



Metric Subspace Indexing for Fast Spoken Term Detection

Taisuke Kaneko¹, Tomoyosi Akiba¹

¹Department of Computer Science and Engineering
Toyohashi University of Technology

kaneko@nlp.cs.tut.ac.jp, akiba@nlp.cs.tut.ac.jp

Abstract

In this paper, we propose a novel indexing method for Spoken Term Detection (STD). The proposed method can be considered as using metric space indexing for the approximate string-matching problem, where the distance between a phoneme and a position in the target spoken document is defined. The proposed method does not require the use of thresholds to limit the output, instead being able to output the results in increasing order of distance. It can also deal easily with the multiple candidates obtained via Automatic Speech Recognition (ASR). The results of preliminary experiments show promise for achieving fast STD.

Index Terms: Spoken Term Detection, Approximate String Matching, Metric Space Indexing, Hough Transform

1. Introduction

STD is the task that, given a spoken document and a query term, finds the positions in the document where the term occurs. In recent times, STD has been actively studied in the context of speech processing. In 1997, STD (called *Known Item Retrieval* at that time) was first evaluated extensively at the TREC Spoken Document Retrieval Track [1]. From 2006 onwards, STD has been re-evaluated in the NIST Spoken Term Detection Evaluation [2], focusing particularly on STD for Out of Vocabulary (OOV) terms. In Japan, a project to construct an STD test collection is under way, as a part of the activity providing resources to promote research in Spoken Document Processing [3].

In this paper, we propose a novel indexing method for STD. The proposed method can be considered as using metric space indexing for approximate string matching problem, where the distance is defined between a phoneme and a position in the target spoken document. The proposed method can also be considered as applying the Hough transform, an algorithm for detecting straight lines in a given visual image, to the STD task. Whereas the target image is available only at detection time in the case of image processing, the target document is considered to be known beforehand for STD. We propose a novel metric space indexing method for the preprocessing, where phonemes are indexed at positions in the spoken document within a metric space defined by the distances between the phonemes and the positions.

The most attractive advantage of the proposed method is that it does not need a threshold for the distance used to make the decision about whether the detected term is adopted or not. It can simply output the detection results in increasing order of distance. Another advantage is that it can deal naturally with the multiple recognition candidates obtained by ASR, because it indexes the distance between a phoneme and a position instead of between phonemes.

In the next section, we outline previous work related to STD. In Section 3, we describe the proposed method based on the Hough transform. In Section 4, we present our experimental results. Finally, in Section 5, we give our conclusions.

2. Related Work

2.1. STD

Most conventional methods for STD use Large Vocabulary Continuous Speech Recognition (LVCSR) to transcribe the target spoken document into textual form and then apply a text-based indexing method to find the positions in the text where the query term appears. Within this framework, one of the problems is how to deal with speech recognition errors. The TREC Spoken Document Retrieval (SDR) Track evaluation has revealed that using multiple candidates can compensate for recognition errors and can improve the STD performance. In addition to this investigation, several methods for representing multiple ASR results efficiently, such as Confusion Networks [4] and Time-Anchored Lattice Expansion [5], have been investigated for STD.

Saying about detection algorithms, many previous STD works simply use continuous Dynamic Programming (DP) matching, without using indexes. In order to improve time efficiency, some others use inverted files for indexing as in text retrieval. However, the inverted file requires the indexes not to include any errors, because an index that has some errors will not be retrieved at all from the corresponding query without error. To deal with this mismatch problem, methods have been proposed that use subwords such as phonemes or syllables for the indexing unit obtained by the subword-based ASR. However, the recognition accuracy of the subword-based ASRs is lower than that of the word-based ASRs. Another problem with using inverted files is that the number of indexes increases and the retrieval efficiency decreases when we index multiple recognition candidates.

Recently, Katsurada et al. [6] proposed to use a Suffix Array for indexing the spoken document in the STD task. With common substrings appearing in a target document being shared in the tree structure of the Suffix Array, DP matching on the Suffix Array enables efficient STD. However, it would seem difficult to apply Suffix Array indexing to multiple recognition candidates in the target spoken document, because of its characteristic reliance on compression in the target document.

Moreover, all the previous indexing methods for STD use binary indexing, which expresses only appearance or nonappearance. Therefore, these methods need to calculate distances to filter out implausible results either during or after using the indexes for searching.

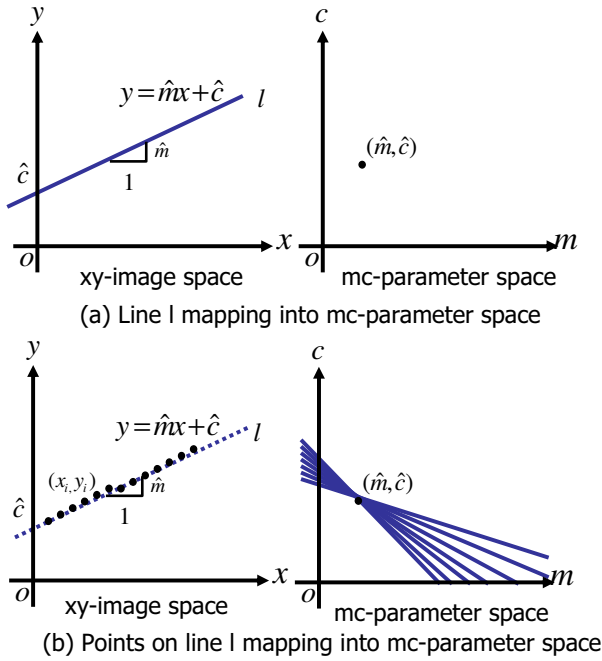


Figure 1: *The Hough transform.*

2.2. Approximate String Matching

In the text-processing field, the task of finding the occurrences of a given pattern (string) in a given text (string), while allowing for some errors in the matching, is called *approximate string matching*. The methods for approximate string matching can be classified into two groups. One is the online method, where the pattern is preprocessed but the text is not. The other is the off-line method, where the text is also preprocessed by indexing. Indexing methods used for off-line approximate string matching are classified into (1) those based on the Suffix Array or Suffix Tree, (2) those based on an n-gram or n-sample index, and (3) those based on metric space indexing [7]. If these methods were to be applied to STD, two problems would arise, namely that method (1) cannot deal with the multiple candidates obtained by ASR, and that method (2) requires the inclusion of at least one n-gram or n-sample index completely matched with those in the input query. On the other hand, method (3) has not been yet been applied to STD in previous work.

2.3. The Hough Transform

The proposed method can be considered as using the Hough transform, which is a method for machine recognition of lines in photographs or other pictorial representations [8]. We can express line l via $y = \hat{m}x + \hat{c}$, where its slope is \hat{m} and the y -intercept is \hat{c} in the xy image space. This means that the line l can be expressed as one point (\hat{m}, \hat{c}) in the mc parameter space (see Fig. 1(a)). For a given point (x_i, y_i) on the line l in xy image space, a line is drawn in the transformed mc parameter space by following equation.

$$\hat{c} = -x_i\hat{m} + y_i \quad (1)$$

Then candidate lines are drawn in mc -parameter space that intersect at one point (\hat{m}, \hat{c}) . (see Fig. 1(b)). By detecting this intersection point, we can obtain the line in xy image space.

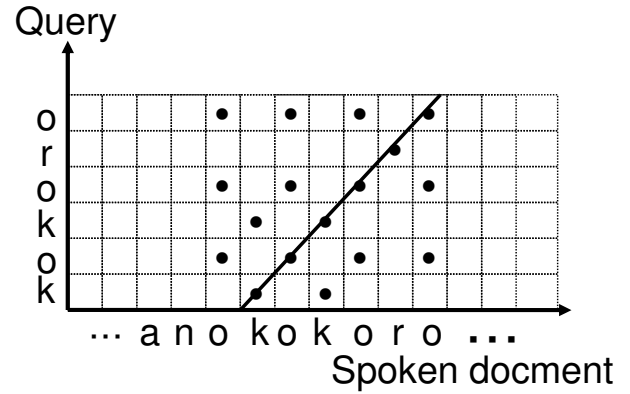


Figure 2: *STD as straight line detection.*

The method for detecting straight lines using this approach is called the Hough transform.

3. STD by Metric Subspace Indexing

Consider a plane where the x and y axis correspond to the phoneme sequence of the spoken document obtained by using ASR and the phoneme sequence of the input query, respectively (see Fig. 2). For each grid point on the plane, the distance between the phoneme in the document at x and the phoneme in the query at y is defined. The distance at a grid point is analogous to the pixel density at an image data point.

Let m be the query length (number of phonemes in the query), let n be the spoken document length (number of phonemes in the spoken document), and let $D_{i,j}$ ($0 \leq i < m, 0 \leq j < n$) be the phoneme distances defined at the grid point (i, j) on the plane. Then the STD problem can be recognized as the line detection problem on the plane and can be formulated as detecting the position j that has the minimum cumulative distance T_j , defined as follows.

$$T_j = D_{0,j} + D_{1,j+1} + \dots + D_{m-1,j+m-1} = \sum_{i=0}^{m-1} D_{i,i+j}$$

In the case of line detection in image data, the pixel densities can be processed only at detection time, because the target image data are not known in advance. In the case of STD, however, the distances $D_{i,j}$ can be processed beforehand, because the target spoken document is known in advance. Let $D(a)_j$ be the distance between a phoneme a and the phoneme that appears at position j in the target spoken document. Then, for each a , the phoneme distance vector $[D(a)_0, D(a)_1, \dots, D(a)_j, \dots, D(a)_{n-1}]$ can be calculated in advance. When a query is supplied, we can easily construct the xy plane by arranging the distance vectors along the y -axis according to the phoneme sequence of the query, so that $D_{i,j} = D(a_i)_j$ for the query $a_0a_1 \dots a_{m-1}$. Here the metric space defined between the query string and every substrings in the target document is divided into the metric subspaces, each of which is defined between each phoneme in the query and every positions in the document.

Furthermore, **the phoneme distance vector can be sorted in advance.** Let S_a be the position vector of a phoneme a that is obtained by sorting the initial position vector $[0, 1, \dots, j, \dots, n-1]$ of the target spoken document according to the phoneme distance vector

$[D(a)_0, D(a)_1, \dots, D(a)_j, \dots, D(a)_{n-1}]$. S_a is handled as a stack, where the stack top is the leftmost element of the vector. Using this S_a as the index, we can obtain a fast STD algorithm that can output the detection results in increasing order of distance (see Fig. 3). The algorithm can be expressed as follows.

1. According to the query phoneme sequence $a_0 a_1 \dots a_i \dots a_{m-1}$, prepare the stack sequence $S_{a_0} S_{a_1} \dots S_{a_i} \dots S_{a_{m-1}}$, each element of which has been calculated and sorted in advance. Initialize the counter $C[j] = 0 (0 \leq j < n)$ and the candidate set $U = \{\}$.
2. Pop the top element s_{a_i} of the stack S_{a_i} that has the minimum distance $\min_i D(a_i)_{s_{a_i}}$ from the set that comprises the top elements $s_{a_0} s_{a_1} \dots s_{a_i} \dots s_{a_{m-1}}$ of the stack sequence $S_{a_0} S_{a_1} \dots S_{a_{m-1}}$. Let $j = s_{a_i} - i$ and add 1 to $C[j]$.
3. If $C[j] \geq k$, then add the position j to the set U .
4. Output the subset V of U that satisfy a certain condition. Let $U \leftarrow U - V$.
5. Repeat Steps 2, 3 and 4 until a certain condition is satisfied.

The simplest version of the above algorithm is to set $k = m$ (the query length) at Step 3 and to impose no condition at Step 4. Note that this algorithm does not need a threshold for distances. It outputs the detection results approximately in the order of smaller to larger distances¹. We could use different conditions at Step 5, such as “until it finds the first result”, “until it finds the N-best results”, “until it passes a certain period”, and, of course, “until the distance exceeds a certain threshold.”

3.1. Dealing with Insertion and Deletion Errors

Because an actual ASR result contains recognition errors, the line’s slope does not necessarily become exactly 1 (see Fig. 4). To deal with the insertion and deletion caused by recognition errors, we can introduce an alternative distance measure that incorporates phoneme neighbors in $D(a)_j$ (see Eq. 2, 3).

$$D(a)_j = \min \begin{cases} d(a, b_{j-1}) \\ d(a, b_j) \\ d(a, b_{j+1}) \end{cases} \quad (2)$$

$$D(a)_j = \frac{\beta d(a, b_{j-1}) + \alpha d(a, b_j) + \beta d(a, b_{j+1})}{\alpha + 2\beta} \quad (3)$$

Here, a is a phoneme in a query, b_j is a phoneme of the position j in the spoken document, $d(a, b)$ is the distance between phoneme a and b , and α, β are weights introduced to balance the terms.

3.2. Dealing with Multiple Candidates

The recall of the detection can be improved by considering the multiple candidates from the speech recognition [1]. The proposed method can cope easily with the “sausage”-like representation of multiple recognition candidates, similarly to the Confusion Network [4] and TALE [5], by redefining the phoneme distance $D(a)_j$ as follows.

$$D(a)_j = \min_{b \in B_j} w(b) d(a, b), \quad (4)$$

¹This simplest algorithm does not guarantee to output the results precisely in order of distance. However, the more complex version of the algorithm, which sets $k = 1$ at Step 3 and imposes $V = \{j | T_j < \sum_{i=0}^{m-1} D(a_i)_{s_{a_i}}\}$ at Step 4, can output the results precisely in order of distance.

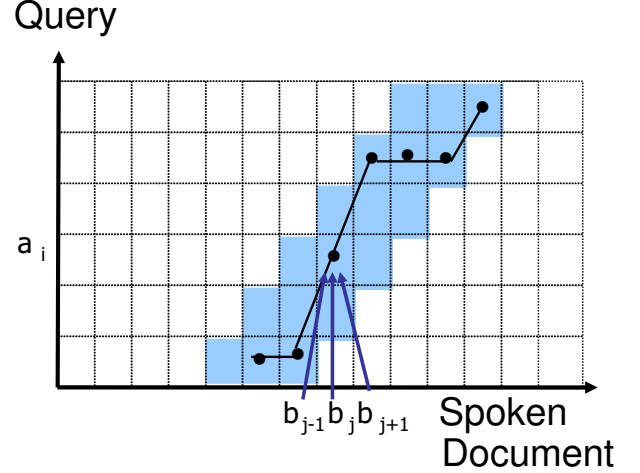


Figure 4: Dealing with insertion and deletion errors.

where B_j is the set of multiple phoneme recognition candidates at the position j in the target spoken document and $w(b)$ is the confidence weight of the phoneme recognition candidate b . For $w(b)$, we can use various measures, including recognition likelihood and a posteriori probability.

4. Experiments

We conducted some preliminary experiments to investigate the potential of the proposed indexing method.

4.1. Experimental Setup

We used the Japanese STD test collection [3] for our evaluation. The target document is one hundred lectures selected from the Corpus of Spontaneous Japanese [10]. We used the automatic transcription of the target document included in the collection, which was obtained by using a word-based LVCSR. The word error rate and the phoneme error rate were about 21% and 17%, respectively. Fifty in-vocabulary (IV) words and fifty OOV words were used as the queries for STD.

The Distinctive Phonetic Feature (DPF) [9] was used for calculating the distance between phonemes. DPF defines a phoneme as a Boolean 15-dimensional feature, based on the manner of articulation, place of articulation, etc. We used the Hamming distance between DPFs for the distance between phonemes, as in [6].

As a baseline method, we used the continuous DP matching described as follows.

$$D_{0,j} = d(a_0, b_j) (i = 0, 0 \leq j < n)$$

$$D_{i,0} = D_{i-1,0} + d(a_i, q_0) (0 \leq i < m, j = 0)$$

$$D_{i,j} = \min \begin{cases} D_{i-1,j-1} + d(a_i, b_j) \\ D_{i-1,j} + d(a_i, b_j) \\ D_{i,j-1} + d(a_i, b_j) \end{cases} \quad (5)$$

Here, $d(a, b)$ is the distance between the phonemes a and b , and $D_{i,j}$ is the cumulative distance for continuous DP matching.

As the proposed method, we implemented the simplest version of the algorithm described in Section 3.

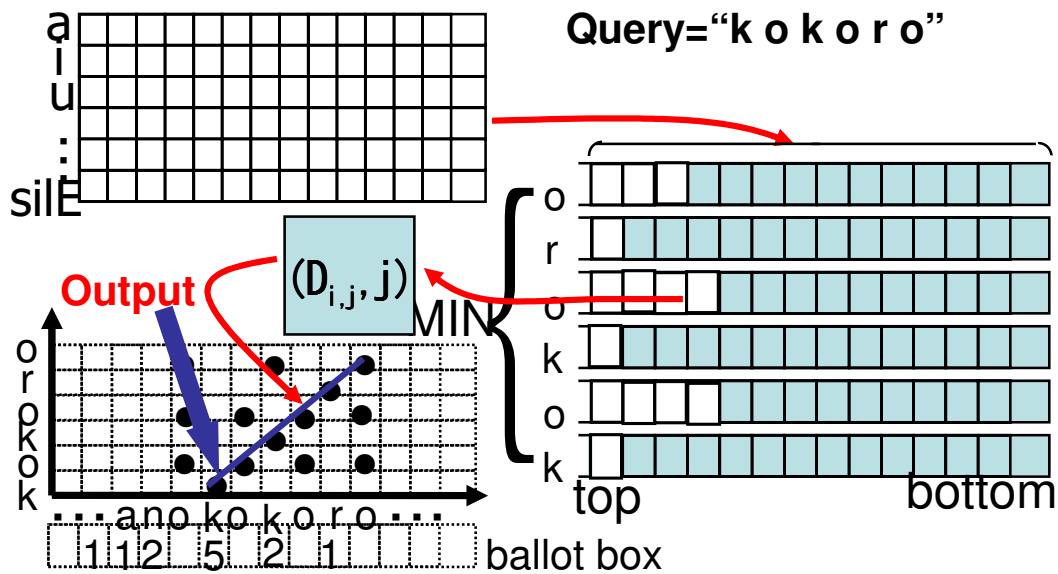


Figure 3: Process for the proposed STD.

Table 1: Experimental results.

Detection set	IV word		OOV word	
	Baseline	Proposed	Baseline	Proposed
Precision	0.96	0.96	0.097	0.103
Recall	0.57	0.57	0.27	0.27
Processing time [ms]	640.2	220.4	620.6	222.6

4.2. Experimental Results

For the baseline and the proposed method, we set the threshold for the distance to zero, to be able to compare their performances. Table 1 shows the results in terms of precision, recall, and processing time. It indicates that the proposed method promises a fast STD, because it boosts the search efficiency without losing detection performance.

We also investigated the number of zero-valued elements in the phoneme distance vector $D(a)_j$ for the input queries. The average proportion of zero-valued elements among all elements was about 6%. This means that, theoretically, we can reduce the processing time to 6% of that obtained by the baseline method, which does not use indexes. It indicates that there is room for improvement in the implementation.

5. Conclusion

We have proposed a novel indexing method for STD based on the Hough transform. The most attractive advantage of the proposed method is that it does not need a threshold for distances, instead outputting the detection results in increasing order of distance. The results of preliminary experiments showed promise for achieving fast STD, although they also showed that the method's implementation had room for improvement.

To improve the efficiency further, we need to choose a better unit for indexing and a better distance measure. Using the phoneme as the index unit, there remain 6% of the elements in the indexes that are zero-valued, which are unavoid-

ably searched. We think that using a larger unit, such as the syllable or syllable n-gram, will improve the efficiency. In addition, the Hamming distance for the DPFs used in this work is rather too discrete. We think that we could use a more continuous distance to improve the performance, such as the Bhat-tacharyya distance between two acoustic models.

The proposed method can output the results in increasing order of their plausibility, which is not possible by the previous methods. Therefore, it can be used to output N-best results or to output the results until it passes a certain period. We will also investigate the application area suited for this advantage.

6. References

- [1] J.S. Garofolo, et al., "The TREC Spoken Document Retrieval Track: A Success Story", In Proceedings of TREC-9, pp. 107–129, 1999.
- [2] NIST, "The Spoken Term Detection Evaluation": <http://www.itl.nist.gov/iad/mig/tests/std/2006/index.html>
- [3] Y. Itoh, et al., "Constructing Japanese Test Collections for Spoken Term Detection", In Proceedings of INTERSPEECH, 2010.
- [4] L. Mangu, et al., "Finding Consensus in Speech Recognition: Word Error Minimization and Other Applications of Confusion Network", Computer Speech and Language, Vol. 14, No. 4, pp. 373–400, 2000.
- [5] P. Yu, et al., "Approximate Word-Lattice Indexing with Text Indexers: Time-Anchored Lattice Expansion", In Proceedings of ICASSP, pp. 5248–5251, 2008.
- [6] K. Katsurada, et al., "Fast Keyword Detection Using Suffix Array", In Proceedings of INTERSPEECH, pp. 2147–2150, 2009.
- [7] G. Navarro, et al., "Indexing Methods for Approximate String Matching", IEEE Data Eng. Bull., 24(4): pp. 12–27, 2001.
- [8] P.V.C. Hough, "Method and Means for Recognizing Complex Patterns", U.S. Patent 3069654, 1962.
- [9] T. Fukuda, et al., "Orthogonalized Distinctive Phonetic Feature Extraction for Noise-Robust Automatic Speech Recognition", IE-ICE Trans., Vol. E87-D, No. 5, pp. 1110–1118, 2004.
- [10] K. Maekawa, H. Koiso, S. Furui, H. Isahara, "Spontaneous Speech Corpus of Japanese", In Proceedings of LREC, pp. 947–952, 2000.