

# KEYWORD SEARCH USING MODIFIED MINIMUM EDIT DISTANCE MEASURE

*Kartik Audhkhasi*

Electrical Engineering Department  
Indian Institute of Technology  
New Delhi, India

*Ashish Verma*

IBM-India Research Lab  
Block-I, Indian Institute of Technology  
New Delhi, India

## ABSTRACT

A popular approach for keyword search in speech files is the Phone Lattice Search [1] [2]. Recently Minimum Edit Distance (MED) has been used as a measure of similarity between strings rather than using simple string matching while searching the phone lattice for the keyword. In this paper, we propose a variation of the MED, where the substitution penalties are automatically derived from the phone confusion matrix of the recognizer, as compared to heuristic or class based penalties used earlier. The results show that the substitution penalties derived from the phone confusion matrix lead to a considerable improvement in the accuracy of the keyword search algorithm.

**Keywords** - Keyword search, speech processing, speech recognition, string matching, information retrieval.

## 1. INTRODUCTION

Keyword search in speech files has been one of the most challenging areas in the field of speech processing over past many years. Some of the important applications for keyword search are audio indexing and information retrieval [3] [4]. For example, with the help of keyword search, it becomes very easy to retrieve all those audio files where a customer has complained of *technical problems* with a product, or all those meetings where the *marketing strategies* of the company were discussed. The Video Mail Retrieval (VMR) project at the Cambridge University and Olivetti and Oracle Research Lab uses keyword search over the audio soundtrack for retrieving stored video documents [3]. Some of the important keyword search approaches are described in the following.

### 1.1. Classical keyword search approaches

- First and most straight-forward approach for keyword search is to use a large vocabulary continuous speech recognizer (LVCSR) to transcribe the speech or audio files into the corresponding text. To find the keyword, a simple text based search can be performed on the transcription using conventional text search algorithms. However, the problem with this approach is that due to the finite vocabulary of the recognizer, it will be unable to recognize Out of Vocabulary (OOV) words, such as, names, acronyms and foreign language words. Hence, these OOVs can not be searched using this approach.
- Another approach for keyword search is called Keyword HMM approach which uses an HMM for each of the keywords and a single "garbage model" for all the other words

[5]. The garbage model is trained over a large speech corpus. A network consisting of the keyword HMMs and the garbage model HMM in parallel is constructed. A time aligned sequence of keyword and garbage tokens is generated as a result of recognition from a given observation sequence.

This method has no limitation as far as the variety of keywords that can be searched is concerned. However, with every new keyword desired to be searched, not only a new HMM for the keyword needs to be trained but the garbage model also needs to be retrained which makes it difficult to use in some conditions.

- The most recent approach for keyword search is the Phone Lattice based approach, in which each node of the lattice is associated with a point in time in the speech utterance and each edge with a phone hypothesis giving the corresponding confidence score [1] [3]. The lattice stores multiple phone hypothesis between two time instants in the utterance. The keyword can now be searched in the phone lattice so generated.

The main advantage of this approach is that it provides more flexibility, *i.e.*, even if a keyword phone is not the best hypothesis between two nodes, it is still retained in the lattice. Moreover, since the phone lattice can be searched for any given phone sequence, there is no concept of vocabulary and even OOV words can be searched.

A variant of the phone lattice based approach is described in more detail in Section 2 as it forms the basis for the approach proposed in this paper.

## 2. DYNAMIC MATCH PHONE LATTICE SEARCH

Dynamic Match Phone Lattice Search is an approach derived from the Phone Lattice Search. It uses Minimum Edit Distance (MED) during lattice search to take into account phone recognizer insertion, deletion and substitution errors [2].

### 2.1. Minimum Edit Distance

The MED between two strings is defined as the minimum cost of converting one string to the other, given three basic operations; insertion, deletion and substitution, and their associated costs. The MED between two strings can be computed by a dynamic programming technique.

A two dimensional matrix,  $M(0, \dots, p)(0, \dots, q)$  is used to hold edit distance values, where  $p$  and  $q$  are the lengths of the two

strings,  $U$  and  $V$ . MED between  $U$  and  $V$  is computed as follows:

$$\begin{aligned} M(0)(0) &= 0 \\ M(i)(0) &= i * I; \quad i = 1, \dots, p \\ M(0)(j) &= j * D; \quad j = 1, \dots, q \\ M(i)(j) &= \min\{M(i-1)(j-1) + S(U(i), V(j)), \\ &\quad M(i-1)(j) + D, M(i)(j-1) + I\} \end{aligned}$$

where  $S$ ,  $I$  and  $D$  are substitution, insertion and deletion penalties respectively. At the time of keyword search in the lattice, all the phone sequences at an MED from the keyword less than a threshold, are declared as keyword hits.

## 2.2. Heuristic class based penalties

In [2], the penalties for substitution were decided on the basis of some rules based on the broad acoustic-phonetic classes. The IBM Indian English phone set has five broad acoustic classes: vowels (e.g. AA, AE, AEN, AAN), semi vowels (e.g. R, L), nasals (e.g. M, N), plosives (e.g. P, PD, B, BH) and fricatives (e.g. F, S, SH, ZH). For the baseline system, a set of heuristic rules for deciding the substitution penalties were developed based on the acoustic classes. These rules are slightly modified from the set of rules described in [2] and are as follows:

1. If the phones belong to the same acoustic class, they are checked for equivalence. A phone is equivalent to its aspirated version (e.g. S and SH are equivalent) and its nasalized version (e.g. AE and AEN are equivalent). Also, phones such as P and PD are equivalent since the latter is a word-end version of the former. Such phones pairs are given 0 substitution penalty. Else, they are assigned a substitution penalty of 1.
2. If the phones belong to different acoustic classes, substitution is deemed invalid by assigning an infinite penalty.
3. Insertion and deletion penalties were both set at  $I$ .

## 3. PROPOSED APPROACH

The heuristic class based penalties are not representative of the type of substitutions that actually occur at the time of lattice generation. This is due to the fact that the amount of confusion between different phone models also depends upon the amount of training data, nature of training data, richness of the co-articulation captured for the phone and so on. The exact nature of the confusion between different phone models can be learned in the form of a phone confusion matrix.

The phone confusion matrix is generated by comparing the phonetic transcription of a large speech corpus generated by the recognizer with the corresponding true phonetic transcription. The proposed approach derives the substitution penalties from confusion matrix scores rather than on the basis of rules as proposed in [2]. Now we discuss the phone confusion matrix followed by the proposed method to derive substitution penalties from it.

### 3.1. Phone confusion matrix

The confusion matrix  $C$  is a  $N \times N$  matrix (where  $N$  is the number of phones in the phone set). The entry  $C(i, j)$  indicates the number of times the recognizer substituted the  $i^{th}$  phone by the  $j^{th}$  phone

as a fraction of the total number of recognized instances of the  $i^{th}$  phone. Table 1. shows a section of the confusion matrix.

It is easy to realize the importance of the confusion matrix scores. If the value of  $C(i, j)$  is more than that of  $C(i, k)$ , then the  $i^{th}$  is more likely to be substituted by the  $j^{th}$  than by the  $k^{th}$  phone.

AA	AX 0.0684	AO 0.0605	ER 0.0363	AE 0.0318	AW 0.0287
DH	D 0.0826	DHH 0.0545	B 0.0353	TX 0.0231	EH 0.0168
SH	S 0.0207	ZH 0.0096	CH 0.0083	F 0.0069	JH 0.0069
M	N 0.0709	NG 0.0288	V 0.0279	B 0.0174	W 0.0109
ZH	Z 0.0833	SH 0.0417	JH 0.0417	EY 0.0417	Y 0.0417

**Table 1.** A section of the confusion matrix showing the top 5 confusing phones for AA, DH, SH, M and ZH.

Hence, the substitution penalty for the phone pair  $(i, j)$  must be set less than that for  $(i, k)$ , irrespective of the acoustic classes to which phones  $i, j$  and  $k$  belong. This is the key motivation behind using the confusion matrix for deriving the substitution penalties. Deriving the substitution penalties from the confusion matrix promises to improve word search accuracy as it not only takes into account the real errors made by the recognizer, but also smoothes out the MED scores.

### 3.2. Determining substitution penalties

One straightforward way to derive the substitution penalty from the confusion matrix would be to directly use the value  $1 - C(i, j)$  as the substitution penalty for the phone pair  $(i, j)$ . However, it was found that the penalties obtained in this manner were very close to each other.

Hence, the substitution penalty for a phone pair  $(i, j)$  was now found out as under:

$$S(i, j) = \log\{C(i, i)/C(i, j)\}; i \neq j \quad (1)$$

Since in most cases  $C(i, i) \gg C(i, j)$ , the  $\log(\cdot)$  function is used to keep the substitution penalties within limits. It can be seen that the function in (1) provides enough variance among the substitution penalties for various phone pairs as compared to directly using the values  $C(i, j)$ . It must be noted that  $S(i, j) \neq S(j, i)$  which was not the case with standard MED described in [2]. For practical purposes, the substitution penalty for only the top  $M$  confusing phones may be computed while for the rest, substitution may be deemed impermissible by setting an infinite penalty.

### 3.3. Phone lattice representation

In our implementation, a phone lattice is represented as a sequence of nodes, one for each instant of time in the utterance. The node stores information about the following:

1. List of phones ending at this node and their number. Each phone contains the following information:
  - (a) The label of the phone.

- (b) The time normalized score, which is obtained from a weighted sum of the acoustic and the language model scores.
  - (c) The index of the starting node of the phone.
2. List of candidate sequences ending at this node and their number. Each candidate sequence contains the following information:
- (a) The phone string for the sequence.
  - (b) The time normalized score for the sequence.
  - (c) The index of the starting node of the sequence.
  - i. Check if  $U_{k,t}$  is a proper prefix of  $K$ . If yes, then keep it for future use without computing its MED with the keyword.
  - ii. If not, then compute  $MED(K, U_{k,t})$ . Let it be denoted by  $m$ .
  - iii. If  $m \leq T_2$  and  $L_K - b \leq L_{U_{k,t}} \leq L_K + b$  (where  $b$  is a constant), declare a keyword hit.
  - iv. Else, if  $m \leq T_1$  and  $0 \leq L_{U_{k,t}} < L_K - b$ , then keep  $U_{k,t}$  alive.
  - v. Else, discard the string  $U_{k,t}$ .

#### 4. EXPERIMENTS AND RESULTS

##### 3.4. Phone lattice search algorithm

The proposed phone lattice search algorithm is a modification of the DMPLS algorithm proposed in [2]. Define  $L_U$  as the number of phones in the string  $U$ . Let  $K = \{k_1, \dots, k_{L_K}\}$  be the keyword phone sequence that has to be searched in a lattice. Also, let at every node, only the best  $D$  ending phones be selected according to their time normalized acoustic score.

Let  $T_1$  be the MED threshold below which all candidate strings are kept alive and  $T_2$  be the maximum MED that a candidate string can have with the keyword for it to be declared a hit. Let both the insertion and deletion penalties be  $I$ . Let  $SC_U$  and  $SC_p$  denote the scores of string  $U$  and phone  $p$  respectively and let  $U_{i,t}$  denote the  $i^{th}$  candidate string ending at node  $t$ . Also, define  $N_t$  as the number of candidate sequences ending at the node at time  $t$ .

Then, the phone lattice search algorithm is as follows:

1. Start from node at time  $t = 0$ .
2. For the node at time  $t$ , do the following:
  - (a) Initialize  $N_t = 0$ .
  - (b) Select the best  $D$  phones ending at this node on the basis of their scores. Let these phones be  $p_{1,t}, \dots, p_{D,t}$ .
  - (c) For a phone  $p_{i,t}$  (where  $i = 1, \dots, D$ ) do the following:
    - i. If  $MED(p_{i,t}, k_1) \leq a$  (where  $a$  is a constant), create a new candidate sequence for this node,  $U_{i,t} = \{p_{i,t}\}$ . Set  $SC_{U_{i,t}} = SC_{p_{i,t}}$  and  $N_t = N_t + 1$ .
    - ii. Visit the starting node of the phone  $p_{i,t}$ , say  $t'$ . Let  $\{U_{1,t'}, \dots, U_{N_{t'},t'}\}$  be the list of candidate sequences for the node at time  $t'$ .
    - iii. Using a candidate sequence  $U_{j,t'}$  (where  $j = 1, \dots, N_{t'}$ ) create a new candidate sequence  $U_{k,t}$  (where  $k = N_t + 1, \dots, N_t + N_{t'}$ ) by appending the phone  $p_{i,t}$  to  $U_{j,t'}$ . Set

$$SC_{U_{k,t}} = \frac{\{L_{U_{j,t'}} SC_{U_{j,t'}} + SC_{p_{i,t}}\}}{L_{U_{j,t'}} + 1} \quad (2)$$

Also set  $N_t = N_t + 1$  and  $L_{U_{k,t}} = L_{U_{j,t'}} + 1$ .

- (d) At this step, we have all the candidate sequences  $U_{k,t}$  (where  $k = 1, \dots, N_t$ ) ending at node  $t$ .
- (e) Now, for a candidate sequence  $U_{k,t}$  do the following:

HTK was used for generating phone lattices [6]. 65 monophone acoustic models (3 state, first-order and left to right HMMs with a 3 component Gaussian mixture at each state) were trained in HTK using approximately 24000 audio files from an Indian English speech database collected in-house at IBM India Research Lab. The audio files were sampled at 22KHz and had a duration of 3 to 12 seconds each. A 39 component feature vector was used which contained MFCC, delta and delta-delta coefficients (12 each) appended with energy values. A bigram language model was also trained in HTK using the time aligned phonetic transcriptions of the audio files, which were generated from the available word level transcriptions.

A set of 6000 audio files (22 KHz PCM, each of duration 4 to 12 seconds), different from the training set, was used for evaluating the monophone recognition accuracy of the recognizer. The audio files were in the form of sentences read out by different speakers. The monophone recognition accuracy was found out to be around 55.5%.

Phone lattices were generated in HTK using 10 tokens per state. For the purpose of quantifying the performance of the keyword search algorithms, standard Information Retrieval metrics, namely recall and precision were used.

##### 4.1. Keyword search experiments

Keyword search was performed on a speech database collected in-house at IBM-India Research Lab, by asking the speakers to speak for 2 to 3 minutes each on a given topic. The test set contained 21 speakers, totalling a duration of around 50 minutes. A total of 33 keywords were selected at random, some of which are: "American", "computer", "economy", "India", "journalism", "outsourcing", "recession". The number of phones in the keywords ranged from 5 to 11.

There were 122 keyword occurrences in the complete test set. Whenever a keyword was detected by the program, its time location was verified by hearing the audio file. If the detected location was off by more than 70% of the keyword length, the detection was classified as a false alarm.

The insertion and deletion penalties were both set equal to 4.0, with  $T_1 = 2.0$  and  $T_2 = 1.5$ . The best 10 phones ending at a node were selected ( $D = 10$ ). The constants  $a$  and  $b$  mentioned in 3.4 were both set equal to 1. All these parameters were set heuristically after repeated keyword search experiments.

The results for the proposed phone lattice search algorithm with standard MED (based upon heuristic class based penalties) and modified MED (using the confusion matrix derived penalties) for  $D = 10$  are given in Table 2, in terms of precision rate, recall

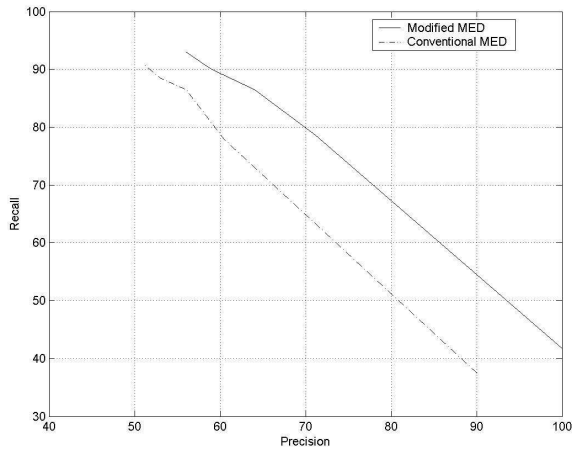


Fig. 1. The Recall-Precision curves

rate and the number of false alarms per keyword occurrence per hour. Figure 1 gives the Recall-Precision curves for conventional MED and modified MED obtained by varying  $D$  between 5 and 20.

Metrics	Standard MED	Modified MED
Precision rate	0.605	0.711
Recall rate	0.779	0.787
FAs\kw\hr	0.474	0.347

Table 2. Keyword search results for  $D = 10$

As can be observed from Table 2, the modified MED proposed in this paper has performed better than conventional MED. In Figure 1, the Recall-Precision curve for the modified MED based algorithm is always above the corresponding curve for the conventional MED based algorithm. This shows that at identical recall rates, modified MED based algorithm provides better precision than the conventional MED based algorithm.

## 5. CONCLUSION

The reported results show that the proposed modified MED measure has improved precision and hence has made keyword search more accurate. The accuracy of the phone lattice search algorithm with modified MED measure is due to the fact that it derives its substitution penalties from the confusion matrix of the recognizer, rather than based on some heuristic class-based penalties. Hence, it is able to take into account some important factors like the amount and nature of training data and richness of the co-articulation captured for the phone, which is not the case in a conventional MED based approach.

## 6. REFERENCES

- [1] D.A. James and S.J. Young, "A fast lattice-based approach to vocabulary independent wordspotting," *IEEE International Conference on Acoustics, Speech and Signal Processing*, vol. 1, pp. 377–380, 1994.
- [2] K. Thambiratnam and S. Sridharan, "Dynamic match phone lattice searches for very fast and accurate keyword spotting," *IEEE International Conference on Acoustics, Speech and Signal Processing*, pp. 465–468, 2005.
- [3] S.J. Young, M.G. Brown, J.T. Foote, G.J.F. Jones, and K.S. Jones, "Acoustic indexing for multimedia retrieval and browsing," *IEEE International Conference on Acoustics, Speech and Signal Processing*, 1997.
- [4] G.J.F. Jones, J.T. Foote, K.S. Jones, and S.J. Young, "Retrieving spoken documents by combining multiple index sources," *Proc. SIGIR*, pp. 30–38, 1996.
- [5] J.G. Wilpon, L.R. Rabiner, C.H. Lee, and E.R. Goldman, "Automatic recognition of keywords in unconstrained speech using Hidden Markov Models," *IEEE Transactions on Acoustics, Speech and Signal Processing*, 1990.
- [6] S.J. Young et al., "The HTK book," *Cambridge University Engineering Department*, 2005.
- [7] L.R. Rabiner, "A tutorial on Hidden Markov Models and selected applications in speech recognition," *Proceedings of the IEEE*, vol. 77, pp. 257–286, 1989.
- [8] F. Siede, Y. Peng, C. Ma, and E. Chang, "Vocabulary-independent search in spontaneous speech," *IEEE International Conference on Acoustics, Speech and Signal Processing*, 2004.
- [9] R.C. Rose and D.B. Paul, "A Hidden Markov Model based keyword recognition system," *IEEE International Conference on Acoustics, Speech and Signal Processing*, pp. 129–132, 1990.
- [10] B.H. Juang and L.R. Rabiner, "Fundamentals of speech recognition," Pearson Education, 2003.
- [11] M. Riley and A. Ljolje, "Lexical access with a statistically-derived phonetic network," *Human Language Technology Conference, Proceedings of the workshop on Speech and Natural Language*, pp. 289–292, 1991.
- [12] K. Livescu and J. Glass, "Lexical modeling of non-native speech for automatic speech recognition," *IEEE International Conference on Acoustics, Speech and Signal Processing*, 2000.