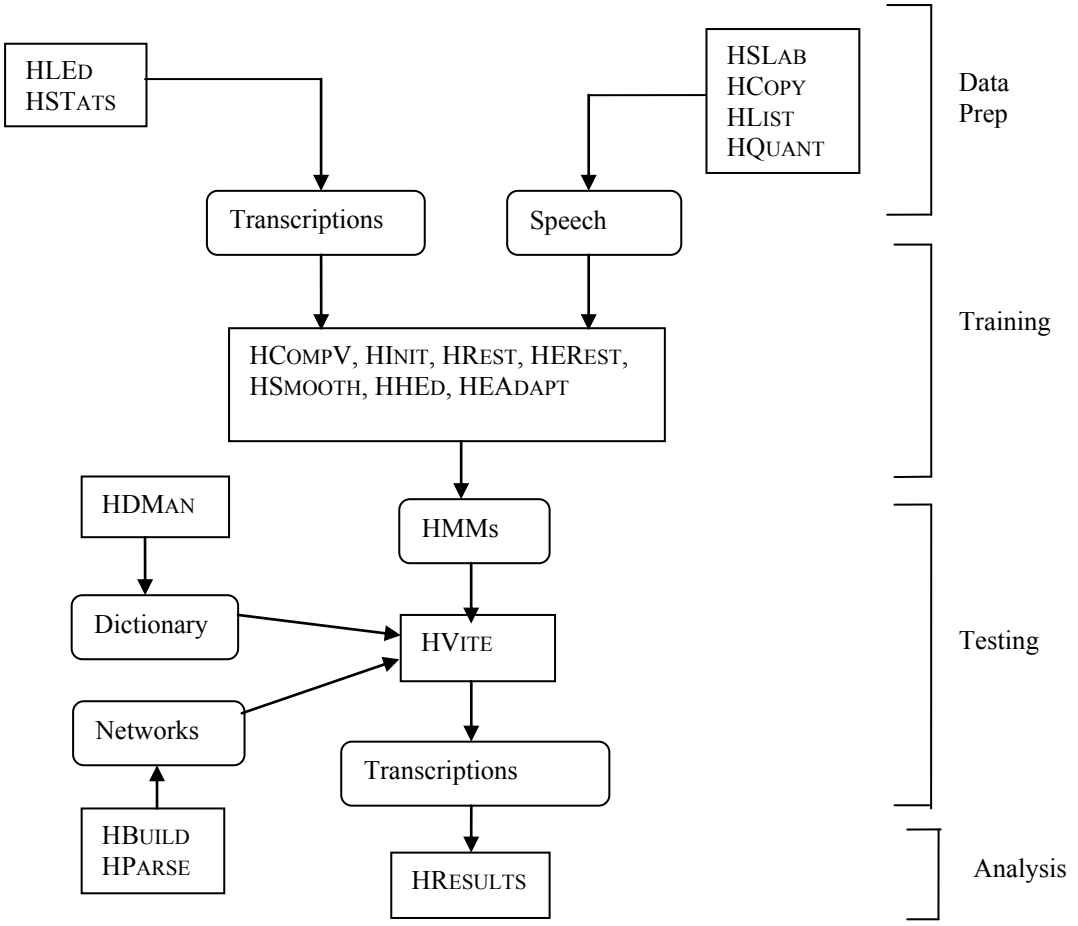

Fig 4. HTK Processing Stages

6. Results and analysis

The dictionary file consisting of the following syllables and their combinations was created.
{ba || cha || ing || da || tta || la || oo || ra || sshh || ma}

Table 1. Comparison table of the outputs from general and proposed method.

Testing word from dictionary	Words recognized by unmodified HTK	Words recognized by modified HTK
Chaba	Racha	Chaba
Racha	Racha	Racha
Lama	Racha	Lama
Baing	Baing	Baing
Ooee	Oo	Oo
Dala	Lama	Dala
Chabaka	Charaba	Chabaka
Oobama	Oobama	Oobama
Sshh	Sshh	Sshh

Table 2. Results obtained from the words recognized by modified HTK

Overall Results
SENT: %Correct=74.65 [H=67, S=87, N=90] WORD:%Corr=92.34, Acc=86.78 [H=690, D=49, S=353, I=626, N=750]

Table 3. Results obtained from the words recognized by unmodified HTK

Overall Results
SENT: %Correct=3.67 [H=3, S=87, N=90] WORD: %Corr=13.89, Acc=9.23 [H=98, D=49, S=353, I=30, N=750]

The first line of Table 4 and Table 5 gives the sentence-level accuracy based on the total number of label files which are identical to the transcription files. The second line is the word accuracy based on the DP matches between the label files and the transcriptions. In this second line, *H* is the number of correct labels, *D* is the number of deletions, *S* is the number of substitutions, *I* is the number of insertions and *N* is the total number of labels in the defining transcription files. The percentage number of labels correctly recognized is given by,

$$\%\text{Correct} = H / N \text{ X } 100\% \quad (1)$$

and the accuracy is computed by,

$$\text{Accuracy} = H - I / N \text{ X } 100\% \quad (2)$$

7. Conclusions and future work

A recognizer for similar sounding phone was implemented. The success rate was remarkable. The accuracy has gone up to 74% in the modified HTK from the initial 3% of the unmodified on the sample similar sounding phonemes. For similar sound, the following method also reduced the burden from adding a large number of trained samples.

The future work would involve to create a on the fly tool for HTK using the proposed method. A further contextual dependent approach to reduce the initial time complexity would be worked upon using a distributed concept mesh like the Conceptnet.[5]

Acknowledgement

I wish to express my sincere gratitude to Mr. B. R. Chandavarkar, Assistant Professor, National Institute of Technology, Surathkal, Karnataka, for providing me an opportunity to work on this project. This project bears on imprint of many people. I also wish to express my gratitude to my friends and family who rendered their support during the period of my project work.

References

[1] Zheng Chen. Hidden markov models and its application in automatic speech recognition; October 5, 2006.

[2] Z.Fodroczi. Hidden markov models and its application in automatic speech recognition, Pazmany Peter Catholic University.

[3] Young,Gunnar Evermann ,Mark Gales,Thomas Hain, Dan Kershaw, Xunying (Andrew) Liu et al, Htk book, version 3.4 March 2009

[4] Lidia Mangu, Eric Brill, Andreas Stolcke. Finding consensus in speech recognition: word error minimization and other applications of confusion networks, Computer Speech and Language 2000

[5] H Liu, P Singh. ConceptNet- a practical commonsense reasoning tool kit, BT Technology Journal, Vol 22 No 4, October 2004.