Confidence Measures for Spoken Dialog Systems

Submitted in partial fulfilment of the requirements for the degree of

Master of Technology

(*Communication & Signal Processing*)

By

Pranav Shriram Jawale

09307606

Under the guidance of

Prof. Preeti Rao

Department of Electrical Engineering

INDIAN INSTITUTE OF TECHNOLOGY BOMBAY

2012

Dissertation Approval

This dissertation entitled **Confidence Measures for Spoken Dialog Systems** by **Pranav Shriram Jawale** (Roll no. 09307606) is approved for the degree of Master of Technology in Communication & Signal Processing.

Prof. Preeti Rao _____ (Supervisor)

Prof. P. C. Pandey _____ (Examiner)

Dr. Samudravijaya K. _____ (Examiner)

Prof. E. Chandrasekhar _____ (Chairman)

22nd June 2012

Declaration

I declare that this written submission represents my ideas in my own words and where others' ideas or words have been included, I have adequately cited and referenced the original sources. I also declare that I have adhered to all principles of academic honesty and integrity and have not misrepresented or fabricated or falsified any idea/data/fact/source in my submission. I understand that any violation of the above will be cause for disciplinary action by the Institute and can also evoke penal action from the sources which have thus not been properly cited or from whom proper permission has not been taken when needed.

Pranav Shriram Jawale

09307606

June 22, 2012

Acknowledgments

I thank Dr. Preeti Rao for giving me the opportunity (and freedom!) to work on the Marathi AgroAccess (ASR) project. Also I thank her for the countless advices on conducting a research project. I thank Dr. Samudravijaya K. for many insightful research discussions and encouragements.

I thank Tejas Godambe, for being a great colleague in the ASR project. We have had numerous (at times hour-long) telephonic discussions regarding even the minute details of IVR system. I also thank the other members of the ASR project – Dr. Nandini Madam, Tauseef, Nikita, Nikul, Pinki, Asha, Minit, Namrata and Joel – for a lot of things. Special thanks to Nikul for being a late-night lab partner. Special respect for Nikita, Asha, Minit, Joel and Tejas who (apart from other tasks) handled speech transcription, which is among the most daunting tasks in this project.

I thank the many members of DAPLAB - Sachin, Chitra, Pradeep, Sujeet, Shrikant, Vishu, Veena, Vaishali, Bhave, Srinivas, Arpita, Rohan, Mayank, Hari, Amruta, Joe, Vinutha Madam and Prateek. Also I thank Mr. K. S. Nataraj from across the corridor for being a motivational speaker.

Also I thank my favourite teachers at IITB - Prof. Vikram Gadre, Prof. D. Manjunath, Prof. Pushpak Bhattacharyya and Prof. B. K. Dey - for inspiring and enlightening interactions.

Finally, I thank my parents for always supporting me.

Pranav Shriram Jawale

Abstract

This thesis is motivated by a speech based access system for agricultural information (AgroAccess). In such systems it is important to know when the speech recognizer has made a mistake and adapt the conversation accordingly. We investigate different components that can be used to improve the performance of the dialog system. One of the problems addressed here is the implementation of a keyword spotting system along with effective post-processing schemes to reduce the number of false alarms. In the latter half of the report we concentrate solely on the different confidence measures that have been investigated and in some cases improved upon. We consider confidence measures based on acoustic score normalisation and N-best list evidence. We also describe an initial version of the utterance level confidence measure computation block for the AgroAccess system.

Table of Contents

List of Figures

List of Tables

Chapter 1. **Introduction**

1.1 Motivation

Traditionally human agent / touch-tone based systems are used for providing service to the users through phone calls. Since it is difficult to handle large call volume with human agents, touch-tone systems were developed wherein users navigate through a series of alternatives using key presses before getting the required information. But the touch-tone systems become cumbersome when the choices are far too many to be selected via key presses (e.g. choosing destination airport using an airline ticket reservation system). With the advent of speech recognition technology it is a viable option to implement a dialog system for such purposes. Here, the users respond to the questions asked by the system and their answers are sent to a speech recognition and understanding unit so that a suitable action can be taken. Together, IIT Bombay and TIFR have built one such dialog system (referred to as **AgroAccess** system in this report) in Marathi for providing prices of agricultural commodities to farmers across Maharashtra. This report is about some of the research problems addressed while building the system.

1.2 Marathi AgroAccess system

1.2.1 Background

For the people involved in agricultural commodity business, it's important to know the prices of various commodities in local markets (mandis). Indian Ministry of Agriculture has a web portal (http://agmarknet.nic.in/) for disseminating the latest commodity prices in local markets from all over India. The user can access the prices through a graphical user interface (Figure 1). For uniformity, the names of commodities are written in English. An internet based interface like this may be difficult to access for an illiterate farmer. AgroAccess system tries to improve upon this by using a speech interface. Of course, since speech recognition is not 100% accurate, the system is fallible. There are challenges such as different pronunciation styles (different accents due to dialectal variation) and non-stationary environmental noise. Through better dialog design, restricted vocabulary, and adopting/improving upon the existing techniques in speech research the system is being improved.

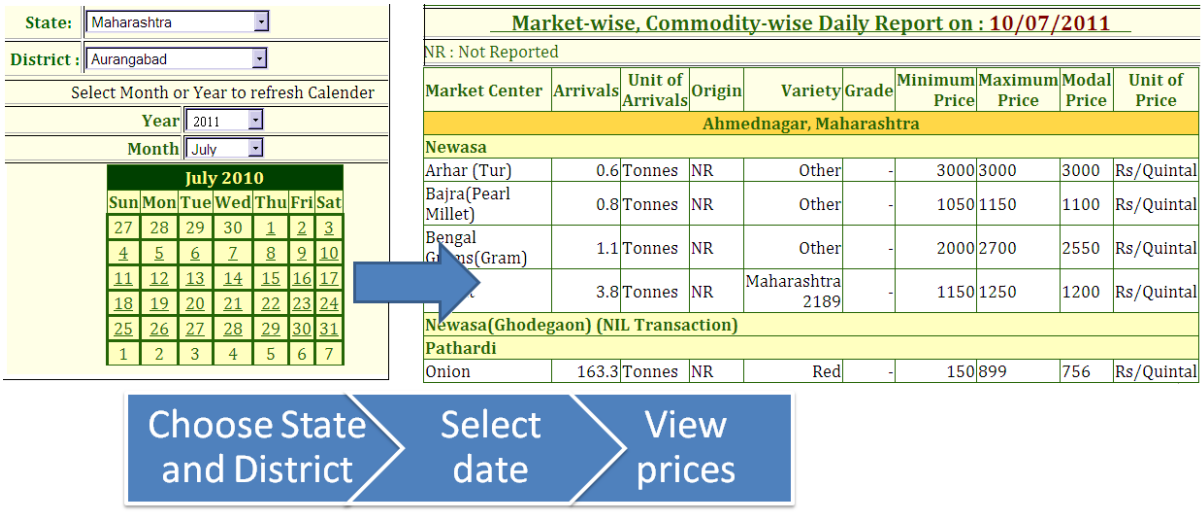

Figure 1: Agmarknet visual interface example. (Screenshots taken from http://agmarknet.nic.in/)

1.2.2 System components

Main software and hardware components of the system are shown in Figure 2. The system

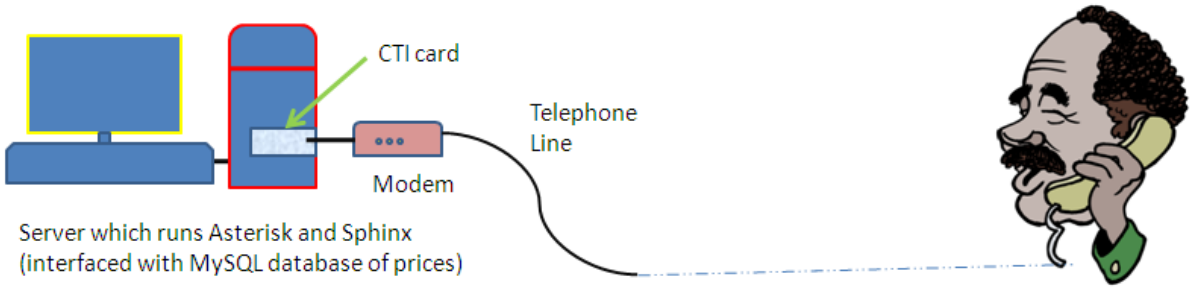

Figure 2: Various hardware and software components of AgroAccess system (Image partially taken from http://www.clipartmojo.com/)

consists of a (Computer Telephone Interface (CTI) card connected to the telephone line via a modem. An open source software **Asterisk** [1] has been used to handle the incoming calls. A daily updated **MySQL** [2] database stores the reported prices of various agricultural commodities in all the mandis of all the districts of Maharashtra. There are overall 34 districts, 280 mandis and 190 commodities in Maharashtra as reported on the Agmark website (crawled from yr. 2004-2011). Whenever a user calls, s/he responds to system's questions and gets the price of intended commodities in a desired Mandi. We use **CMUSphinx toolkit** [3] to build our speech recognition system.

1.2.3 System call flow

Overall call flow of AgroAccess is shown in Figure 3. There are 3 main nodes in the system call flow; namely district, mandi and commodity node (in that order). The system (through a dialog with user) first determines the user's district, then mandi and then the intended commodity. We follow this order to reduce size of recognition searchspace (~ language

model) at each node. For example, overall there are 196 commodities but on an average there are only 22 commodities per mandi. So, the commodity searchspace is reduced by using a mandi dependent language model (LM). This language model is created based on commodities reported in that mandi over the last 8 years (yr. 2004-2011).

At a higher level, the user's speech is first sent to a recognizer and the output of the recognizer (a string of words) is further processed by the system to decide how to respond to the user. For example, when a commodity name is recognized, the system accesses the MySQL database and obtains the price information. Based on this decision, the system responds by playing concatenated pre-recorded speech prompts (rudimentary speech synthesis).

1.2.4 System vocabulary

The districts are single word entities whereas mandis and commodities are multiword entities. Number of words in a mandi's name varies from 1 to 3. The commodity names sometimes contain acronyms (e.g. LRA, DJVI). If acronyms are considered as a single word then commodity names contain from 1 to 6 words. Overall there are ~1000 words in the system vocabulary but only those words that are present in the language model at a particular node can be recognized by the system.

1.2.5 Design problems

While building a dialog system application one faces the trade-off between having human-like interaction and the recognizer accuracy. If the user is given too much 'freedom of speech', then the recognizer accuracy degrades drastically.

Response Validity Check block: One important block of the call flow is the one which determines "*Should the user response be accepted?*" This block tries to deal with the error types in Table 1. All these 6 types of errors depend on the user behaviour, but to further compound the problem, the recognizer itself may make mistakes. With each recognizer hypothesis we associate the degree to which we trust it. This degree is called as confidence measure in speech recognition.

Table 1: Types of error creating situations

Error Type	Description
1	The speaker didn't say anything, but there is background noise / babble which can get decoded as a valid sequence of words.
2	Each word spoken by the speaker is out-of-vocabulary for the system.
3	The user spoke out of vocabulary words along with the in-vocabulary words (e.g. *Hello, what's the price of tomato?*)
4	Speech disfluency, false-starts
5	Speaker uttered the intended phrase only partially (e.g. *Vaashi Navi* instead of *Vashi Navi Mumbai*)
6	Speaker spoke words in a phrase in a different order than that assumed in the system (e.g. *Tomato Red* instead of *Red Tomato*)

Research problems that arise from these error creating situations are as follows –

1. **Classification problem:** Here, we classify the recognition output as either correct or incorrect (*not to be processed further*) based on an utterance level confidence measure. If it is incorrect, then we again ask the user the same question. Error type 1, 2 and 4 can be handled with such binary decisions.
2. **Information extraction problem:** Here, we don't directly pass a binary judgement on the complete recognition output, but rather try to find out the intended name of item (district/mandi/commodity) spoken by the user. The error types 1, 2, 3 and 4 (Table 1) can be handled by a keyword spotting system (+ confidence measures). The error types 5 and 6 can be handled using task specific linguistic knowledge (in addition to other techniques).

For handling these errors different structures for *Response Validity Check* block can be used. These make use of three of the different types of search graph structures (described next) that we can use/implement with the CMU-Sphinx toolkit.

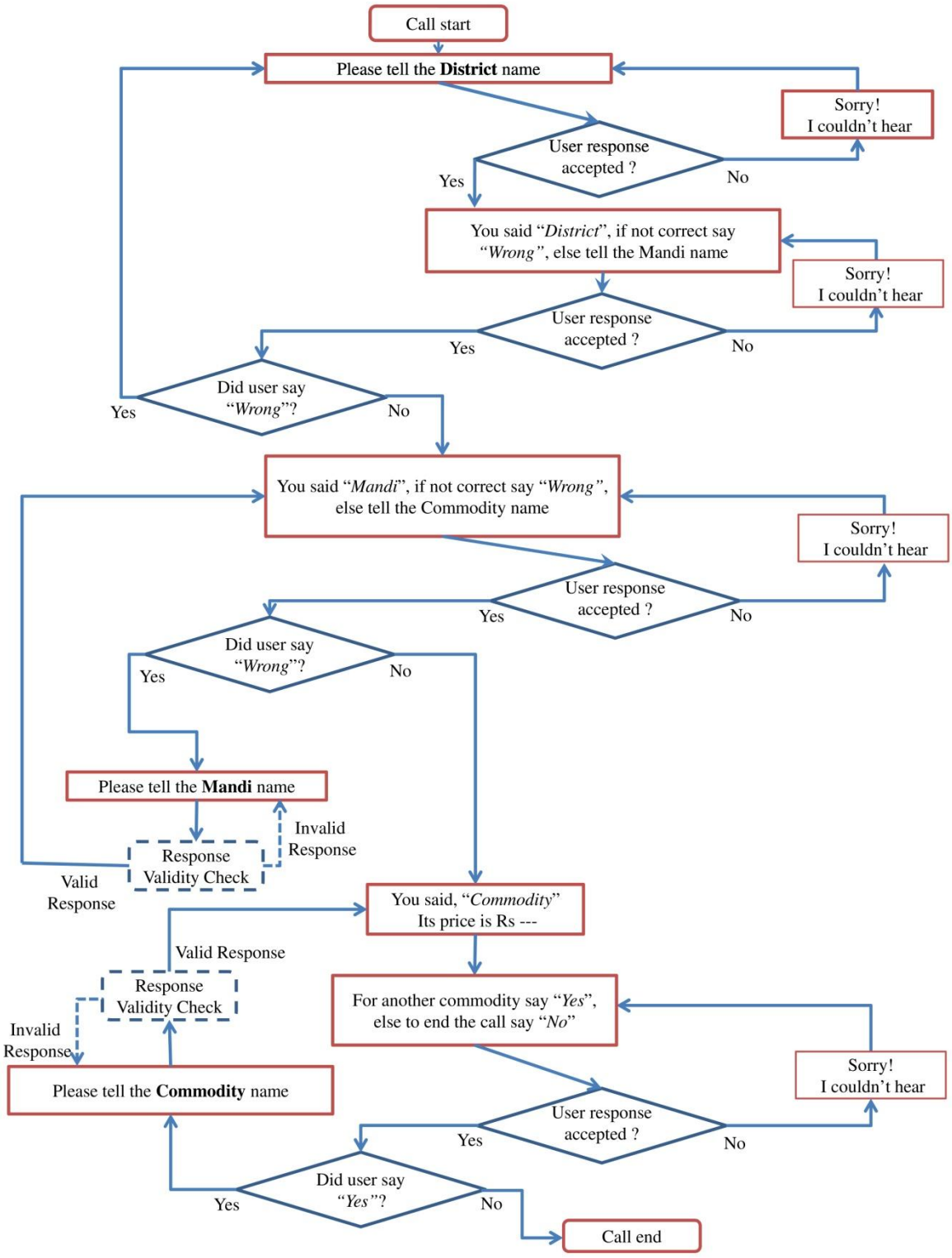

Figure 3: AgroAccess system call flow

1.2.6 Language model and search graphs

Speech recognition can be considered as a Viterbi search over a word/phone graph. These word graphs are constructed based on the particular type of language model fed into the

decoder. Shown in Figure 4 is a (partial) search graph structure for a three word vocabulary using trigram language model (no backoff). In an n-gram language model, prior probability of a word is learned from a training transcription and it depends on n-1 previous words. If some trigrams are absent in the training transcription then those trigram probabilities are approximated using bigram probabilities. This type of LM is called as backoff trigram LM.

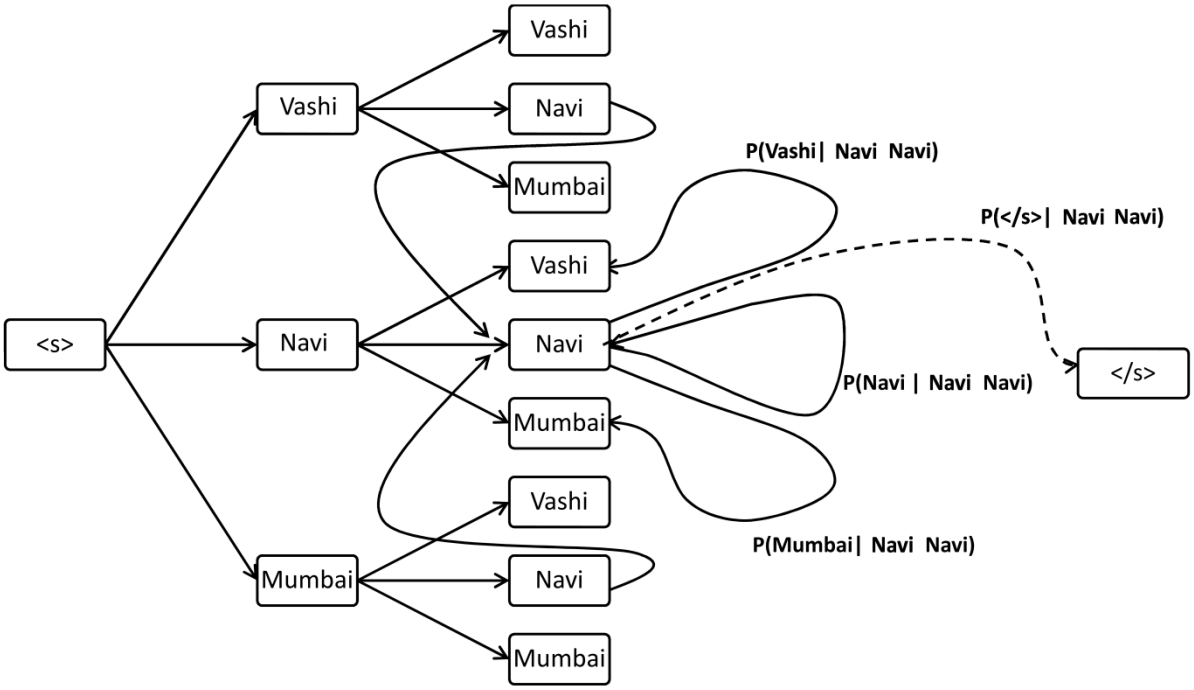

Figure 4: (Partial) Trigram search graph

If one is aware of the exact word sequences that the user will be speaking, then a simplified language model called as Finite State Grammar can be used. Figure 5 is an example of search graph for a finite state grammar for above trigram case. By default, all the branches emanating from a node are equiprobable (i.e. not learnt from training transcription). Note another important feature of this particular structure, there is no loop back. Only finite length word sequences can be recognized with this graph. Also sometimes the decoder can output a partial phrase or no phrase at all depending on the speech content. Even though we have not shown explicitly, in both n-gram and FSG based search, fillers (non speech sounds, including silence) can be recognized in between any two words.

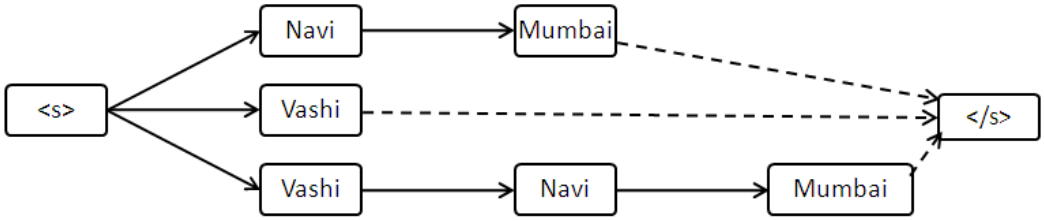

Figure 5: FSG search graph

There is another possibility, though traditionally it's not referred to as a language model as it can be implemented using a finite state grammar. It is a keyword spotting technique in which the search graph includes an optional allphone loop is present before and after the keyword (or key-phrase) node (see Figure 6). It can be written as <phone>* key-phrase <phone>* (* denotes zero or more). This kind of search graph can absorb any OOV words into the allphone loops.

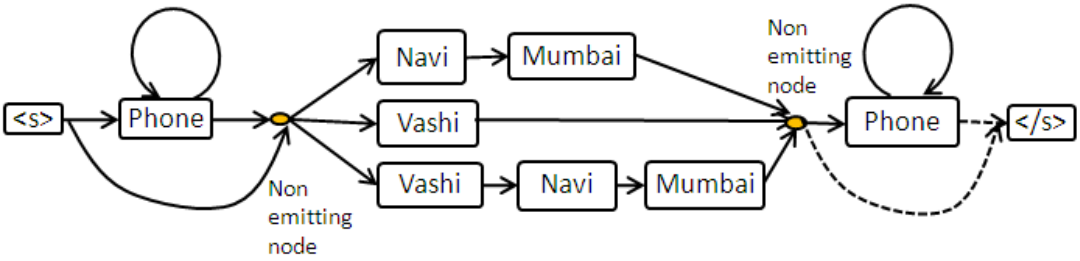

Figure 6: Keyword spotting search graph

In Chapter 6 we describe Version 1.0 of the Response Validity Check block. The basic idea is that we use a **preliminary hypothesis generator** to generate one or more possible hypothesis phrases. A confidence measure computed for each word in each hypothesis is used to accept one of the hypotheses or reject all of them.

Various preliminary hypothesis generator blocks differ mainly in the structure of the Viterbi search network and the network parameters (e.g. transition probability). We can use 3 structures, namely, Finite State Grammar, Trigram language model and keyword spotting (word-filler network). The keyword spotting network has the ability to ignore out of vocabulary words in the search process itself.

1.3 Thesis structure and contributions

In **Chapter 2** we review the literature on keyword spotting and various confidence measures. Experiments for building and evaluating acoustic keyword spotting framework are described in **Chapter 3**. In **Chapter 4 and 5** we describe the implementation and obtain the performance of various word level confidence measures computed using following techniques based on

- Phone recognition based acoustic score normalisation (**Chapter 4**)
- N-best list based confidence measures (**Chapter 5**)

In **Chapter 6**, we describe the implemented Response Validity Check block. We conclude with Chapter 7.

Thesis contributions are listed in Table 2.

Table 2: Thesis contributions

Problem considered	**Contribution**	**Offline evaluation**	**Integrated with AgroAccess?**
Keyword spotting + CMs	Implemented basic acoustic KWS framework + two post-processing modules (CMs)	Done on a small sentence database, with 4 keywords	No
Confidence measures in Medium Vocabulary speech recognition	Implemented a class of CMs + proposed some variants	Done on AgroAccess speech database	V 1.0 up and running.
More intelligent information extraction	-	-	No

Chapter 2. **Literature Review**

In view of the nature of 'Response Validity Check block' that was introduced in the first chapter, here we review the related previous efforts/achievements reported in literature. Initially, I started working on the keyword spotting problem. As noted in Section 1.2.5 a keyword system can help under the cases in which user embeds the intended item name with other OOV words. At that time the database for Agro-Access system was yet unavailable (the system building was still in the initial stages). As the field data became available, I concentrated on implementing and evaluating various word level confidence measures. These confidence measures can be applied to prune the hypotheses generated by a keyword spotting system. The literature review has been divided in two parts, Sections 2.1, 2.2 and 2.3 are concerned with the review of the major approaches to KWS problem, followed by a description of performance metrics for a keyword spotting task. Section 2.4 reviews confidence measures in general. It is followed by a discussion on choices to be made while choosing confidence measure for a dialog system application.

2.1 KWS problem definition

The keyword spotting (KWS) task is to identify occurrences of certain keywords in an input speech stream. The accuracy with which the keyword boundaries are hypothesized is not that important but from an evaluation point of view, more than a certain amount of overlap (decided by the system designer) with the actual spoken keyword is required. During this process the KWS systems generate certain confidence scores associated with each hypothesis. The confidence scores are computed according to different confidence measures employed and are aimed at discriminating between true hits and false alarms.

2.2 Approaches for keyword spotting

There are 3 Main approaches to keyword spotting as described below.

2.2.1 Large Vocabulary Continuous Speech Recognition (Word Index KWS)

This is the most obvious approach to keyword spotting. The Large Vocabulary Continuous Speech Recognition (LVCSR) systems aim at recognizing all the words spoken in the input speech. The KWS task reduces to searching in the word level hypothesis produced by LVCSR system [10]. The recognizer output could be either in the form of 1-best hypothesis, N-best

list or word lattice. All of these can be termed as word level indexing methods. An example of a word lattice is shown in Figure 7.

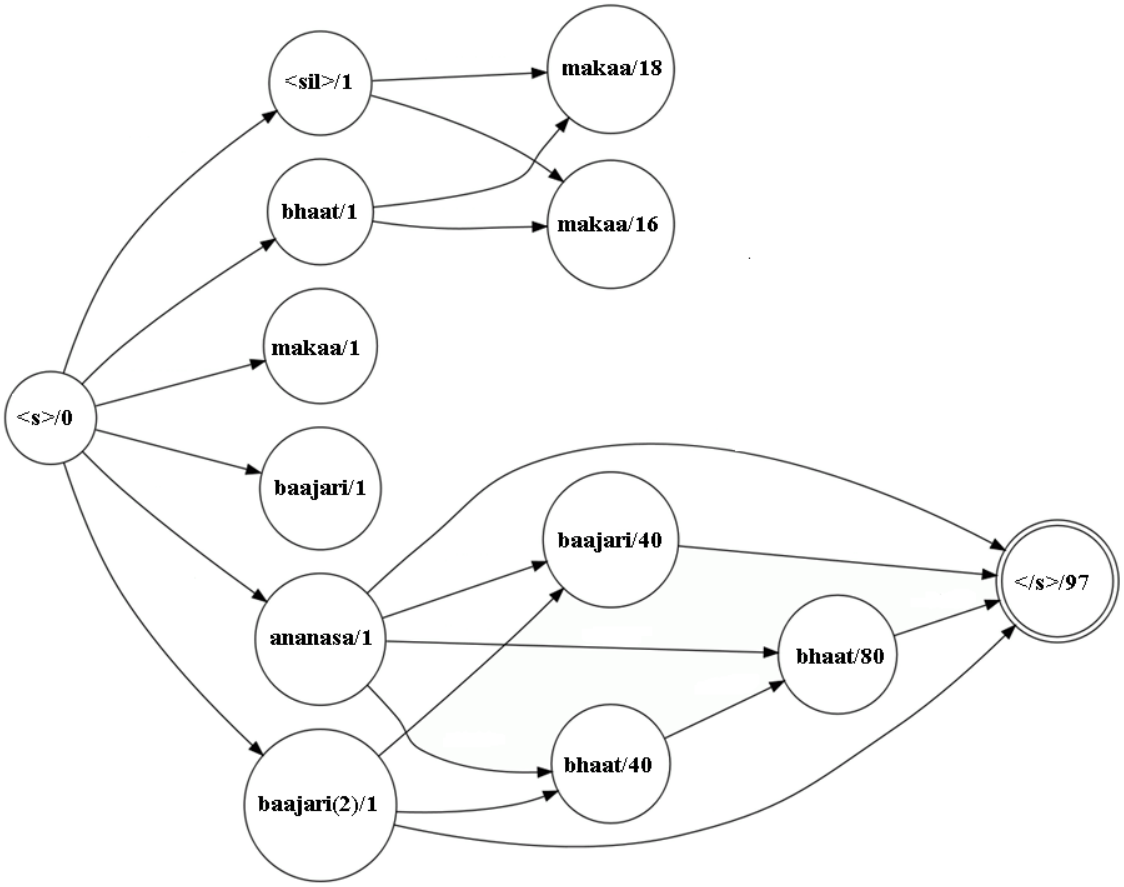

Figure 7: A word lattice created after recognition; in each hypothesis (denoted by a circle) the recognized word and its starting frame are shown; <s> and </s> denote start and end of the utterance respectively. Image created by porting the lattice file to GraphViz software.

In the context of keyword spotting from large audio archives, word level indexing is done offline. When the user queries for a keyword, an online search is performed through the word index. One drawback is that if the queried keyword is absent from the recognizer vocabulary, it is impossible to locate it. Hence a large recognizer dictionary has to be maintained and updated from time to time. Another drawback of LVCSR is that a statistical language modelling of speech has to be done to improve the word recognition performance. Large amount of domain specific text data is required for language modelling.

2.2.2 Indexing and searching sub-word content (Phone Index KWS)

In this approach, phonetic content present in the speech is encoded in the form of 1-best or phone/syllable lattice. The phone sequence corresponding to the keyword to be searched is

first obtained either from a predefined dictionary or by using a grapheme-to-phoneme converter. Phone recognition accuracy is generally lower than word recognition due to lack of knowledge about plausible phone sequences. To account for errors in phone recognition, an approximate search is run through the phone index. James et al. [5] have proposed a dynamic programming based lattice search technique. Here, the keyword phones were labelled as either 'strong' or 'weak'. The strong phones are those which must be present in hypothesized lattice segment whereas weak phones could be deleted or substituted. Further improvement in lattice based search was proposed in [6]. Here, each phone sequence in the lattice is scored against the keyword phone sequence using Minimum Edit Distance (MED, also known as Levenshtein distance) metric. A threshold is kept on MED to compensate for the phone recognition errors.

2.2.3 Acoustic keyword spotting (Acoustic KWS)

Acoustic KWS uses the continuous speech recognition framework with a crucial difference that instead of trying to recognize all the words in speech (like LVCSR), models are build for keywords and all the non-keyword speech in general. Roehlicek et al. [7] proposed a HMM recognition network in which keyword and non-keyword models are kept in parallel in the recognition network. They trained whole-word models for keywords using keyword instances in the training data. The non-keyword models were trained on segments from keyword instances themselves. As they are kept in parallel, keyword and non-keyword models compete with each other during recognition. Rose and Paul [8] generalized this system and modelled the keywords as concatenation of subword models. They proposed various methods for building non-keyword (filler) models, such as training filler models specifically on non-keyword speech and using a monophone loop (all monophones in parallel) structure as a model of non-keyword speech. More details about Rose and Paul system are in Section 3.4

For good performance, the filler models should match the non-keyword speech more closely than the keyword models. In order to reduce the number of keyword false rejections, in [9] it has been proposed to use filler models from another language. For their Japanese KWS system they build keyword models from Japanese triphones and filler models from English monophones. Their results show that while phone HMMs trained on English cover the Japanese non-keyword intervals efficiently, they do not absorb the keywords (causing false rejections) when competing with Japanese KW models.

2.2.4 Comparison of the three approaches for keyword spotting

See Table 3 for a brief comparison of the three KWS approaches. In [10] Szoke et al. have compared LVCSR, phone-lattice and acoustic KWS techniques. They found that the best accuracy is provided by LVCSR system by searching in word lattices and refining the search results using a confidence score derived from forward and backward lattice path likelihoods. The FOM performances were found to be in the order LVCSR > Acoustic KWS >> Phone lattice KWS. In [11] Szoke et al. have proposed a hybrid word-subword spoken term detection (general form of KWS in which a multiword query is searched for) system which merges the word and phone lattices and is shown to perform better than the individual systems in isolation.

Table 3: Comparison of LVCSR, Phone index based and Acoustic KWS

	LVCSR	Phone index based KWS	Acoustic KWS
Vocabulary	Pronunciations of both keywords and non-keywords have to be kept in the dictionary	Only phones	Only keyword pronunciations have to be kept in the dictionary
Offline task	Word indexing	Phone indexing	The KWS process has to be re-run on the whole audio when new keywords have to be searched.
Online task	Searching through word index is very fast but disadvantage is that any keyword not part of vocabulary cannot be recovered.	Fuzzy search through phone index has to be performed which is slower, but advantage is that any keyword can be searched	
Critical issues	Domain specific Statistical language modelling, comprehensive vocabulary	Phone recognition accuracy	Modelling of non-keyword speech
Suitable for searching in	Large audio archives and short utterances in spoken dialog systems when the vocabulary is very comprehensive and statistical language model is available.	Large audio archives where no knowledge about keyword/ non-keyword vocabulary is available.	Short utterances in spoken dialog systems.

2.3 Performance metrics for KWS

In keyword spotting literature various kinds of metrics have been used for performance evaluation. Here we briefly define each of them.

Input to a keyword spotting system is speech and a keyword set (list of keywords to be searched for in the speech). The output of a keyword spotting system is a list of keyword occurrences with start time and end time information for each occurrence. If there is more than a certain amount of overlap between a hypothesized keyword and corresponding keyword in the ground truth, then it is considered as a true hit, otherwise a false alarm is declared. Overlap value is the choice of the system designer and is usually > 50%.

The keyword spotting system aims at finding all occurrences of all the keywords from a keyword set. If there are total N_G keywords occurrences in the ground truth and N_H of them have been detected at a certain system operating point, then the **hit rate** (HR) is given as in Equation 1.

$$HR = 100\frac{N_H}{N_G} \tag{1}$$

False rejection rate (FR) is the converse of hit rate and is given as $(100 - HR)$. The **False alarm rate** (FA) of a KWS system at a particular operating point is given as in Equation 2

$$FA = \frac{N_F}{S_{KW}T} \tag{2}$$

Here, N_F is the total number of false alarms that occurred in a test dataset of duration T hours, S_{KW} is the size of the keyword set. Unit of false alarm rate is FA/KW/Hour. It is better to include approximately equal number of occurrences of all the keywords in the test dataset; otherwise the results could be biased by performance on a few keywords from the keyword set only. An alternative measure of incorrect detections is **false acceptance rate** (FAR) which is computed as in Equation 3.

$$FAR = 100\frac{N_F}{N_G} \tag{3}$$

The **Figure of Merit** (FOM) [7] of a KWS system is defined as the average hit rate taken over false alarm rates ranging from 0 FA/KW/Hour to 10 FA/KW/Hour. Higher the FOM, better

the system is. FOM can be used compare systems which output a confidence score / which have some easily tuneable system parameters. FOM is not a good measure for comparing systems where achieving very high hit rate is more important than very low false alarm rate. This is because usually operating regions for very high hit rate and very low false alarm rate are separate.

The **Receiver Operator Characteristic** (ROC) curve for a KWS system is obtained by plotting 'hit rate Vs false alarm rate' or 'hit rate Vs false acceptance rate'. It gives a graphical indication of system performance over a large range of tuning parameters.

Equal Error Rate (EER) is another metric for comparing tuneable KWS systems. It is defined as the false rejection rate (*FR*) at that point on the 'false rejection rate Vs false acceptance rate' curve where *FR* = *FA*. Lower the EER, better the system is. Though it can be used to compare two systems, generally KWS systems are not operated at the EER.

2.4 Review of confidence measures

Definition: Confidence measure (CM) denotes the degree to which a recognition hypothesis is to be trusted. Depending on the feature(/s) used, CM may be confined to [0, 1] range, or it may take any real value. Confidence measures can be computed at phone, word or sentence level. In the literature, the confidence measure problem appears in various forms (as a post processing scheme in keyword spotting [8], for finding new words in the lexicon [12], utterance verification [13], dialog management [14], [15] and as a standalone problem by itself [16]. Below we give an overview of various classes of confidence measures. Surveys on this problem can also be found in [17], [18].

2.4.1 Decoder based features and their combination

During decoding (and during latter passes in a multipass recognition system), the Viterbi search process generates different kinds of information which can help to distinguish correct hypotheses from incorrect hypotheses. These features are listed in Table 4. Here, we are only referring to word level CMs.

Table 4: List of Decoder based features

Sr No.	**Feature**	**Description and comments**
1	Number of phonemes in word [19]	This is *not* a decoder generated feature, but still is a valid predictor of correctness. It is observed that longer words (those with more phonemes) are more often decoded correctly than shorter words. *Greater the number of phones, more the confidence.*
2	Word duration [16], [20], [21]	A word hypothesis is likely to be wrong if the hypothesis duration deviates too much from its mean duration (which may be computed from the mean of durations of constituent phones). *Smaller the deviation from mean duration, more the confidence.*
3a	Normalised log-likelihood acoustic score [8]	It's the duration normalised log likelihood acoustic score of a word. *Higher the score, more the confidence.*
3b	Mean log-likelihood score [14]	Average log-likelihood (i.e. acoustic score) of all repeating and overlapping instances of a particular word hypothesis in an n-best list. *Higher the score, more the confidence.*
3c	CM based on probability distribution of acoustic likelihood scores [16]	Since best possible likelihood score for one GMM may differ from another, rather than using raw likelihoods, we need to compensate for this difference between GMMs. It helps to compute area under the likelihood distribution, between mean of the distribution and raw likelihood obtained from the recognizer. *Smaller the area, more the confidence.* This CM is obtained at frame level, and augmented to give word level score.

4	Standard deviation in loglikelihood scores [14]	Standard deviation in acoustic scores across *all* the nbest hypotheses within the duration of the word hypothesis. *Higher the deviation, more the confidence.*
5	Weighted combination of acoustic and language model scores [22]	*Higher the combined score, more the confidence.*
6	N-gram language model backoff behaviour [23], [24], [25]	Based on whether the word trigram/bigram existed in the training data, a score is given to the word hypothesis (trigram gets highest score). *Higher the score, more the confidence.*
7a	N-best purity / N-best word-rate [14], [26]	Fraction of the n-best list hypotheses in which the word appears in (roughly) the same time duration. *Higher the fraction, more the confidence.*
7b	Measure of N-best impurity [27]	A measure of how similar the phones hypothesized in the n-best hypotheses are to the phones in top word hypothesis. The measure could be phonologically based or based on phone confusion matrix on training data. *Greater the value of similarity metric, more the confidence.*
8	Difference between acoustic (+ possibly language model) scores of adjacent n-best hypotheses [28]	*Higher the difference between top and next-best hypothesis, more the confidence on the top hypothesis.*
9a	Hypothesis density at the word beginning, word end [19], [29]	*Larger the number of different hypothesis in a word lattice at specific time (word beginning/end) more the confidence of the word hypothesis is low.*
9b	Average hypothesis density [29]	Hypothesis density averaged over complete word duration. *Greater the density, more the confidence.*

9c	Number of active senones [30]	Number of active senones (i.e. those above pruning threshold) near the hypothesized word end. *Larger the number, smaller the confidence.*
9d	Phone perplexity [25]	Average number of phones searched along the frames where the recognized word is hypothesized. *Larger the number, smaller the confidence.*
10	Acoustic score entropy [30]	Framewise entropy of acoustic scores computed over complete phoneset; averaged over the word duration. *Higher the entropy, lesser the confidence.*
11	Acoustic stability / LM jitter [27], [30]	Word lattice is rescored multiple times (approx. equivalent to decoding the test utterance multiple times) for different combinations of the language weight and word insertion penalty. Fraction of times a hypothesized word occurs in (roughly) the same time segment across the multiple hypotheses is computed. *Larger the fraction, more the confidence.*
12a	Phonetic match at frame level [26]	Two decodings, one at word level another at phone level are performed. Percentage of frames in which phone in the word hypothesis matches with the phone in phone level hypothesis indicates confidence in the word hypothesis. *Greater the percentage, more the confidence.*
12b	Phonetic match at phone level [26], [27]	Same as above, but the percentage is computed over number of phones, instead of number of frames. Phonetic match is can also be computed using phonologically based similarity measure and/or phone confusion matrix based distance measure. *Greater the percentage, more the confidence.*

13	Distance between HMM states [31]	Two decodings, one at word level another at phone level are performed. Using Kullbac-Leiber measure, frame level distance between state level hypotheses of word and phone decoders is computed and averaged over the word duration. *Smaller the distance, more the confidence.*

As can be seen from Table 4, the number of decoder based features is large. There is a gradation across these features but none of these features is very good/self-sufficient in terms of performance. Hence, some authors have tried to combine some of these features (and some other features based on posterior probability computation) using machine learning techniques. Chase [22] has experimented with 4 different ways of combining features, namely decision trees, generalised linear models (GLMs), generalised additive models (GAMs) and neural nets and found that GAMs perform better than the rest. Zhang and Rudinky [26] combined the features using Support Vector Machines, decision trees and neural nets and reported that SVMs performed better. In [30] neural nets and linear classifiers were used for combination and neural nets were found to work better.

2.4.2 CMs based on posterior probability computation

The decoder hypothesizes a word sequence hypothesis based on Maximum A posteriori Criterion (MAP) rule. Let this hypothesis be denoted by $\widehat{W}$. If O is the observation vector of MFCCs corresponding to input speech and $\boldsymbol{\Psi}$ is the set of all possible word sequences then

$$\widehat{W} = \underset{\boldsymbol{\Psi}}{\operatorname{argmax}}\, P(W|O) \tag{4}$$

Here, $\mathrm{P(W|O)}$ i.e. posterior probability of the word sequence, can be computed as

$$P(W|O) = \frac{P(O|W).P(W)}{\sum_{\boldsymbol{\Psi}} P(O|W)\,P(W)} \tag{5}$$

The denominator term in above equation as it remains constant, irrespective of what the word hypotheses are. Hence it is ignored during decoding; nevertheless, it is required for computation of posterior probability of hypothesized word sequence and is a good candidate

for confidence measure. The posterior probability can be computed in different ways as below.

Using parallel phone decoder [8], [10]**:** Equation 5 can be approximated as (assuming each word sequence is equiprobable and relaxing the constraints on possible phone sequences)

$$P(W|O) \approx \frac{P(O|W)}{\sum_{\boldsymbol{\phi}} P(O|\boldsymbol{F})} \tag{6}$$

Here, $\boldsymbol{\phi}$ is the set of all possible phone sequences. The denominator in the above equation can be further approximated by the maximum term in the summation as below

$$P(W|O) \approx \frac{P(O|W)}{\max_{\boldsymbol{\phi}} P(O|F)} \tag{7}$$

The denominator $\max_F P(O|F)$ can be computed using a phone/filler decoder. If $\widehat{W}$ is the word sequence $(w_1, w_2, \ldots, w_n)$ then, for each word, the posterior probability can be computed as in Equation 8.

$$P\left(w_i|O_{w_i}\right) \approx \frac{P\left(O_{w_i}|w_i\right)}{\max_{\phi} P(O_{w_i}|F)} \tag{8}$$

Using N-best list [32], [33]: N-best list is a list of hypotheses having total score (acoustic + language model scores) within a certain threshold of the total score of the top hypothesis. It is created by sphinx3 decoder in two steps. First, a word lattice is created during Viterbi search process. Next, the n-best list is created by processing the word lattice with A* search. N-best list can be used to given a confidence measure as described next. Equation 5 can be approximated using an n-best list as follows,

$$P(W|O) \approx \frac{P(O|W).P(W)}{\sum_{\boldsymbol{W_{nbest}}} P(O|W)\,P(W)} \tag{9}$$

Here, $\boldsymbol{W_{nbest}}$ is the set of utterance level hypothesis from n-best list. If a particular word belongs to a subset of n-best list (in approximately the same location) then the following confidence metric based on n-best list (C_{nbest}) seems reasonable to use

$$C_{nbest}(w_i) = \frac{\sum_{\boldsymbol{W_{nbest}}: w_i \in W} P(W|O)}{\sum_{\boldsymbol{W_{nbest}}} P(W|O)} \quad (10)$$

Using word lattice: Wessel et al. [32], [34] computed the posterior probability more sophisticatedly using the word lattice which is a more compact representation of all possible word hypotheses. They computed it by using a forward backward algorithm to compute the ratio of sum of the scores of all paths going through a node, normalized by the sum over all paths through the lattice. In [32] they show that this approach performs better than n-best list based approach. But note that this approach is more computationally expensive as compared to n-best list approach.

In general, on a LVCSR task (e.g. large lattice size) it is accepted that word lattice based CM outperforms all other posterior probability based methods as well as the other decoder based features. Of course, their combination always helps.

2.4.3 CMs based on statistical hypothesis testing formulation

These CMs mainly come from utterance verification literature. Here the problem is formulated as a choice between two complementary hypotheses.

H_0: The hypothesized word is correct

H_1: The hypothesized word is incorrect

A likelihood ratio test is performed by taking ratio of likelihoods of these two hypotheses. In [13] Sukkar has proposed a discriminative training technique for modelling subword HMMs. Here, corresponding to each subword, an anti-subword model is trained. During keyword verification a likelihood ratio between null and alternative hypotheses is computed for each component subword using subword and anti-subword models.

The anti-model of a subword is discriminatively trained on the speech that was misrecognized as that subword. The confidence score of a keyword hypothesis is taken to be average of all the confidence scores of the component subwords. It was shown that this system performed better than the likelihood ratio scoring method [8] based on taking difference between word and sub-word recognition output scores.

2.4.4 CMs based on additional features from acoustic signal

1. **Phone specific features:** In their keyword spotting system Ma et al. [35], [36] used an artificial neural network (ANN) based phone recognizer to provide phone information and

manner attributes for each small segment within a word hypothesis. As a first step they use phone durational constraints to reject some false alarms. Next, they perform manner and place attribute based rejection based on some manual rules on frequent confusion pairs in development set. For example, they observed that sometimes "nine" is getting recognized as "one". So those segments hypothesized as "one" whose manner attribute sequence didn't contain glide are rejected. They also used signal processing based features. For example, when they observed that "five" and "nine" are getting confused often they used low frequency energy ratio and a voicing detector to decide whenever either of them was recognized. Similarly, they used different spectral features *specific to each confusion pair* as a confidence measure.

2. **Other spectrographic features:** In [37] they present a method to capture patterns of high-energy tracks (seams) in spectrograms. They hypothesize that these seams could potentially carry relatively invariant signatures of underlying sounds. Their task is discriminative word spotting for which they train these patterns on exemplars of words in vocabulary and classify a word based on an SVM classifier.

2.5 Which CMs to use?

One should choose the CMs given constrains of the task. Following questions are needed to be answered for a good choice of CMs

1. Is the vocabulary size so small (and fixed) so that word specific discriminative training can be done?
2. How much decision delay can be afforded (What is the computational effort involved in computation of CMs)?
3. What are the linguistic constraints of the task (Whether semantic knowledge about words can be used)?
4. If a combination of CMs is to be used, which CMs should be combined?

Even though so many different kinds of CMs have been reported in the literature, to quote Jiang [17], *overall performance of CMs (even the best ones) remains fairly poor, which largely limits their applications.*

Chapter 3. **Keyword Spotting (KWS) Experiments**

3.1 Database for KWS experiments

An acoustic KWS system using phone models requires following 4 types of datasets.

1. Phonetic training set – Used for training the phone HMMs.
2. Phonetic test set – Used for tuning the phone recognition system parameters.
3. Keyword spotting development set – Used for tuning the KWS system parameters (e.g. word insertion penalty).
4. Keyword spotting test set – Used for testing the KWS system.

The acoustic conditions (background noise level, channel characteristics) of all these sets should match with each other.

Currently, we are using the TIFR Hindi database (TIFR-Hin) [38] for all the experiments. Developed on the lines of the TIMIT database, it comprises phonetically rich Hindi sentences uttered by 100 native speakers of Hindi. Each speaker utters 8 unique and 2 common sentences. The speech is recorded in quiet at 16 kHz, 16 bit PCM mono format. Total duration of the database is 75 minutes. The database includes phonetic (time-aligned) and word level (not time-aligned) transcriptions. The complete phone set of the database comprises of 94 phones / sub-phones. These were divided in 37 classes based on considerations such as acoustic similarity, amount of training duration available for each phone. The 37 class phone set includes 10 vowels, 4 nasals, 4 semi-vowels, 4 fricatives, 10 stop/affricate bursts, and 1 general model each for unvoiced stop/affricate closures, voicebar of the voiced stops, glottal pause, flaps and long silence. For convenience, we refer to both phone and sub-phone models as phone models.

Our keyword set contains 4 keywords (*dhobin, bartan, katghar* and *dakshin*) each occurring 100 times in the 2x100 common sentences of TIFR-Hin. The ground truth locations of the keywords were obtained by searching the phone sequences (corresponding to the pronunciation variants of the keywords) in the time-aligned phonetic transcripts. One drawback from the evaluation point of view is that all the keywords always occur in the same left and right word context. But since different speakers speak the keywords differently, a large number of pronunciation variants were observed for each keyword. So the dataset seems good on the count of speaker variability.

Due to lack of sufficient amount of separate test dataset we had to overlap the 4 datasets as described as follows -

1. **Phonetic training set** – The 37 phone HMMs were trained on 8 non-common sentences from 76 speakers (total duration of around 45 minutes) of TIFR-Hin.
2. **Phonetic test set** – Phone recognition parameters were tuned on the test set comprising 8x24 non-common sentences of 24 speakers (separate from the training set) of TIFR-Hin. The phone recognition accuracy was 64%.
3. **Keyword spotting development set** – 500 sentences from TIFR-Hin were used as the development set. These included 100 keyword carrying sentences (with 4x50 keyword occurrences) and 400 keyword-free sentences. The parameter tuning procedure is described in Section 3.4.2.
4. **Keyword spotting test set** – All the 1000 sentences of TIFR-Hin were taken as the test set. These contain 200 keyword carrying sentences (100 occurrences of each of the 4 keywords).

3.2 Keyword spotting system description

A Multistage KWS system has been implemented (see Figure 8). Its various components are:

1. Baseline system (An acoustic KWS system after Rose and Paul [8]) – The baseline system is tuned to operate at a very high recall and as low number of false alarms as possible.
2. Refinement stages – The refinement stages aim at scrutinizing baseline hypotheses in order to reduce the number of false alarms using various confidence measures without significantly decreasing the hit rate. Two separate refinement stages are implemented (one based on Isolated Word Recognition and one using re-recognition with KW-Filler network). Also preliminary work on a burst detection based refinement stage is described.

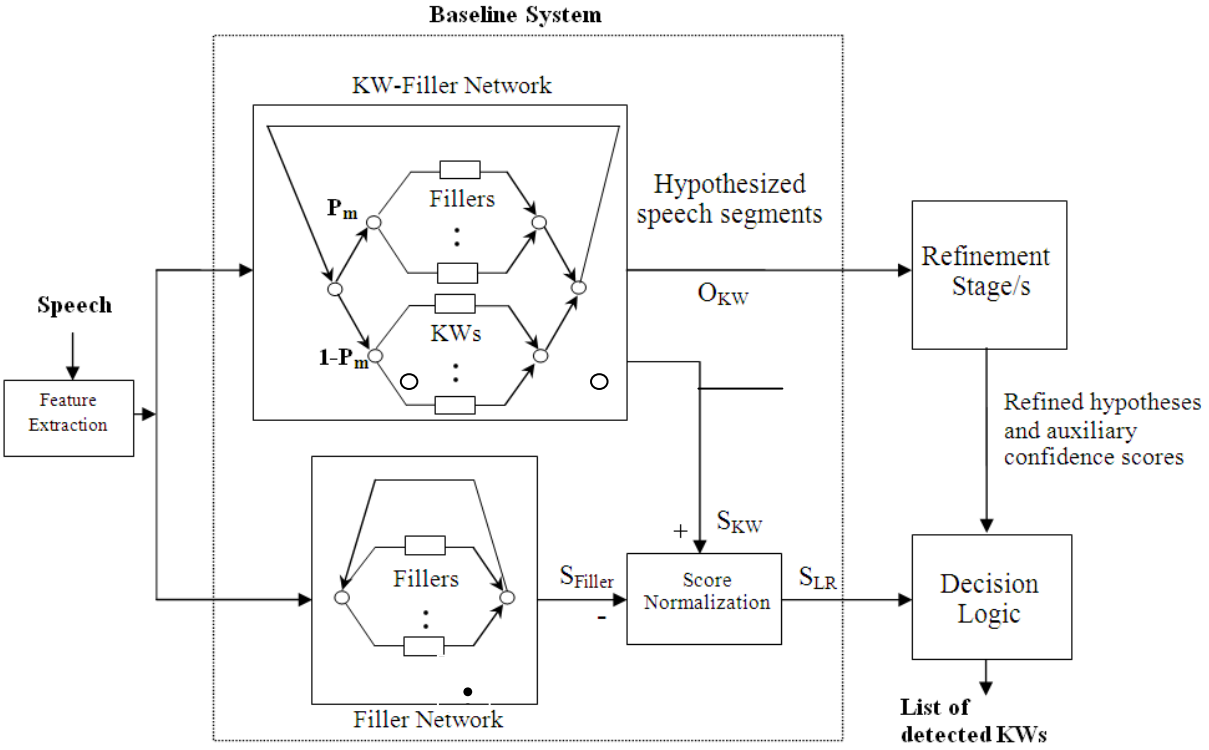

Figure 8: Overall KWS system configuration - the baseline and refinement stages

3.3 Feature extraction and phone HMM training

Feature extraction is the first step before both HMM training and recognition. The feature extraction module extracts 13 Mel-frequency Cepstral Coefficients (MFCCs), from each 25 msec of Hamming windowed pre-emphasized (alpha = 0.97) input audio at the frame rate of 100 fps. Cepstral mean normalization is applied where normalization is done over the whole utterance. Total 39 i.e. 13 MFCCs (including energy coeff.) + 13 delta + 13 delta-delta coefficients are used as an observation vector for a frame.

37 context independent phone HMMs (3 state, left-to-right topology, no skip transition) were trained on the phonetic training set using the SphinxTrain tool from the CMUSphinx toolkit [3]. The GMM parameters such as number of Gaussians per mixture and phone recognition parameters such as word insertion penalty (WIP) were optimized by experimenting on the phonetic test set. Note that during recognition, each phone is considered as a monophone 'word'; hence the WIP parameter is applicable for phone recognition also. More information about WIP and other tuning parameters is given in Section 2.4.2.

3.4 Baseline System

The baseline system is composed of two blocks, a keyword-filler network (KW-Filler n/w) block and a filler network block as shown in Figure 8. Both the blocks are standard HMM based continuous speech recognition systems in which frame synchronous Viterbi search is

run on the respective recognition networks. These blocks are implemented using the Sphinx-3 decoder [3].

3.4.1 Configuration of KW-Filler and Filler Networks

In the KW-Filler block, the non-keyword speech is modelled by a sequence of monophones and the keyword HMMs are concatenation of suitable monophone models (as per the pronunciation dictionary). For an input speech utterance the KW-Filler block outputs a stream of keyword and filler hypotheses along with the log-likelihood score and timing information for each hypothesis. The log-likelihood score of each keyword hypothesis is duration normalized and a confidence score S_{KW} is computed as in Equation 11.

$$S_{KW} = \frac{\log P(O_{KW}|KW)}{NF_{KW}} \tag{11}$$

Here, O_{KW} is the sequence of observation vectors corresponding to the audio segment within which the keyword KW was hypothesized and NF_{KW} is the number of frames in this segment. Even after duration normalization, the keyword likelihood scores exhibit a large variation. A further score normalization using the score of fillers overlapping with the decoded keyword was proposed in [8]. These fillers are obtained by passing the complete speech utterance through a filler network. This normalization can be viewed as a likelihood ratio test between keyword probability and filler probability for a hypothesized segment. The normalized score S_{LR} is given as in Equation 12.

$$S_{LR} = S_{KW} - S_F \tag{12}$$

In our implementation, S_F was obtained from a filler network (each filler being a monophone). It is computed as per Equation 13.

$$S_F = \frac{\sum_{i=1}^{m} frac \cdot S_{(F,i)}}{NF_{KW}} \tag{13}$$

Here, $S_{(F,i)}$ is the log-likelihood score of the i^{th} filler overlapping with the keyword hypothesis and $frac$ is the fraction of the filler which overlaps with the keyword (see Figure 9).

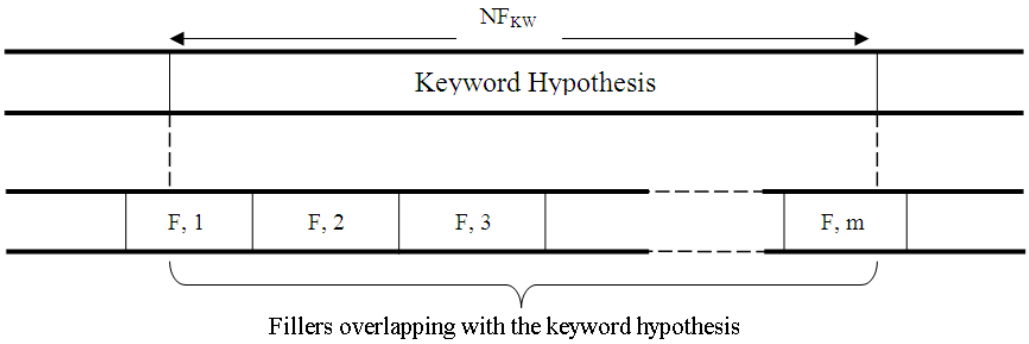

Figure 9: Outputs of KW-Filler network and parallel filler network block

Shown in Figures 10 and 11 are the score distributions for true hits and false alarms with S_{KW} and S_{LR} as confidence scores. These plots show that S_{LR} is a better discriminator of true hits and false alarms than S_{KW}.

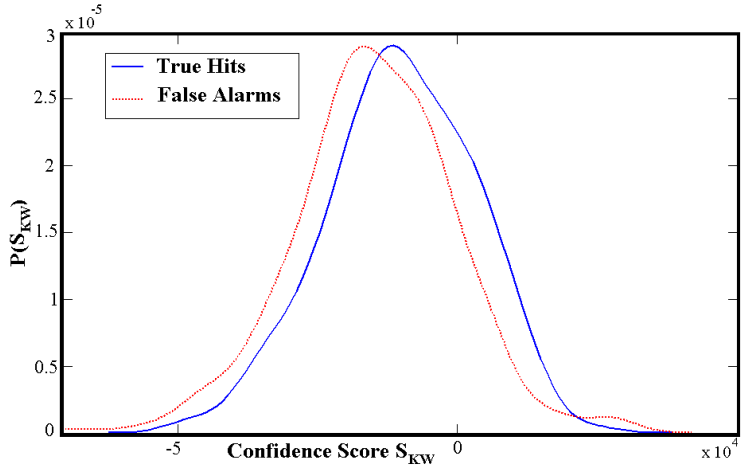

Figure 10: Distribution of confidence score S_{KW} over true hits and false alarms obtained on the KWS development set. The pdf has been smoothened in Matlab

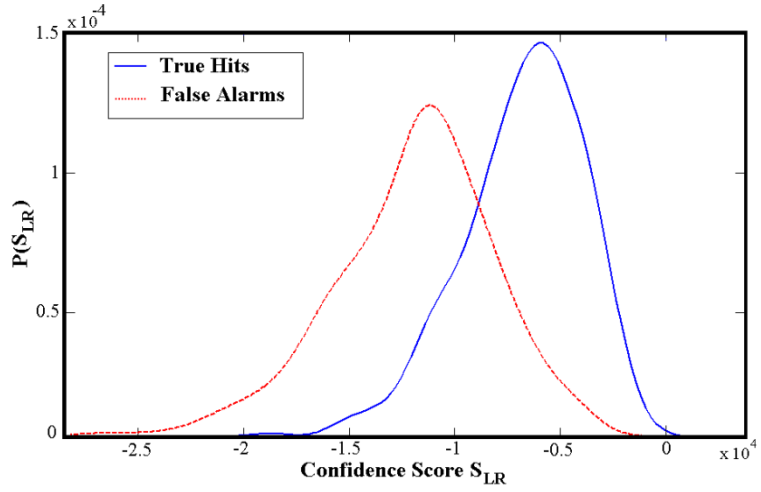

Figure 11: Distribution of confidence score S_{LR} over true hits and false alarms obtained on the KWS development set. The pdf has been smoothened in Matlab

3.4.2 Parameter tuning of KW-Filler network

We assume no knowledge about either the monophone sequence/distribution in the non-keyword speech or the number of occurrences of each keyword in the database. Let N_m and N_{KW} be the number of monophones and keywords in the KW-Filler network. A priori probability of the monophone branch is P_m (see KW-Filler n/w in Figure 8), a priori probability of each monophone is then P_m/N_m and that of each keyword is $(1-P_m)/N_{KW}$.

The decoder hypothesizes a keyword-filler sequence $\widehat{W} = (W_1, W_2, \ldots, W_N)$ for a given utterance as per the maximum likelihood criterion [39] in Equation 14.

$$\widehat{W} = \underset{W}{\operatorname{argmax}}\left(\sum_{i=1}^{N} \log P\left(O_{W_i}|W_i\right) + LW \sum_{i=1}^{N} \log P(W_i) + N \log(WIP)\right) \tag{14}$$

Here, O_{W_i} corresponds to the sequence of observation vectors for the segment in which word W_i is hypothesized, $P(W_i)$ is the a priori probability of the word W_i, LW is the language weight and WIP is the word insertion penalty. The dynamic ranges of $P(W_i)$ (which comes from a discrete distribution) and $P\left(O_{W_i}|W_i\right)$ (which comes from continuous mixture Gaussian distribution) are vastly different. The LW parameter (> 1) acts as a balancing factor between the dynamic ranges of language model and acoustic model probabilities so that an (approximately) equal importance is given to both. But in doing so it reduces the cost associated with each word that is added to the hypothesis (due to its multiplication with $P(W_i)$, which is a negative number). If the cost to add a word decreases, the decoder will hypothesize a greater number of shorter words in the hypothesis. To compensate for this side-effect, a word insertion penalty (WIP) is introduced. Thus, for each word added to the hypothesis, log(WIP) cost is added to the total hypothesis score. In speech recognition systems, the optimal values of LW and WIP parameters are empirically decided on the development set via a time consuming process [40].

Unlike LVCSR based keyword spotting systems, where $P(W_i)$ is determined by a statistical language model trained on a large amount of text data, in acoustic KWS systems the a priori probability of keywords has to be derived empirically [41]. We tuned three parameters P_m, LW and WIP of the baseline system to achieve very high hit rate and false alarm rate as low as possible on the KWS development set. First of all, LW and WIP were kept at their default values (9.5 and 0.7) in Sphinx-3 decoder. Since likelihood of a keyword occurring at any point in speech is smaller than that of the non-keywords, P_m was varied from 0.8 to 0.95 in the steps of 0.05 to find out its value which gives minimum number of false alarms. At P_m =

0.95, lowest number of false alarms was observed. Then, optimal values of WIP and LW were searched for. First, LW was varied from 1-10 and then WIP was varied from 1-12 keeping LW at its best value, based on hit rate and number of false alarms). We are aware that sequential optimization is sub-optimal but in future experiments we will try to perform a joint optimization of these parameters. The criterion of optimization was to achieve very high hit rate (99%) on the KWS development set and number of false alarm as low as possible.

Following general observations were made during the tuning process.

1. False alarm rate decreases with increase in P_m and WIP. Greater P_m means that fillers will be favoured to a keyword; hence the number of false alarms can be reduced by increasing P_m. Very large WIP results in large number of short words (i.e. fillers) in the hypothesis. A small WIP causes a large number of longer words (i.e. keywords) in the hypothesis. This can be explained as follows (assuming $WIP > 1$). During the Viterbi search, a word lattice is formed which represents various hypotheses at different points in the speech. Finally backtracking is done to retrieve the highest scoring path through this lattice. When WIP is very large, the lattice is mainly populated with fillers. This is because, in order to add a word to the lattice, its score has to cross a certain threshold. Larger WIP adds up to the scores of the words and helps shorter words to cross the threshold earlier than competing longer words during the search process.
2. False alarm rate increases with increase in LW (LW varied from 1-10). But hit rate either increases or decreases with LW, depending on the value of WIP (see Table 5). At low WIP values (e.g. 1), hit rate increases with LW and saturates at 100%. While at higher WIP values (e.g. 10), the hit rate reaches a maximum value (< 100%) and then starts decreasing with further increase in LW. In both the cases (low and high WIP), at LW is increased to very high values (> 20) both hit rate and false alarm rates start decreasing from their maximum values and reach zero.

It is important to achieve very high hit rate at this stage because if some keywords go undetected at this stage then they cannot be recovered by the later stages (which are aimed at reducing the number of false alarms).

Table 5: Effect of LW on hit rate and false alarm rate for different values of WIP. Numbers are obtained from experiments on the KWS development set

LW	WIP = 1		WIP = 12	
	Hit rate (%)	False alarm rate (FA/KW/Hour)	Hit rate (%)	False alarm rate (FA/KW/Hour)
1	81.5	3.16	41	0.00
2	59.5	50.12	60	1.19
3	99	113.45	74	1.58
4	100	270.16	84	4.74
5	100	452.23	91	12.25
6	100	615.55	96	23.72
7	100	779.55	92	38.34
8	100	865.20	85	49.41
9	100	985.11	79	60.88
110	100	1150.10	69	71.55

3.5 Isolated Word Recognition (IWR) block

Tajedor et al. [28] have proposed a refinement stage based on isolated word recognition. The motivation behind this approach is as follows. Since the Viterbi search computes an optimal path for the entire utterance, all the keyword hypotheses are influenced by all other hypotheses over the utterance. In order to obtain a confidence measure exclusively based on the hypothesized speech segment (O_{KW}), in the IWR block, the acoustic log-likelihood scores $\log P(O_{KW}|KW_i)$ are computed for all the keywords KW_i in the keyword set. These scores are then sorted in a decreasing order. Based on this list, the decision logic block decides whether to accept or reject the baseline hypothesis. Two confidence measures (CMs) that are implemented are as follows -

1. **Exact Match** - Let KW_k be the k[th] keyword in the sorted score list. If the top-scoring keyword (KW_1) in the list is the same as the baseline hypothesis, then the hypothesis is accepted.
2. **Difference of log-likelihoods** - If the 'Exact Match' condition is satisfied, then two scores ($diff_{12}$ and $diff_{13}$) are computed as in Equation 15 and 16.

$$diff_{12} = \log P(O_{KW}|KW_1) - \log P(O_{KW}|KW_2) \quad (15)$$

$$diff_{13} = \log P(O_{KW}|KW_1) - \log P(O_{KW}|KW_3) \quad (16)$$

If $diff_{12} \geq T_1$ and $diff_{13} \geq T_2$ then the baseline hypothesis is accepted. The thresholds T_1 and T_2 were decided based on the experiments on the development set. The difference of log-likelihood CM is motivated by the following argument. If the baseline hypothesis is a true hit, then both $diff_{12}$ and $diff_{13}$ would be of greater magnitude than the case when it is a false alarm. For a false alarm, the log-likelihood scores would not exhibit much variation over the keywords as it may not be acoustically similar to either of them.

3.6 Re-recognition with KW-Filler network

The isolated word recognition system is constrained to output one of the keywords as top scorer. This was observed to cause false alarms when the hypothesized speech segment was phonetically very similar to one of the keywords. For example, when the actual spoken word was *gharghar*, the isolated word recognizer gave highest likelihood for the keyword *katghar*. The $diff_{12}$ and $diff_{13}$ confidence measures do not seem well equipped to handle such false alarms as the keyword likelihood is high due to phonetic similarity. Hence we replaced the IWR stage with a "KW_{HYP}-Filler network" block. In this network, the baseline keyword hypothesis (KW_{HYP}) is kept in parallel with all the monophones. The a priori probability of the keyword is kept the same as in the baseline $(1 - P_m)/N_{KW}$ so as not to cause more alarms than the baseline system.

It might seem that using the same KW-Filler network concept in both 1st and 2nd stage might not bring any advantage to the system. But the Viterbi search process in the baseline is not optimal for keyword recognition as it tries to maximize the path likelihood over the entire utterance while coming up with keyword/filler hypotheses. This drawback could possibly be overcome by re-recognition of hypothesized segments (either by IWR or modified KW-Filler network). In a sense the re-recognition approach is an optimized way of sliding-window search in audio. Here one does not have to slide the window over the entire utterance, but only the segments hypothesized by a baseline system.

3.7 Refinement stage based on burst detection

So far, we have discussed the methods to reduce the number of false alarms has by using the likelihood scores given by the Viterbi search based on phone HMMs trained on MFCC features. To investigate the possibility of using acoustic features of individual phones as a confidence measure, we decided to first look for the presence of burst in the keyword hypothesis. There are other acoustic features corresponding to each phone in Hindi, but presently we are using bursts.

For each keyword in our keyword set, certain strong stop bursts were observed. For example – /k/ and /t/ in *Katghar* (**/k/** /a/ **/t/** /vb/ /g/ /a/ /r/)

Keyword *katghar* had the most number of false alarms with the baseline system, and was chosen for testing. The decoder provides the boundaries of each phone in the hypothesis. In this method, all the hypothesized bursts are extracted with 30ms extra duration on each side of the boundary. This boundary judgment was based on the analysis of burst boundaries obtained during phone recognition. To find out the presence of burst, rate-of-rise (ROR) of energy in the frequency band 3500 Hz to 5000 Hz is computed (similar to [42]). Energy in this band is used as bursts have high energy in this band. Energy in this band is computed every 1ms, using a Hamming window of length 6ms. ROR at i^{th} frame is computed as

$$ROR(i) = E(i) - E(i-k) \tag{17}$$

Here, the timestep of 10 ms is used (i.e. i-k = 10). Value of ROR is expected to be large at the burst onset. In all the hypothesized *katghar* segments maximum value of ROR was computed for the two bursts (/k/, /t/) within respective phone boundaries. All the true hits and false alarms were sorted in decreasing order of maximum RORs, separately for the two bursts. This was done to decide a threshold value on ROR to separate true hits from false alarms. Following observations (Table 6) were made after counting number of false alarms for whom maximum ROR is greater than maximum ROR of the 10^{th} false rejection.

Table 6: Observations on the /katghar/ hypotheses

Initial number of false alarms at the output of baseline system before 10^{th} false rejection	**Number of false alarms before 10^{th} false rejection (after application of burst detection)**	
	/k/	/t/
160	70	65

3.8 Experiments and results

The results reported here are for the KWS test data which is the entire TIFR Hindi database. A keyword hypothesis is considered as a true hit if it overlaps more than 70% duration of the same keyword in the ground truth. The evaluation is performed in two ways. We report the Figure of Merit (FOM) performance metric. For FOM computation, the threshold on S_{LR} score was varied to compute hit rate at various false alarm rates. FOM results are given in Table 7.

Table 7: FOM results on KWS test set

System Index	System Configuration		FOM	Maximum hit rate	FA/KW/Hour at maximum hit rate
	First Stage	Second Stage			
1	KW-Filler n/w		51.27	99	66
2	KW-Filler n/w	IWR (Exact Match CM)	53.6	98.2	56
3	KW-Filler n/w	IWR (Exact Match and Difference of log-likelihoods)	56.7	91.5	36
4	KW-Filler n/w	KW_{HYP}-Filler n/w	58.6	95.25	34

The other evaluation involved counting the actual number of true hits for the whole test set. Here, for each system S_{LR} scores of all true hits and false alarms are sorted in a descending order. Then S_{LR} score is thresholded at each false alarm. The total number of true hits appearing in the list before 10^{th}, 20^{th}, … , 50^{th} false alarm are reported. Unlike FOM, here cumulative numbers of false alarms of all the keywords are considered. This table is useful for knowing where to set the operating point of the system.

Table 8: Number of hits before n^{th} false alarm [Total number of keyword occurrences in ground truth = 400]

System Index	Cumulative number of false alarms (all keywords)				
	10	20	30	40	50
1	112	192	234	263	271
2	141	195	241	261	288
3	136	225	264	281	299
4	160	213	268	292	306

From Table 7 and 8 it can be seen that the system performance has improved with the introduction of each refinement stage. The FOM performance of the KW-Filler re-recognition system is slightly better than the 'difference of log likelihoods' CM and also the maximum hit rate achieved is higher. This could be attributed to the flexibility in KW-Filler network to output a sequence of monophones instead of other keywords as in IWR block.

Chapter 4. Confidence Measure Experiments: Normalisation of Acoustic Score

4.1 Introduction

Recall the filler/phone recognition based keyword spotting post processing scheme (Equation 13). The same scheme is justified in a different way below. As previously discussed in Section 2.4.2, posterior probability of a word can be computed using a parallel phone decoder using Equation 18.

$$P\left(w_j|O_{w_j}\right) \approx \frac{P\left(O_{w_j}\middle|w_j\right)}{\max_F P(O_{w_j}|F)} \tag{18}$$

Equation 18 can be rewritten (with duration normalisation) as below to give formula for frame level phone decoder normalization based confidence measure C_{NLS}^f

$$C_{NLS}^f(w) = \sum_{k=1}^{N_w} \frac{aScore\left(O_k\middle|phn_{w,i}^k\right) - aScore\left(O_k\middle|phn_p^k\right)}{N_w} \tag{19}$$

Here, $phn_{w,i}^k$ corresponds to the HMM state of the i[th] phone belonging to the decoded word, in the k[th] frame of the word. Similarly phn_p^k is the HMM state of the parallelly decoded phone in the same k[th] frame. The acoustic score difference is normalized with $N_{w,}$ i.e. number of frames of the hypothesized word W. The term $aScore(O_k|phn)$ is acoustic score of corresponding to feature frame O_k (assumed 1 dimensional for simplicity)**,** computed on the probability distribution function of one of the states corresponding to the phone phn. It is given by Equation 20 as

$$aScore(O_k|phn) = \log(P_{trans}) + \sum_{j=1}^{M} -\frac{1}{2}\left(\log\left(2\pi\sigma_j^2\right) + \left(\frac{x-\mu_j}{\sigma_j}\right)^2 + \log(wt_j)\right) \tag{20}$$

Here, P_{trans} is the state transition probability corresponding to transition of states between current frame and next frame; $\mu_j, \sigma_j \; and \; wt_j$ are the mean, standard deviation and mixture weight respectively of the j[th] Gaussian mixture density corresponding to the HMM state decoded in current frame. Total M Gaussian densities are present in the GMM.

Due to similarity between the decoded phones in the phone decoder hypothesis (phn_p^k) and phones belonging to the decoded word ($phn_{w,i}^k$), the magnitude of the normalized word score is comparatively lower for a correct hypothesis than for an incorrect hypothesis. This is our **baseline** word score normalization.

Even though duration normalization is already applied, Cox and Rose [19] observed that the performance of the score normalization based confidence measure further improves when hypothesized words are grouped according to number of phones in them. Here, we will be evaluating on the confidence measure for a group of words with 5 phones, as they are most frequent in our database.

4.2 Database and evaluation criterion

4.2.1 Various subsets of Agmark Marathi database

This telephone speech database was collected and transcribed as a part of the DIT project. For more details about data collection see [43]. Speech data in the form of short phrases (mandi/commodity names) and Marathi sentences has been collected from 1500 speakers. Speakers are from all the 34 districts of Maharashtra. The database is recorded under realistic conditions; many utterances contain noise and/or background speech. As the database was still being collected and transcribed as the experiments in this report were carried out, following 2 subsets of the database were used for overall experimentation. The short phrases were originally recorded for duration of 3 – 5 seconds. Using a speech – silence segmentor [44] (implemented by TIFR team) the silence segments of more than 300 ms are chopped off before using this data for training/testing. The subsets of the Agmark database are listed in Table 9.

Table 9: Agmark Marathi databases

Name of database	Number of speakers	Duration	Nature of speech	Male:Female ratio
marathiAgmark850	850	~9 Hrs.	Short phrases (1-3 words)	80:20
marathiAgmark1500	1500	~20 Hrs.	Short phrases (1-3 words)	83:17

For the experiments described in this chapter we used **marathiAgmark850** database. It contains 24,617 short Marathi utterances (3-6 sec). The vocabulary size is 1084 words.

The CMUSphinx toolkit [3] has been used for training and testing the system. Sphinx3.8 decoder was used. For feature extraction, Hamming window of size 25.6 msec and frame rate

of 100 frames/sec was used. 39 MFCCs (13 cepstral + 13 delta + 13 double-delta) were extracted from each frame. Total 64 CD phones and 6 CI fillers were been trained. The triphone HMMs (3 states per model, no skip states) were trained which shared 500 senones among them. Each senone was modeled with 16 Gaussian mixture densities with a diagonal covariance matrix.

The database **marathiAgmark850** was divided in 2 parts -

1**.** 567 speakers (850x2/3) data for training CD triphone models which are used to do phone-loop as well as word decoding (Referred to as **567spkrSubset).**

2. 283 speakers (850x1/3) data for testing confidence measure (Referred to as **283spkrSubset)**.

In another experimental configuration, the phone models for phone-loop decoder were trained on the complete 850 speaker data. More details about configuration of the experiments are in **Section 4.4.**

4.2.2 Evaluation criterion

Word level decoding is done on **283spkrSubset** (unseen data). Then, baseline word score normalization C_{NLS}^{f} is applied and an ROC curve (Probability of Correct Acceptance Vs Probability of Wrong Acceptance) is obtained. To obtain the ROC curve, score threshold is varied over its complete range and following quantities are computed for each value of the threshold –

$$P(\text{Correct Acceptance}) = \frac{\text{No. of correct word hypotheses with score} > threshold}{\text{Total No. of word hypotheses}} \quad (21)$$

$$P(\text{Wrong Acceptance}) = \frac{\text{No. of insertions/substitutions with score} > threshold}{\text{Total No. of word hypotheses}} \quad (22)$$

For comparison, similar curves are drawn for the other proposed variants of score normalisation.

4.3 Proposed modifications in the acoustic score normalization technique

4.3.1 Phone-level score normalization

The baseline word score normalization gives equal importance to each frame. But as it turns out, each phone has a different duration, so scores of longer duration phones (e.g. Vowels) dominate the word score. The baseline confidence measure is mainly affected by the

normalized frame scores in which such long duration phones are present. To reduce this bias, instead of considering each frame independent of another, per phone normalised scores were computed and added together to give normalized word score. Similar idea was used by Rivlin et al. in [45] in the context of computing phone posterior probability.

The expression for Phone level normalized score S_{NLS}^{p} is given by

$$S_{NLS}^{p}(phn_{w,i}) = \sum_{k=phn_{w,i}^{Start}}^{phn_{w,i}^{End}} \frac{aScore(O_k|phn_{w,i}^{k}) - aScore(O_k|phn_{p}^{k})}{phn_{w,i}^{End} - phn_{w,i}^{Start} + 1} \tag{23}$$

Here, $phn_{w,i}^{Start}$ and $phn_{w,i}^{End}$ correspond to the start and end frame respectively of the phone belonging to the decoded word. Normalized acoustic scores of all phones are clubbed together to get phone level score normalization C_{NLS}^{p} (Equation 24).

$$C_{NLS}^{p}(w) = \frac{1}{N_{phn,w}} \sum_{i=1}^{N_{phn,w}} S_{NLS}^{p}(phn_{w,i}) \tag{24}$$

Here, $N_{phn,w}$ is the total number of phones in the word *w*.

To further bring phone dependence into picture following modifications were done.

4.3.2 Phone accuracy based normalization

Phone recognition was done on the training data (**567spkrSubset**) using the acoustic models prepared from the same data. On this data, accuracy of each phone *phn* was computed as-

$$\mathrm{A}(phn) = \frac{\text{No. of frames correctly decoded frames of } phn}{\text{Total No. of frames of } phn} \tag{25}$$

We have less confidence (about phone identity) in those frames in which a low accuracy phone has been decoded. For each phone, the normalized acoustic scores S_{NLS}^{p} (Equation. 23) are scaled by a factor $[2 - \mathrm{A}(phn)]$ so that, the normalized acoustic score is decreased by a larger amount for low accuracy phones, than that for high accuracy phones. The formula for phone accuracy based normalization is given by Equation 26.

$$\begin{aligned} &\text{if } S_{NLS}^{p}(phn_{w,i}) < 0 \\ &\quad C_{A}^{p}(w) = \frac{1}{N_{phn,w}} \sum_{i=1}^{N_{phn,w}} \mathrm{C}_1 \cdot \left(2 - \mathrm{A}(phn_{w,i})\right) \cdot S_{NLS}^{p}(phn_{w,i}) \\ &\text{else} \\ &\quad C_{A}^{p}(w) = C_{NLS}^{p}(w) \end{aligned} \tag{26}$$

Constance C_1 was used to adjust the dynamic range of scaling. Its value was kept at 6 (obtained after some trial and error). The reason we are applying scaling only if $S_{NLS}^{p}(phn_{w,i}) < 0$ is that a non-negative S_{NLS}^{p} denotes that the phone has been decoded with high confidence and further scaling is not required.

4.3.3 Phone F-Score based normalization

F-Score of a phone is a distance measure between two acoustic score distributions, one from the correctly recognized frames of that phone and another from the wrongly recognized frames. For computing F-Score, the reference phone level transcript for the training data (**567spkrSubset**) is force-aligned to the audio. Similarly, the phone decoder output transcript is also force-aligned. For each phone, frame-level normalized acoustic score C_{NLS}^{f} is computed (difference between reference and hypothesized frame scores). F-Score gives an indication of the discriminative power of a phone (criterion being the acoustic score distributions). Its formula is -

$$\text{Fscore}(phn) = \frac{\mu_{\text{correct}} - \mu_{\text{wrong}}}{\sigma_{\text{correct}} + \sigma_{\text{wrong}}} \tag{27}$$

Here, μ_{correct} and σ_{correct} are the mean and standard deviation of the acoustic score distribution obtained from the correctly recognized frames. Similarly, μ_{wrong} and σ_{wrong} are obtained from the wrongly recognized frames.

Figure 12 shows acoustic score distributions coming from correctly and incorrectly frames of the phone /b/. Note the peakedness of correct frame score distribution Vs the spread of wrong frame score distribution.

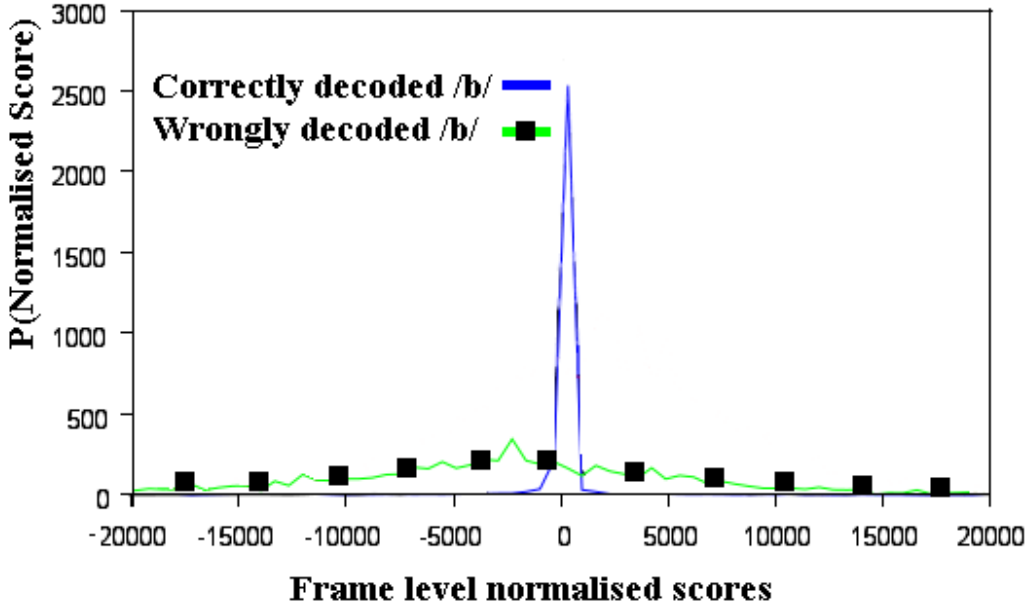

Figure 12: Acoustic score distributions of phones /e/ and /b/. These distributions are used to compute the F-Score of the corresponding phones.

Interestingly, some phones have negative Fscore. These mainly are the phones with small frequency in the data. Since Fscore are negative for some phones, the scaling factor applied for each phone has to be changed accordingly so that finally the phone Fscore based normalisation $C_F^p(w)$ is given by

if $S_{NLS}^p(phn_{w,i}) < 0$

$$C_F^p(w) = \frac{1}{N_{phn,w}} \sum_{i=1}^{N_{phn,w}} \mathrm{C}_1 \cdot (2 - \frac{\mathrm{Fscore}(phn_{w,i}) - minFscore}{\mathrm{maxFscore} - \mathrm{minFscore}}) \cdot S_{NLS}^p(phn_{w,i}) \quad (28)$$

else

$$C_F^p(w) = C_{NLS}^p(w)$$

The scaling factor ensures that the normalized acoustic scores of phones are decreased by a larger amount for phones with smaller F-Score, than those with larger F-Score. Here, too, scaling is applied only when phone-level normalized acoustic score is negative. Constance C_1 was used to adjust the dynamic range of scaling. Its value was kept at 6 (with some trial and error).

4.3.4 Phone confusion matrix based normalization

A phone confusion matrix was obtained on the training data (**567spkrSubset**). Phone confusion between a reference/ground truth phone phn_{ref} and a hypothesized phone phn_{hyp} is computed as in Equation 29.

$$\mathrm{PC}(phn_{ref}, phn_{hyp}) = \frac{\text{No. of frames in which } phn_{ref} \text{ was substituted as } phn_{ref}}{\text{Total No. of frames of } phn_{ref}} \quad (29)$$

If a pair of phones is highly confusable (e.g. /i/ and /ii/) then the normalized acoustic score should be decreased by a smaller amount than the case in which the phones that are highly distinct (e.g. /i/ and /th/). A symmetric distance measure between two phones can be defined in terms of entries in the phone confusion matrix as in Equation 30.

$$\mathrm{D}(phn_{ref}, phn_{hyp}) = (1 - PC(phn_{ref}, phn_{hyp})) \cdot (1 - PC(phn_{hyp}, phn_{ref})) \quad (30)$$

Note that this distance measure can be applied only over those contiguous frames over which a single phone is present in both word decoder and phone decoder hypothesis. Let there be $N_{seg,w}$ such segments in a hypothesized word w.

Thus, we obtain another scaling scheme based on phone confusion matrix as below

if $S_{NLS}^{p}(segment_{w,j}) < 0$

$$C_{PC}^{p}(w) = \frac{1}{N_{seg,w}} \sum_{j=1}^{N_{seg,w}} C_1 \cdot (2 - D(phn_{w,j}, phn_{p,j})) \cdot S_{NLS}^{p}(segment_{w,j}) \quad (31)$$

else

$$C_{PC}^{p}(w) = C_{NLS}^{p}(w)$$

Here, $phn_{w,j}$ is the phone belonging to the word in j[th] segment, and $phn_{p,j}$ is the corresponding phone in the phone decoder output. Here, too, scaling is applied only when segment-level normalized acoustic score is negative. Also C_1 is kept at 6 here as well.

4.4 Experiments, results and discussion

In order to test the 5 confidence measures (C_{NLS}^{f}, C_{NLS}^{p}, C_{A}^{p}, C_{F}^{p}, C_{PC}^{p}) we run experiments under 2 different configurations as described in Table 10. Basic motivation behind these two configurations is that we have to see the effect of phone recognition performance on the implemented CMs. In configuration1, phone recognition performance is better because the test data is a subset of training data. Also, since we are using different models for phone and word recognition, the acoustic scores are not consistent. In order to obtain consistent scores we do phone/word alignment with models trained on **567spkrSubset.**

Table 10: Details of Configuration 1 and Configuration 2

	Configuration 1	**Configuration 2**
Test data (word level decoding)	**283spkrSubset** (1/3 part) of 850 speaker database	-same as config1-
Phone models for word level decoding	500 senone, 16 Gaussian CD phone HMMs trained on 2/3 of the 850 speaker data (**567spkrSubset**)	-same as config1-
Phone models for phone level decoding	500 senone, 16 Gaussian CD phone HMMs trained on complete 850 speaker data	500 senone, 16 Gaussian CD phone HMMs trained on 2/3 of the 850 speaker data (**567spkrSubset**)
Language model for word level decoding	Backoff trigram LM trained on complete 850 speaker data	-same as config1-
Language model for phone level decoding	Bigram phone LM, derived from complete 850 speaker data was used for	Bigram phone LM, derived from **567spkrSubset** was

	phone recognition. Also during phone decoding, the **+babble+** filler was removed from filler dictionary as it was causing lot of substitutions.	used for phone recognition. +babble+ removed from filler dictionary.
Phone models for word level alignment	500 senone, 16 Gaussian CD phone HMMs trained on **567spkrSubset**	-same as config1-
Phone models for phone level alignment	500 senone, 16 Gaussian CD phone HMMs trained on **567spkrSubset**	-same as config1-
WIP for word decoding	Is set based on word decoding of **567spkrSubset** using models created from the same data. WIP for word decoding = 25	-same as config1-
WIP for phone decoding	Is set based on performance on **567spkrSubset**, when models were trained on the same **567spkrSubset**. Criterion was that approx. equal deletions and insertions should occur. WIP for phone decoding = 5000	-same as config1-

Since the phone models used for phone decoding are different under the two configurations, we obtain phone recognition results on **283spkrSubset** (Table 11)**.**

Table 11: Phone recognition on **283spkrSubset** speaker database under **Configuration1** and **Configuration2**

Phone recognition result	**Configuration1**	**Configuration2**
% Correct	77.8	54.0
% Substitution	11.2	31.0
% Deletions	11.0	15.0
% Insertions	4.2	9.2
% Phone Accuracy	73.6	44.8

ROC curves for 5 phone words: The hypothesized words were categorized according to number of phones in them. Words with 5 phones are most frequent in the database and are

considered here for performance evaluation. Plots are shows in Figures 13 and 14.

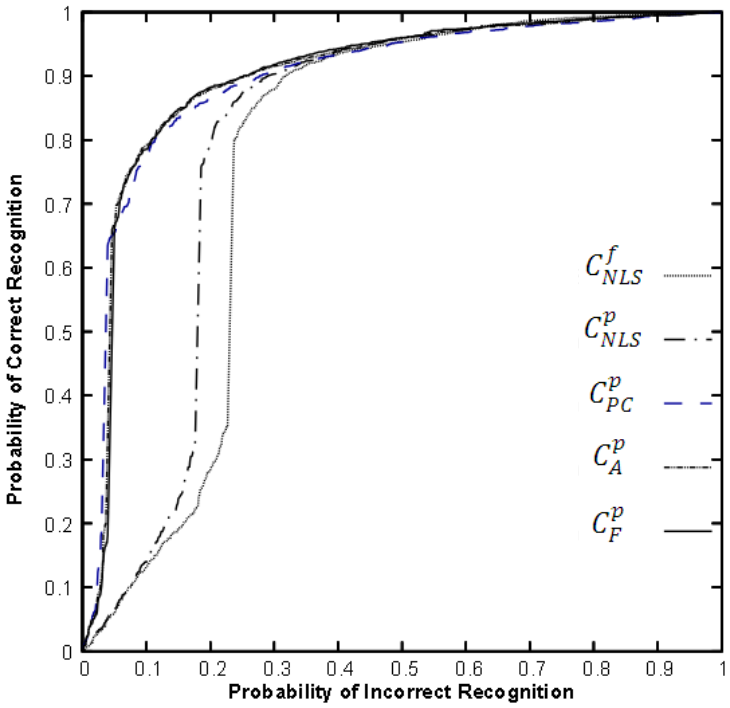

Figure 13: ROC plots (**283spkrSubset**) using configuration 1 (phone decoding using models trained on complete database), Total no. of correct words = 2768, Total no. incorrect words = 1040

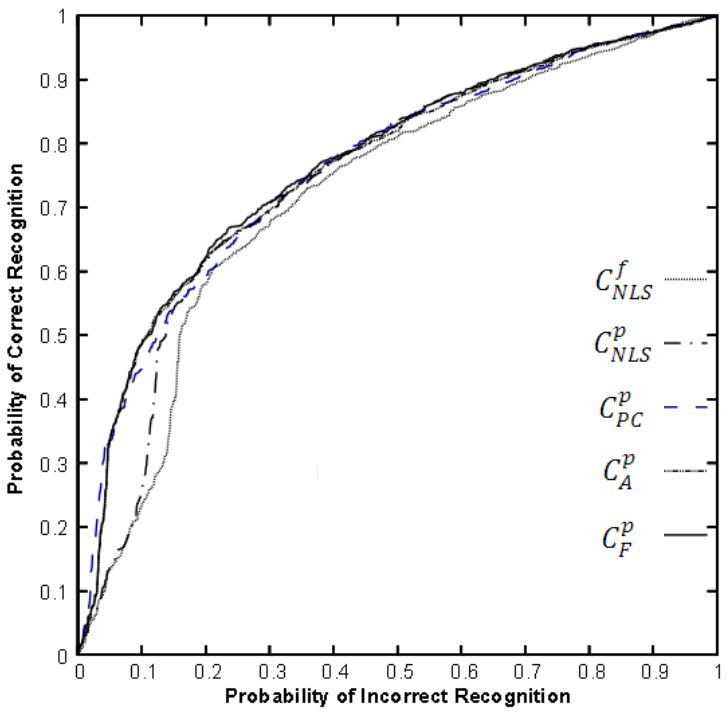

Figure 14: ROC plots (**283spkrSubset**) using configuration 2 (phone decoding using models trained on **576spkrSubset**), Total no. of correct words = 2768, Total no. incorrect words = 1040

Discussion: The proposed word score modifications showed more improvement when the phone recognition performance was high (77.8% correct, 73.6% accuracy). When phone recognition was done using models derived from a separate data, correctness and accuracy dropped (54% correct, 44.8% accuracy). In this case the improvement in performance was marginal (for probability of correct recognition > 0.6), as seen in the ROC curves. Note that this area (probability of correct recognition > 0.6) corresponds to the case when normalized word scores are very negative. I had tried the **allphone** mode in sphinx3 which is supposed to work with CD models for phone recognition, but it only gave 2% improvement in correctness.

Why the shapes of ROC curves differ for the two configurations?

Both the ROC curves can be divided in 3 parts. Consider the ROC curve corresponding to C^p_{NLS} (phone level normalisation) in both the figures for the below explanation.

1. **Part1:** The middle straight line like part, it corresponds to hypotheses words with normalized *near zero* scores. Roughly, It goes from 0.35 to 0.8 (Y-Axis value) in Figure 13 and 0.3 to 0.5 in Figure 14. Smaller part1 in Figure 14 means that number of correct hypotheses with near zero scores is *much less* under configuration 2, than under configuration

2. **Part2:** The rise in lower left corner – it corresponds to hypotheses words with *more than zero* normalized score. The lower-left-rise corresponds to 0 to 0.2 (X-axis) in Figure 13 and 0 to 0.1 (X-axis) in Figure 14. This means that the number of incorrect hypothesis with high positive normalized scores is *less* under configuration 2.

3. **Part3:** The slow rise in upper right corner – It corresponds to hypotheses words with *less than zero* normalized scores. For Figure 13, it lies between 0.85 – 1 (Y-axis) and 0.25 – 1 (X-axis). In Figure 14, it lies between 0.5 to 1 (Y-Axis) and 0.2 to 1 (X-Axis).

This means, the number of correct hypotheses words with lower normalized scores are much less under configuration 1 than under configuration 2. The reason is mismatch between phones from word decoder and phones from phone decoder (due to poor phone recognition under configuration 2.)

Figure 15 shows effect on score distributions. As the scaling is applied only when normalized score is negative, all the distributions get scattered to left (ideally correct score distributions should have drifted so much to the left.) This also explains why the ROC curve is better with the modifications in the left most part of ROC curve (i.e. high threshold on the score) and worse in the rightmost part (i.e. smaller threshold on the score).

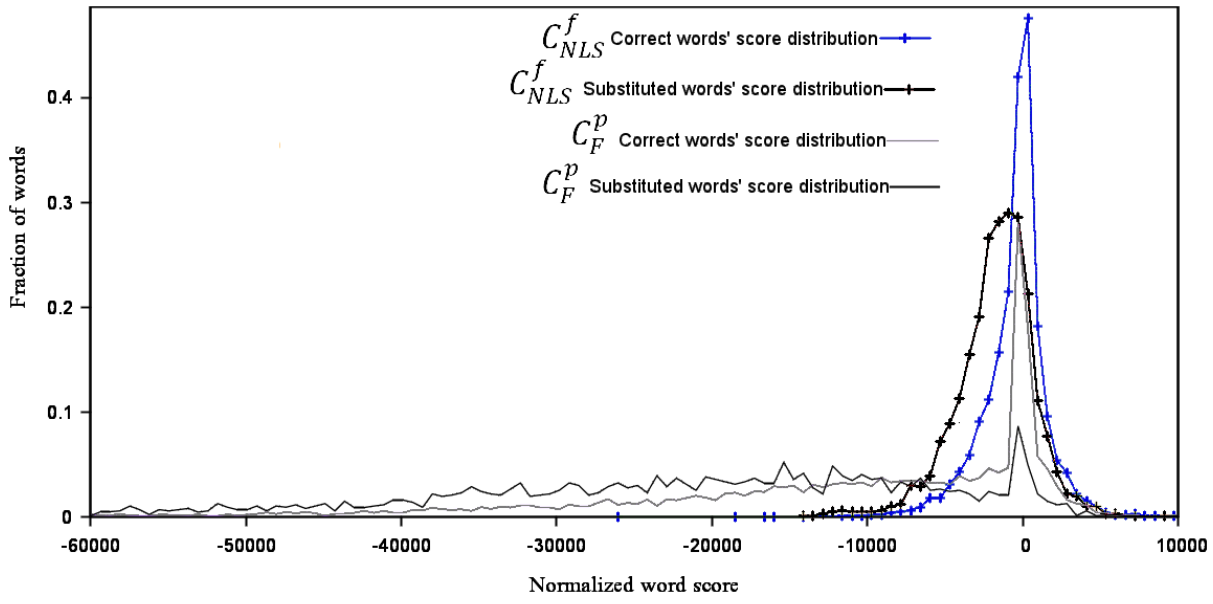

Figure 15: Fraction of words Vs Normalised word scores for C^f_{NLS} (baseline) and C^p_F (F-Score based normalization), under configuration 2.

Which among these is the best scheme for acoustic score normalization?

Confidence measure C_{PC}^{p} seems better in Part1 and Part2 of the ROC curve in Figures 13 and 14 but seems equal /slightly worse than C_{F}^{p} in upper-right region (Part3) of the ROC curve.

Improvement due to C_{PC}^{p} in Part1 and Part2 can be explained as follows: These correspond to the words for which phone recognition output closely matched with word decoder output. So these are examples of good/ideal articulations (w.r.t. training data). Phone recognizer is less likely to make mistakes for these words. Hence if it is an incorrect word hypothesis, with just 1 or 2 phones mismatched (e.g. reference word *Haapus* Vs hypothesis word *Kaapus*), then the phone decoder is likely to hypothesize a different but correct phone. Here, knowledge about phone confusability is more useful than only the phone accuracy of the decoded word's phones (C_{A}^{p}).

Ineffectiveness of scaling modifications in Part3 of the ROC curve is because in this region the phone recognizer is making more mistakes. So any scaling modifications which take help of phone decoder are not that useful.

Chapter 5. Confidence Measure Experiments: Using N-best list evidence

5.1 Confidence measures using N-best list

5.1.1 Various formulations for posterior probability from N-best list [26]

As seen in **Section 2.4.2** posterior probability for each word in top hypothesis is can be obtained with following formula-

$$C_{nbest}(w_i) = \frac{\sum_{\boldsymbol{W_{nbest}}: w_i \in W} P(W|O)}{\sum_{\boldsymbol{W_{nbest}}} P(W|O)} \quad (32)$$

Here, the relation $\boldsymbol{W_{nbest}}: w_i \in W$ signifies a subset of n-best hypotheses which contain the same word as top hypothesis in an overlapping position. Any non-zero overlap is allowed. This will be easier to understand from Figure 16.

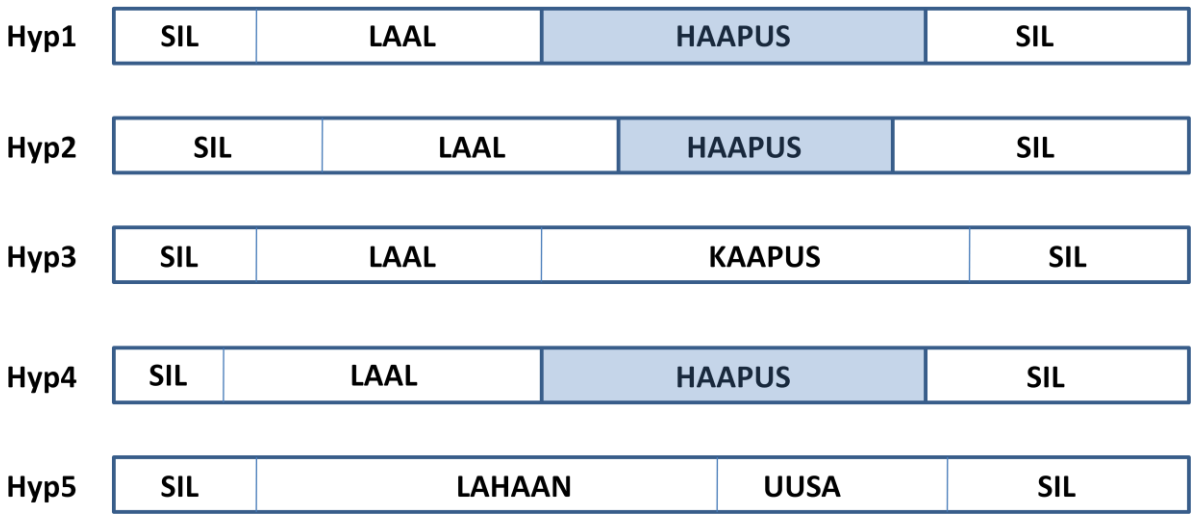

Figure 16: Example n-best list for an utterance. Here, for computing C_{nbest} of the word HAAPUS in the top hypothesis, hypotheses 1, 2 and 5 are considered in the numerator of Equation 32, while all the 5 hypotheses are considered while computing the denominator.

Equation 32 can be rewritten (after taking language weight into consideration)

$$C_{nbest}(w_i) = \frac{\sum_{\boldsymbol{W_{nbest}}: w_i \in W} P(W)^{lw} P(O|W)}{\sum_{\boldsymbol{W_{nbest}}} P(W)^{lw} P(O|W)} \quad (33)$$

Here, *lw* is the language weight. The probabilities $P(W)$ and $P(O|W)$ are very small and are not handled directly (to prevent loss of precision); instead we do all the computations in the log domain as in Equation 34. Base of the logarithm (*base*) is 1.0003 for Sphinx decoder.

$$C_{nbest}(w_i) = \frac{\sum_{\boldsymbol{W_{nbest}}: w_i \in W} \text{base}^{(\alpha \cdot lw \cdot \log P(W) + \beta \log P(O|W))}}{\sum_{\boldsymbol{W_{nbest}}} \text{base}^{(\alpha \cdot lw \cdot \log P(W) + \beta \log P(O|W))}} \quad (34)$$

Here, α *and* β are the individual scaling factors for language model score and acoustic score (in addition to language weight). We used the relation $\beta = 1 - \alpha$ in our experiments.

5.1.2 N-best word rate [14], [26]

Another confidence metric we use is N-best word rate $C_{nbest}^{fraction}$ is computed as follows. First, for a given word w_i belonging to the top level n-best hypothesis, we find all the words in all the n-best hypotheses (w_k^{nbest}) that have non-zero overlap with w_i. Then $C_{nbest}^{fraction}$ is given by

$$C_{nbest}^{fraction}(w_i) = \frac{\text{Number of } w_k^{nbest} \text{ that are same as } w_i}{\text{Total number of } \mathrm{w_k^{nbest}}} \quad (35)$$

Thus, we have more confidence about those words which repeat a lot in the n-best list.

5.2 Database description

Phone models used in these equations were 500 senone, 16 Gaussian CD phone HMMs trained on 2/3rd of the 850 speaker data (**567spkrSubset**). For word decoding, backoff trigram LM trained from the complete 850 speaker data was used. Test data was 283spkrSubset (remaining 1/3rd of) of 850 speaker database.

5.3 Experiments, results and discussion

ROC curves were obtained for 5 phone words in the test data (as done in Chapter 4), for three different values of α parameter (whereas $\beta = 1 - \alpha$). Language weight was 9.5

Case1: $\alpha = 0.5$

Here, both $P(W)$ and $P(O|W)$ contribute towards computation of posterior probability.

Case2: $\alpha = 1$

Here, only language probability $P(W)$ contributes towards computation of posterior probability.

Case3: $\alpha = 0$

Here, only acoustic probability $P(O|W)$ contributes towards computation of posterior probability.

Also, ROC curve for $C_{nbest}^{fraction}$ was also obtained.

Figure 17 shows the ROC curves obtained using n-best list based CMs, for comparison purpose ROC curve obtained with phone confusion matrix based score normalisation C_{PC}^{p} (Equation 31).

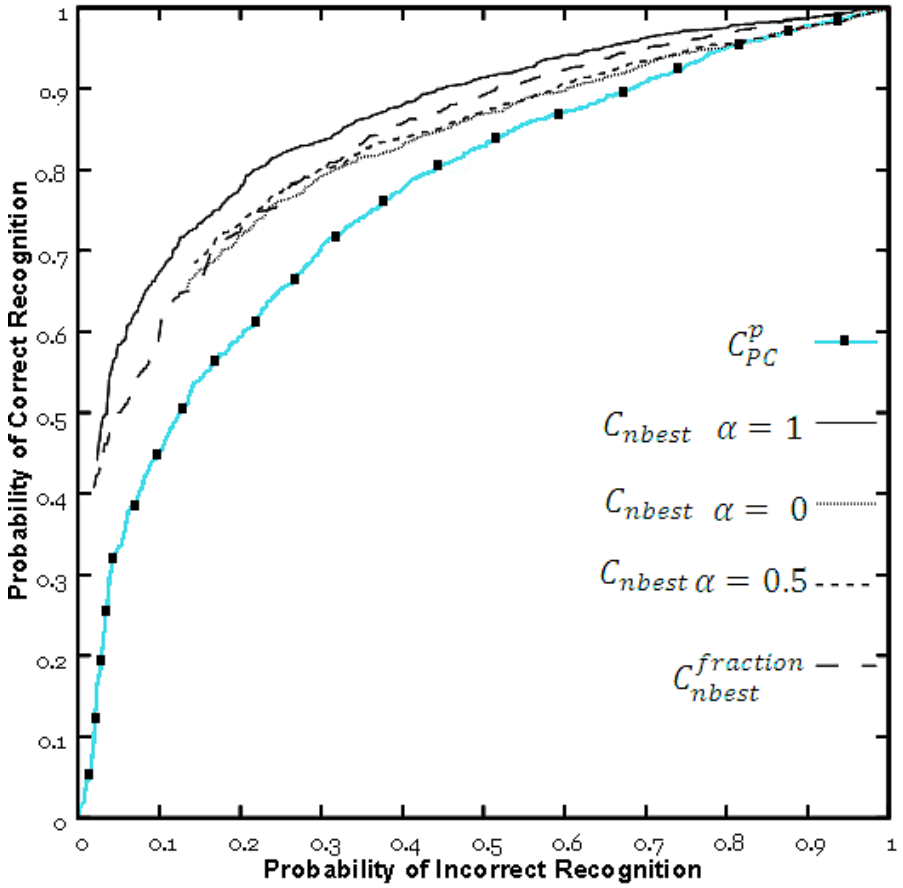

Figure 17: ROC curves for $C_{nbest}^{fraction}$, C_{PC}^{p}, and C_{nbest} (at $\alpha = 0, \alpha = 0.5$ and $\alpha = 1$) for testing on **283spkrSubset.** Total number of (5 phone) hypothesized words = 3808, total number of correct words = 2768; total number of incorrect words = 1040.

As observed in Figure 16, the confidence measures based on n-best list are all better than C_{PC}^{p} that was used previously. This is expected taking into consideration that success of C_{PC}^{p} depends on a good phone recognition performance. Also, theoretically, expression for

posterior probability using n-best list is more correct than the expression for posterior probability using parallel phone recognition as in the latter case we assume that each word sequence is equiprobable (Equation 6).

Most surprising thing is that the performance of C_{nbest} is best when $\alpha = 1$ i.e. only language probability $P(W)$ is used towards computation of posterior probability. This could be happening because of 2 reasons. Firstly, since we are using backoff trigram grammar for word recognition, there is a possibility that multiple words are recognized (even due to small amount of babble) causing lots of insertions. These insertions together with the correctly recognized words form sequences which were not seen in the training data. So for the inserted words, the language model probability is bound to be low. Performance of $C_{nbest}^{fraction}$ is only slightly better than that of C_{nbest} ($\alpha = 1\ or\ 0.5$). Given the simplicity of its expression, it is seems to be a good confidence measure.

Figure 18 shows the histograms obtained for the two CMs ($C_{nbest}^{fraction}\ and\ C_{nbest}$ ($\alpha = 1$)) for on **283spkrSubset** (unseen data). Here we consider all the words (irrespective of number of phones in them) hypothesized on the test data. In the upper inset, note the sudden rise in the fraction of correctly decoded words when CM value is increased beyond 99%. Two sudden peaks in the distribution for C_{nbest} at 0.35 and 0.5 are puzzling and may be due to some specific words in the database.

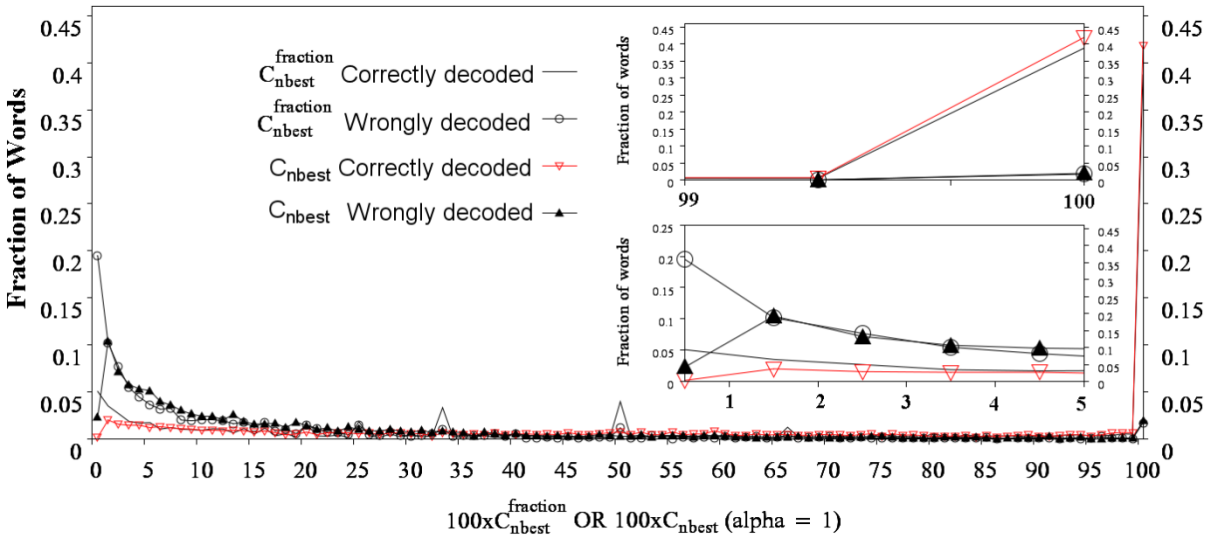

Figure 18: Fraction of total number of words Vs corresponding %CM values obtained for $C_{nbest}^{fraction}\ and\ C_{nbest}$ ($\alpha = 1$)

Chapter 6. **AgroAccess System Level Experiments**

The Version 1.0 of Response Validity Check block has been implemented in the following way in the AgroAccess system. I decided to use N-best fraction $C_{nbest}^{fraction}$ and N-best posterior probability C_{nbest} ($\alpha = 1$) CMs after they proved to be better than acoustic score normalisation based on phone recognition. Of course, there better ways of combining various confidence measures (e.g. decision trees, linear discriminant analysis), but as of now we went ahead with a simple scheme. Both these CMs need n-best list in order to generate them. But N-best list is created only when we use n-gram LM (and not FSG, n-best list creation module has not yet been written for sphinx3 decoder). Sometimes (rarely), the A* search which creates N-best list, fails to create N-best list (reason yet unknown, may be because of some lattice traversal problem). In such a case, we assume that the **utteranceConfidence** is 1. Another thing is that the top-level N-best list hypothesis doesn't always match the decoder o/p. We consider the top-level N-best list hypothesis for CM analysis, whenever it is available.

Phone triphone HMMs (2000 senones, 16 Gaussians) used in the AgroAccess system have been trained on the **marathiAgmark1500** database. The number of senones and Gaussians were optimized based on 3 fold cross-validation experiments.

The pseudo code for Response Validity Check block is as given below (Refer to Figure 3 to put things into perspective) –

If (speechDetector(inputWavFile) == 0) // i.e. speech detector doesn't find any speech
    utteranceConfidence = 0
elseif (A* search failed)
    utteranceConfidence = 1
else
    foreach $word_i$ in top-level N-best hypothesis
        if ($C_{nbest} < 0.98$ && $C_{nbest}^{fraction} < 0.49$)
            $\boldsymbol{wordConfidence(word_i) = 0}$
        elseif ($C_{nbest}^{fraction} < 0.60$)
            $\boldsymbol{wordConfidence(word_i) = 0.5}$
        else
            $\boldsymbol{wordConfidence(word_i) = 1}$
    endfor

utteranceConfidence = $\frac{\Sigma_i wordConfidence\ \ (word_i)}{number\ \ of\ words\ \ in\ hypothesis}$

If the sequence of words in hypothesis is not valid (due to n-gram search, insertions may occur / the speaker may have spoken words in a different sequence)

System rejects utterance

elseif utteranceConfidence < 1.00

System rejects utterance

else

System accepts utterance

The CM threshold values used in the above algorithm were obtained by trial and error on demo system (more sophisticated approach is necessary). Based on informal testing, the system seems to reject OOV words reasonably well. But it still gets confused a lot when the spoken utterance is phonetically very similar to other word in LM (Solapur and Kolhapur).

Another problem with this scheme is that if the user has spoken both in vocabulary and OOV words, then the utterance level CM may have a low value (due to OOV words) so we may have to reject the complete utterance. More systematic experiments on improving and evaluating the performance of this block are planned in immediate future.

Chapter 7. **Conclusions and Future Work**

7.1 Conclusion

This thesis started off with the introduction of Marathi AgroAccess system and described one aspect (Response Validity Check block) where various confidence measure techniques are needed.

In this thesis we reported experiments done in building a keyword spotting system. We learnt from those experiments that parameter tuning of an acoustic KWS is very time consuming, at the same time critical for the system. The KWS system should include a good post-processing module. Two post-processing modules were implemented, one based on Isolated Word Recognition and another using re-recognition with KW-Filler network. Latter one gave 7% increment in the Figure of Merit metric (w.r.t the baseline method).

We experimented with various forms of acoustic score normalisation techniques trying to bring phone dependence into picture and showed improvement in the performance over a baseline score normalisation scheme. We also evaluated two n-best list based confidence measures. Overall n-best list based CMs were superior to score normalisation based CMs.

We described the AgroAccess system and how confidence measures can be useful in the system. Though, this work is not yet complete at the time of writing this draft.

7.2 Future work

Implement more sophisticated confidence measures (for example posterior probability from word lattice)

More generalized scheme of confidence measure that can take into account various problem cases is needed to be useful in AgroAccess system. Also investigation into usefulness of keyword spotting block can be done.

Sphinx3 decoder does not create n-best list during finite state grammar search. Need to implement this module as FSG gives better performance than trigram LM based recognition, in an application like AgroAccess where syntax is simple. After implementing this module, n-best list based CMs can be used in conjunction with FSG decoding.

References

[1] "Asterisk open source toolkit for telephony applications", http://www.asterisk.org/, 1 June 2012

[2] "MySQL database management software", http://www.mysql.com/products/enterprise/database/, 1 June 2012

[3] "CMUSphinx speech recognition toolkit", http://cmusphinx.sourceforge.net, 1 June 2012

[4] M. Weintraub, "Keyword-spotting using SRI's DECIPHER large-vocabulary speech-recognition system," in *Proc. ICASSP, 1993*, vol. 2, pp. 463–466.

[5] D. A. James, S. J. Young, "A fast lattice-based approach to vocabulary independent wordspotting," in Proc. *ICASSP, 1994*, vol. 1, pp 377–380.

[6] K. Thambiratnam and S. Sridharan, "Dynamic match phone-lattice searches for very fast and accurate unrestricted vocabulary keyword spotting," in Proc. *ICASSP, 2005*, vol. 1, pp 465–468.

[7] J. R. Rohlicek, W. Russell, S. Roukos, and H. Gish, "Continuous hidden Markov modeling for speaker-independent word spotting," in *Proc. ICASSP 1989*, vol. 1, pp. 627–630.

[8] R. C. Rose and D. B. Paul, "A hidden Markov model based keyword recognition system," in Proc. *ICASSP 1990*, vol. 1, pp129-132

[9] P. Heracleous and T. Shimizu, "An Efficient Keyword Spotting Technique Using Complementary Language for Filler Models Training," in Proc. *EUROSPEECH, 2003*, pp. 921-924

[10] I. Szoke et al., "Comparison of keyword spotting approaches for informalcontinuous speech," in Proc. *Interspeech, 2005*, pp. 633–636

[11] I. Szoke, M. Fapso, L. Burget, J. Cernocký, "Hybrid Word-Subword Decoding for Spoken Term Detection," In Proc. *SSCS 2008*: *Speech Search Workshop at SIGIR*, 2008

[12] S. Young and W. Ward , "Learning New Words from Spontaneous Speech: Automatic Detection, Categorization and Acquisition", in Proc. *ICASSP*, 1993

[13] R. Sukkar, "Subword-based minimum verification error (SB-MVE) training for task independent utterance verification," in Proc. *ICASSP 1998*, pp 229–232

[14] T. J. Hazen, T. Burianek, J. Polifroni and S. Seneff, "Recognition confidence scoring for use in speech under-standing systems" in Proc. *ISCA ITRW Workshop on ASR*, Paris, France, 2000

[15] L. Mathan and L. Miclet, "Rejection of extraneous input in speech recognition applications, using multi-layer percep-trons and the trace of HMMs", in Proc. *ICASSP 1991*, pp. 93–96, 1991

[16] J. Pinto and R.N.V. Sitaram, "Confidence measures in speech recognition based on probability distribu-tion of likelihoods," in Proc. *Interspeech, 2005*

[17] H Jiang, "Confidence measures for speech recognition: A survey", *Speech Communication*, 2005

[18] F. Wessel, R. Schlüter, K. Macherey and H. Ney, "Confidence measures for large vocabulary continuous speech recognition", *IEEE Transactions on Speech and Audio Processing* , 2001

[19] S. J. Cox and R. C. Rose, "Confidence measures for the SWITCHBOARD database", in Proc. *ICASSP, 1996*

[20] S. Kwon and H. Kim, "Utterance verification using word voiceprint models based on probabilistic distributions of phone-level log-likelihood ratio and phone duration" *IEICE - Trans. Inf. Syst*, 2008, pp. 2746-2750

[21] C. Ma, C-H Lee, "A study on word detector design and knowledge-based pruning and rescoring", *Interspeech* , 2007

[22] L. Chase, "Error-responsive feedback mechanisms for speech recognizers", *Ph.D. Thesis*, School of Computer Science, Carnegie Mellon University, Pittsburgh, USA, 1996

[23] L. Chase, R. Rosenfeld and Wayne Ward, "Error-responsive modifications to speech recognizers: negative n-grams", in Proc. *ICSPL*, Yokohama, Japan, September 1994

[24] C. Uhrik and W. Ward, "Confidence metrics based on N-gram language model backoff behaviours", in Proc. *European Conference on Speech Communication Technology*,1997

[25] R. San-Segundo, B. Pellom, K. Hacioglu and W. Ward, "Confidence measures for spoken dialogue systems", in Proc. *ICASSP*, 2001

[26] R. Zhang and A. I. Rudnicky, "Word level confidence annotation using combinations of features", in Proc. *European Conference on Speech Communication Technology*, 2001

[27] L. Chase, "Word and acoustic confidence annotation for large vocabulary speech recognition," in Proc. *Eurospeech*, Rhodes, 1997

[28] J. Tejedor, S. King, J. Frankel, D. Wang, J. Colás, and J. Garrido, "A novel two-level architecture plus confidence measures for a keyword spotting system," *V Jornadas en Tecnología del Habla*, Bilbao, November 2008

[29] P. J. Moreno, B. Logan and B. Raj, "A boosting approach for confidence scoring", in Proc. *European Conference on Speech Communication Technology*, 2001

[30] T. Schaaf and T. Kemp, "Confidence measures for spontaneous speech recognition", in Proc. *ICASSP*, 1997

[31] M. C. Benitez, A. Rubio and A. Torre, "Different confidence measures for word verification in speech recognition", *Speech Communication*, 2000

[32] F. Wessel, K. Macherey and H. Ney "A comparison of word graph and N-best list based confidence measures", in Proc. *European Conference on Speech CommunicationTechnology*, pp. 315–318,1999

[33] M. Weintraub, "LVCSR log-likelihood ratio rescoring for keyword spotting," in *Proc. ICASSP 1995*, vol. 1, pp. 297–300

[34] F. Wessel, K. Macherey and R. Schluter, "Using Word Probabilities as Confidence Measures," in *Proc. ICASSP 1998*, vol. 1, pp. 225-228

[35] C. Ma and C. H. Lee, "A study on word detector design and knowledge based pruning and rescoring", in Proc *InterSpeech* , 2007

[36] C. Ma, "A detection-based pattern recognition framework and its applications," *Ph.D. Thesis*, Georgia Tech, April 2010

[37] S. Barnwal, K. Sahni, R. Singh and B. Raj, "Spectrographic seam patterns for discriminative word spotting", in Proc. *ICASSP*, 2012

[38] K. Samudravijaya, K. D. Rawat and P. V. S. Rao, "Design of Phonetically Rich Sentences for Hindi Speech Database," *J. Acoust. Soc. Ind.*, vol. XXVI, pp. 466-471, December 1998

[39] X. Huang, A. Acero and H.W. Hon, "Spoken Language Processing: a Guide to Theory, Algorithm, and System Development," Prentice-Hall, New Jersey, 2001, pp 610

[40] A. Ito, M. Kohda and S. Makino, "Fast optimization of language model weight and insertion penalty from n-best candidates," *Acoustical science and technology*, vol. 26, no. 4, pp 384–387, 2005

[41] J. G. Kim, H.Y. Jung and H.Y. Chung. "A keyword spotting approach based on pseudo n-gram language model." *9th Conference on Speech and Computer*, 2004

[42] S. A. Liu, "Landmark detection for distinctive feature basewd speech recognition," *J. Acoust Soc. Am.*, Vol 100, pp 3417-3430, 1996

[43] T. Godambe and K. Samudravijaya, "Speech data acquisition for voice based agricultural information retrieval, *39th All India DLA Conference*, Punjabi University, Patiala, June 2011

[44] Qi Li, J. Zheng, A. Tsai and Q. Zhou, "Robust endpoint detection and energy normalization for real-time speech and speaker recognition", in *IEEE Transactions on Speech and Audio Processing* 10(3): pp 146-157 (2002)

[45] Z. Rivlin, M. Cohen, V. Abrash and T. Chung, "A phone dependent confidence measure for utterance rejection", inProc. *ICASSP*, May 1996