# DTW-Distance-Ordered Spoken Term Detection

*Teppei Ohno*[1], *Tomoyosi Akiba*[1]

[1]Department of Computer Science and Engineering
Toyohashi University of Technology
ohnoteppei@nlp.cs.tut.ac.jp, akiba@cs.tut.ac.jp

## Abstract

The amount of Web-based multimedia data that includes speech is increasing rapidly. Spoken term detection (STD) enables rapid identification of desired-information candidates from a large quantity of speech data. Considering that these STD candidates ultimately have to be checked one at a time by the user, a long list of candidates is not desirable. However, setting an appropriate cutoff threshold for a particular STD request beforehand is not easy. In this work, we propose a novel indexing and search method for STD that requires no cutoff threshold for detection but can output detection results in increasing order of their dynamic time warping (DTW) distances for a given query term. Our experimental evaluation showed that, whereas using the strict algorithm for our method gave detection results that were exactly in increasing order of their DTW distances, its relaxed variants were able to execute much faster at the cost of only a slight loss in the exact ordering.

**Index Terms**: spoken term detection, approximate string matching, metric subspace indexing, dynamic time warping.

## 1. Introduction

Given a query term and a spoken document, spoken term detection (STD) is the task of finding the position in the document at which the term is uttered. With the quantity of Web-based multimedia data increasing rapidly, STD can be considered as a fundamental technology for retrieving multimedia data that includes speech tracks. It has been studied actively in the context of speech processing.

In 1997, a task called known-item search, similar to the present-day STD task, was evaluated in the TREC Spoken Document Retrieval Track [1]. In 2006, STD was evaluated in the NIST Spoken Term Detection Evaluation [2], focusing particularly on STD for out-of-vocabulary (OOV) terms. In 2010, a project to construct an STD test collection targeting Japanese spoken lectures was conducted [3]. By extending the test collection, STD was evaluated as one of the subtasks of the spoken-document retrieval task in the evaluation campaign for NTCIR-9 SpokenDoc [4] and for the upcoming NTCIR-10 SpokenDoc-2.

Until now, many studies of STD have focused on improving search accuracy and time efficiency. To improve time efficiency, some proposals have introduced indexing methods similar to those used in text retrieval. The two principal methods used in text retrieval involve inverted files [5][6][7] or suffix arrays [8]. In an actual application, other than time efficiency, it is also important to output search-result candidates in order of their confidence measure. Considering that the detection candidates ultimately have to be checked one at a time by the user, a long list of candidates is not desirable.

In a method that presented candidates in order of confidence measure, Kaneko and Akiba [9] proposed a novel indexing method for STD called metric subspace indexing. The method can be considered as using metric space indexing for an approximate string-matching problem, where the distance is defined to be between a phoneme and a position in the target spoken document. One of the distinctive advantages of the method was that it could output the detection results in increasing order of distance, which corresponded to their confidence measure, without requiring a predefined threshold for the distance. We evaluated a simple implementation of the method as described in [9]. However, we found that this implementation offered poor detection performance.

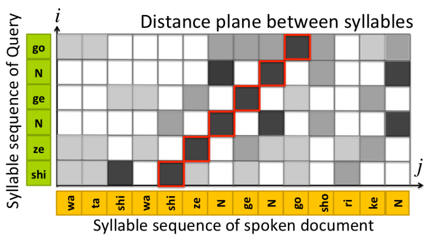
In this work, instead of the previous simple implementation, we implement a strict algorithm that outputs the candidates in exact order of their distances, which has already been proposed but not evaluated in [9]. To further improve the STD performance, we extend the previous method by calculating the distance via a dynamic time warping (DTW) algorithm instead of a line-detection-based algorithm. The strict version of our DTW-based algorithm can output the detection candidates exactly in increasing order of their DTW distances. Our experimental evaluation showed that, whereas the strict DTW algorithm presented the detection candidates exactly in increasing order of their DTW distances, its relaxed variants execute much faster at the cost of only a slight loss in the exact ordering.

The organization of the paper is as follows. We briefly examine related work using indexing for STD in Section 2. Section 3 describes the proposed indexing method for STD in detail. In Section 4, we evaluate the method experimentally. Finally, we give our conclusions in Section 5.

## 2. Related Work

In general, STD methods first translate speech to word/subword sequences by automatic speech recognition (ASR), then search for appearances of the given query term in the word/subword sequences. Many methods for dealing with recognition error and the OOV problem have been proposed. Representing the multiple candidates from the ASR system by lattice or confusion networks [10][11][12] is one of the approaches to recognition error. Another approach is approximate matching, which allows the search results to contain several recognition errors [8][13]. To detect OOV terms without depending on the ASR vocabulary, the subword unit is commonly used [10]. These approaches focus on improving search accuracy.

On the other hand, indexing methods for spoken documents have been proposed for speeding up the detection process. In previous indexing methods for STD, inverted files [5][6][7] and suffix arrays [8] have been used. Because they involved binary indexing, which expresses only the appearance or nonappearance of items, these methods set a threshold value initially and then calculate distances to filter out implausible results, either during or after use of the indexes for searching. Setting a threshold value is a difficult problem, as the optimal threshold could depend on the quality of the spoken document, the performance of the ASR system, or the demands of the application. Depending on the threshold value, we might obtain too many detection candidates or might not get any detection results at all. Furthermore, after a first attempt at detection, if more detection results are needed, the complete detection process has to be repeated.



i
Distance plane between syllables
Syllable sequence of Query
go
N
ge
N
ze
shi
j
wa
ta
shi
wa
shi
ze
N
ge
N
go
sho
ri
ke
N
Syllable sequence of spoken document

Figure 1: STD with straight line detection

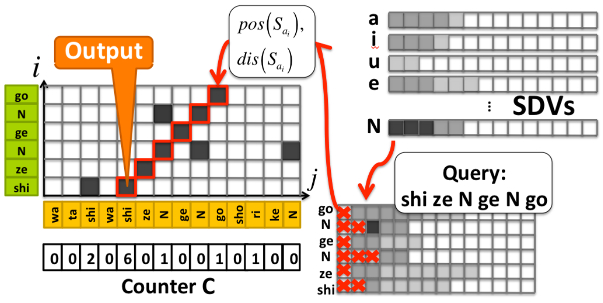


Figure 2: Detection process using a metric subspace index

Kaneko and Akiba have proposed an STD method based on line detection using metric subspace indexing [9]. This method indexes a spoken document efficiently by using information about both the distance between syllables and the utterance position of the syllable. It enables detection without setting the threshold values required by previous indexing methods. However, the line detection nature of the method suffered from degradation of its detection performance, because of its poor handling of insertion and deletion caused by speech recognition errors.

## 3. STD by Metric Subspace Indexing

In this section, we recast the work described in [9]. Although the phoneme was used as the processing unit in the previous work, we use the syllable in this version.

Consider a plane in which the $x$ and $y$ axes correspond to the syllable sequence for the spoken document obtained via ASR and the syllable sequence for the input query, respectively (see Figure 1). For each grid point on the plane, the distance between the syllable in the document at $x$ and the syllable in the query at $y$ is defined. The distance at the grid point is analogous to the pixel density at an image data point. Our proposed method consists of an indexing process and a detection process.

### 3.1. Indexing Process

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

$$T_j = D_{0,j} + D_{1,j+1} + \cdots + D_{m,j+m-1} = \sum_{i=0}^{m-1} D_{i,i+j}. \quad (1)$$

For 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. For STD, however, the distances $D_{i,j}$ can be processed beforehand, because the target spoken document is known in advance. Let $d(a, b)$ be the distance between syllables $a$ and $b$. We define $D(a)_j$ as the distance between a syllable $a$ and the syllable $b_j$ that appears at position $j$ in the target spoken document:

$$D(a)_j = d(a, b_j). \quad (2)$$

Then, for each $a$, the syllable distance vector $[D(a)_0, D(a)_1, \ldots, D(a)_j, \ldots, D(a)_{n-2}, D(a)_{n-1}]$ can be calculated in advance. When a user enters a query, we can easily construct the $xy$-plane by arranging the distance vectors along the $y$-axis according to the syllable sequence of the query, so that $D_{i,j} = D(a_i)_j$ for the query $a_0 a_1 \ldots a_{m-1}$. Here, the metric space defined between the query string and every substring in the target document is divided into metric subspaces, each of which is defined between each syllable in the query and every position in the document.

Furthermore, the syllable distance vector can be sorted in advance. We pair distance $D(a)_j$ with position $j$ and make a vector $[(D(a)_0, 0), (D(a)_1, 1), \ldots, (D(a)_j, j), \ldots, (D(a)_{n-2}, n-2), (D(a)_{n-1}, n-1)]$. We then sort this vector according to the distances (the first item of each pair). We call this vector the sorted distance vector (SDV). Let $S_a$ be the SDV for syllable $a$. We handle $S_a$ as a stack, where the stack top is the leftmost element of the vector. Using $S_a (a \in A)$ for the set of syllables $A$ as the index, we obtain a fast STD algorithm that can output the detection results in increasing order of distance, as described in the next subsection.

Note that we can easily introduce any "sausage-like" representation of multiple recognition candidates, such as a confusion network, into the index, as its element $D(a)_j$ can be defined as a distance that depends on the position $j$, instead of the syllable $b_j$, in the spoken document.

### 3.2. Detection Process

Let $s_a = (dis(s_a), pos(s_a))$ be the top element of SDV $S_a$, where $dis(s_a)$ and $pos(s_a)$ are the distance and the position recorded in the paired element $s_a$, respectively. The detection process is as follows (see Figure 2).

1. According to the query syllable sequence $a_0 a_1 \ldots a_i \ldots a_{m-2} a_{m-1}$, prepare the SDVs $S_{a_0} S_{a_1} \ldots S_{a_i} \ldots S_{a_{m-2}} S_{a_{m-1}}$. Initialize the counter (ballot box) $C[j] = 0 (0 \leq j < n)$ and the candidate set $U = \{\}$.
2. Pop the top element $s_{a_i}$ of $S_{a_i}$, which has the minimum distance $min_i\{dis(s_{a_i})\}$, from the set that comprises the top elements $s_{a_0} s_{a_1} \ldots s_{a_i} \ldots s_{a_{m-2}} s_{a_{m-1}}$ of the SDVs $S_{a_0} S_{a_1} \ldots S_{a_i} \ldots S_{a_{m-2}} S_{a_{m-1}}$. Let $j = pos(s_{a_i}) - i$ and add 1 to $C[j]$ (voting).
3. If $C[j] = k$, then add the position $j$ to the set $U$.
4. Output the subset $V = \{j | T_j < \sum_{i=0}^{m-1} dis(s_{a_i})\}$ of $U$ in order of their cumulative distances $T_j$. Let $U \leftarrow U - V$.
5. Repeat Steps 2, 3 and 4 until the required condition is satisfied.

Note that this algorithm does not use a threshold for distances; instead, it outputs the detection results approximately in order of small to large distances. Note also that the candidate set $U$ can be implemented by using any efficient data structure for the priority queue.

In particular, when we set $k = 1$ at Step 3, we define a strict algorithm that outputs results ordered exactly by their distances. This is proved by the following two lemmas.

**Lemma 1** Let $T_Q$ be the sum of the current top elements of the SDVs $S_{a_0}S_{a_1}\dots S_{a_i}\dots S_{a_{m-2}}S_{a_{m-1}}$ at some point during the runtime of the algorithm, i.e., $T_Q = \sum_{i=0}^{m-1} dis(s_{a_i})$. If there is no vote at position $j$, i.e., $C[j] = 0$, the final cumulative distance $T_j$ is not less than $T_Q$, i.e., $T_j \geq T_Q$.

**(Proof)** Because each SDV is sorted by distance, for all positions $j$ at which there has not yet been a vote, the distance for each syllable $D(a_i)_{i+j}$ is not less than the distance for the top elements of the current SDV of $a_i$, i.e.:

$$D(a_i)_{i+j} \geq dis(s_{a_i}). \tag{3}$$

Therefore:

$$T_j = \sum_{i=0}^{m-1} D(a_i)_{i+j} \geq \sum_{i=0}^{m-1} dis(s_{a_i}) = T_Q. \tag{4}$$

**Q.E.D**.

**Lemma 2** At each position $j$ for which $T_j < T_Q$ stands, there has been at least one vote at that position $j$.

**(Proof)** Lemma 2 is the contrapositive of Lemma 1. **Q.E.D.**

Therefore, the candidate $V$ at Step 4 can be safely output, because it can be guaranteed that no better candidate exists at any position where there has not yet been a vote.

We could use various 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.”

### 3.3. Dealing with Insertion and Deletion Errors

Because an actual ASR result contains recognition errors, the detection of a term does not always lie on the line. To deal with insertions and deletions caused by recognition errors, we introduce an alternative distance measure instead of Equation 2, which incorporates syllable neighbors as follows:

$$D(a)_j = \min \begin{cases} d(a, b_{j-1}) + \gamma \\ d(a, b_j) \\ d(a, b_{j+1}) + \gamma \end{cases} \tag{5}$$

Here, $a$ is a phoneme in a query, $b_j$ is the syllable at position $j$ in the spoken document, $d(a, b)$ is the distance between syllables $a$ and $b$, and $\gamma$ is introduced to penalize the use of a neighboring syllable.

Note that we are only replacing the distance measure here, with the algorithm itself not being revised at all.

### 3.4. DTW Distance-Ordered Detection

To further improve the detection performance, we replace the cumulative distance computed by Equation 1 with the DTW distance, with the aim of achieving robust detection of recognition errors. For the cumulative distance calculation in Step 4 in the algorithm described in Section 3.2, we use the DTW distance computed by Equations 6 and 7 instead of Equation 1.

$$\begin{aligned}
&W_{0,j'} = D(a_0)_{j'} && (j - m \leq j' < j + 2m) \\
&W_{i',j-I} = D(a_{i'})_{j-I} + W_{i'-1,j-I} && (0 < i' < m) \\
&W_{i',j'} = D(a_i)_j + \min\{W_{i',j'-1}, W_{i'-1,j'-1}, W_{i'-1,j'}\} \\
&\qquad (0 < i' < m,\ j - m \leq j' < j + 2m)
\end{aligned} \tag{6}$$

Here, we select the minimum cumulative distance by Equation 7 after computing the cumulative distances defined by Equation 6.

$$W_j = \min_{j-m \leq j' < j+2m} W_{m-1,j'} \tag{7}$$

This change in computing the cumulative distance is introduced by recasting Step 4 of the detection process described in Section 3.2 as:

4. Compute the DTW cumulative distance within a range of $[\ j - m,\ j + 2m\ )$ via Equation 6, select the minimum cumulative distance $W_j$ via Equation 7, and add a pair $(j, W_j)$ to the candidate set U.

For this modified algorithm, if we set k = 1 at Step 3 and assume that the maximum DTW path is shorter than $2m$, we obtain the strict DTW algorithm, which outputs the results in exact order of their distance as computed by DTW.

## 4. Experiments

### 4.1. Test Collection

We used the Japanese STD test collection [3] for our evaluation. The target documents are 177 lectures selected from the Corpus of Spontaneous Japanese (CSJ) [14] and amount to 44 hours of talk. The collection includes 50 query terms for the target documents.

The automatic transcription obtained by using syllable-based speech recognition is used for the textual representation of the target documents. The acoustic model and the language model of the recognizer were trained in open conditions by using the target documents themselves: the target documents were divided in two and the first half was used for training the models to recognize the latter half, and vice versa. The unit of the acoustic model is the Japanese phoneme. Japanese syllables are defined in the recognition dictionary and the syllable-trigram language model was trained by using the training data.

Each lecture in the CSJ is segmented by pauses that are no shorter than 200 ms. These segments are called inter-pausal units (IPUs). In this test collection, the IPUs are used as the basic unit for searching. Recall and precision are used as the evaluation metrics, and are calculated in terms of the correct IPUs and the detected IPUs.

### 4.2. Processing Units and Distance Measures

In our previous work [9], the phoneme was used as the processing unit, and the Hamming distance between the distinctive phonetic features of phonemes was used as the distance measure. In this work, however, the syllable is used as the processing unit, and the Bhattacharyya distance between acoustic models is used as the distance measure. The Bhattacharyya distance measures the dissimilarity between two probability distributions. In this work, our acoustic model uses a hidden Markov model (HMM) for the syllables. The distance between two HMMs $a$ and $b$ is defined as follows, based on the Bhattacharyya distance for Gaussian mixture models:

$$d(a, b) = \frac{1}{M} \sum_{\alpha=1}^{M} \min_{\beta,\gamma} BD\{P_a(S_a^\alpha, \beta), P_a(S_a^\alpha, \gamma)\} \tag{8}$$

$$\begin{aligned}
BD(P_a, P_b) &= \frac{1}{8}(\mu_a - \mu_b)\{\frac{\Sigma_a + \Sigma_b}{2}\}^{-1} \\
&\quad \cdot(\mu_a - \mu_b)^t + \frac{1}{2} log(\frac{|(\Sigma_a + \Sigma_b)/2|}{|\Sigma_a|^{\frac{1}{2}}|\Sigma_b|^{\frac{1}{2}}}),
\end{aligned} \tag{9}$$

where $S_a^\alpha$ is the $\alpha$th state of the HMM for syllable $a$, $P(S_a^\alpha, \beta)$ is the $\beta$th Gaussian distribution of $S_a^\alpha$, $\mu_a$ is the mean vector of $a$, $\Sigma_a$ is the covariance matrix of $a$, $M$ is the number of states, and $BD(P_a, P_b)$ is the Bhattacharyya distance between two Gaussians.

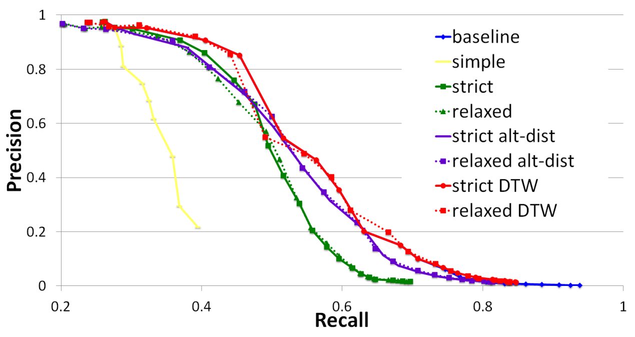


Figure 3: The recall–precision curve for varying cutoff thresholds

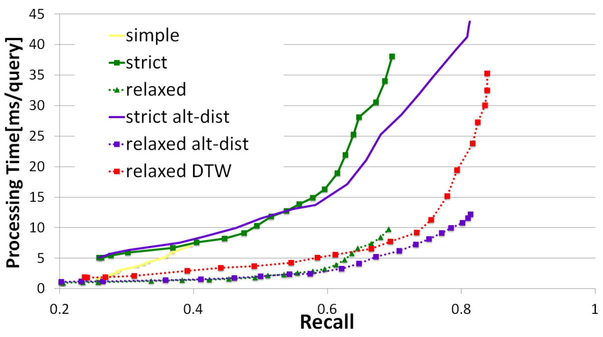


Figure 4: The relation between recall and processing time

Table 1: F1 and processing time

| Method | $k$ | F1 | Processing Time [ms/query] |
|---|---|---|---|
| baseline | - | 0.592 | 233 |
| simple | - | 0.443 | 3.6 |
| strict | 1 | 0.562 | 8.3 |
| relaxed | $\lceil 0.4m \rceil$ | 0.545 | 1.5 |
| strict alt-dist | 1 | 0.558 | 10.0 |
| relaxed alt-dist | $\lceil 0.5m \rceil$ | 0.561 | 1.7 |
| strict DTW | 1 | 0.592 | 82.3 |
| relaxed DTW | $\lceil 0.3m \rceil$ | **0.582** | **3.4** |

### 4.3. Compared Methods

As our **baseline** method, we used the DTW algorithm that traverses all the spoken documents, and is defined as follows:

$$\begin{aligned} T_{0,j} &= D(a_0)_j && (0 \leq j < n) \\ T_{i,0} &= D(a_i)_0 + T_{i-1,0} && (0 < i < m) \\ T_{i,j} &= D(a_i)_j + \min\{T_{i-1,j}, T_{i-1,j-1}, T_{i,j-1}\}, && (10) \end{aligned}$$

where $T_{m-1,j}$ is the cumulative distance for DTW. The detection was performed by varying the threshold values for $T_{m-1,j}$. We compared the following our methods with the **baseline** method:

**simple** Simple algorithm described in [9].

**strict** Strict algorithm based on line detection, described in Section 3.2 ($k = 1$).

**relaxed** Relaxed version of **strict** ($k = \lceil 0.4m \rceil$).

**strict alt-dist** Strict algorithm based on line detection, with the alternative distance measure described in Section 3.3 ($k = 1$).

**relaxed alt-dist** Relaxed version of **strict alt-dist** ($k = \lceil 0.5m \rceil$).

**strict DTW** Strict algorithm based on the DTW described in Section 3.4 ($k = 1$).

**relaxed DTW** Relaxed version of **strict DTW** ($k = \lceil 0.3m \rceil$).

Here, the parameter $k$ for each of the relaxed versions was selected to balance the detection performance against the efficiency of the method.

### 4.4. Results

Figure 3 shows the recall–precision curves for the compared methods, obtained by varying the cutoff threshold. The maximum F-measures on these curves (referred to as F1) and their processing times are also given in Table 1.

Firstly, the **strict** method we implemented in this study greatly outperforms **simple**. This indicates that the exact ordering of the results by their distances does improve the STD performance. Moreover, using the alternative distance (**strict alt-dist** and **relaxed alt-dist**) gives an effective increase to the accuracy in the high-recall region.

Secondly, **strict DTW**, which sets $k$ to 1, outputs the results in the exact order of their DTW distances. Its recall–precision curve therefore almost completely agrees with that for**baseline**. (To be precise, the curves in the high-recall region diverge slightly, because there are DTW paths that violate the path assumption described in Section 3.4, namely that the maximum DTW path is to be shorter than $2m$.) As shown in Table 1, **strict DTW** executes about three times faster than **baseline**, while achieving the same detection performance.

Furthermore, by adjusting the value of $k$ to relax the constraint on exact ordering, we can achieve faster detection while only slightly degrading the detection performance. As shown in Table 1, **relaxed DTW** achieves detection results about 70 times faster than **baseline**, while its F1 drops by only $1.7\%$. Compared with **relaxed** and **relaxed alt-dist**, its detection performance is improved particularly in the high-recall region, which is considered to involve many recognition errors.

Finally, we investigate the efficiency of detection. Figure 4 shows the relation between recall and processing time per query for each method. Because **baseline** requires about 230 ms for detection per query, the methods that use our indexing are approximately 10 to 100 times faster than **baseline**. In particular, **relaxed** has greatly improved detection efficiency in comparison to **simple**. Among the relaxed methods, **relaxed alt-dist**) has increased efficiency in the high-recall region (over $0.6$). Because **relaxed DTW** involves a high computational cost for the DTW calculation, its efficiency is not superior to the non-DTW algorithms.

Note again that all these results for the proposed distance-ordered method can be obtained incrementally within a single run of the detection process, along with the curves of Figures 3 and 4, because results are output in distance order without any predefined threshold interrupting the process.

## 5. Conclusion

In this work, we propose a novel indexing and search method for STD that requires no cutoff threshold in the detection process but outputs detection results in increasing order of their DTW distances for a given query term. Our experimental evaluation showed that, while the strict algorithm of our method presented detection results in exact increasing order of their DTW distances, its relaxed variants can run 70 times faster than the baseline method while dropping only $1.7\%$ in F1.

In future work, we would like to investigate those applications that take can advantage of the distinctive properties of the proposed distance-ordered STD.

## 6. References

[1] J. S. Garofolo, C. G. P. Auzanne, and E. M. Voorhees, "The TREC spoken document retrieval track: A success story," in *Proceedings of TREC-9*, 1999, pp. 107–129.

[2] National Institute of Standards and Technology, "Spoken term detection evaluation portal," "`http://www.nist.gov/speech/tests/std/`".

[3] Y. Itoh, H. Nishizaki, X. Hu, H. Nanjo, T. Akiba, T. Kawahara, S. Nakagawa, T. Matsui, Y. Yamashita, and K. Aikawa, "Constructing japanese test collections for spoken term detection," in *Proceedings of International Conference on Speech Communication and Technology*, 2010, pp. 677–680.

[4] T. Akiba, H. Nishizaki, K. Aikawa, T. Kawahara, and T. Matsui, "Overview of the IR for spoken documents task in NTCIR-9 workshop," in *Proceedings of The Ninth NTCIR Workshop Meeting*, 2011, pp. 223–235.

[5] K. Iwami and S. Nakagawa, "High speed spoken term detection by combination of n-gram array of a syllable lattice and LVCSR result for NTCIR-SpokenDoc," in *Proceedings of The Ninth NTCIR Workshop Meeting*, 2011.

[6] C. Chelba and A. Acero, "Position specific posterior lattices for indexing speech," in *Proceedings of Annual Meeting of the Association for Computational Linguistics*, 2005, pp. 443–450.

[7] Z.-Y. Zhou, P. Yu, C. Chelba, and F. Seide, "Towards spoken-document retrieval for the internet: Lattice indexing for large-scale web-search architectures," in *Proceedings of Human Language Technology Conference*, 2006, pp. 415–422.

[8] K. Katsurada, S. Teshima, and T. Nitta, "Fast keyword detection using suffix array," in *Proceedings of International Conference on Speech Communication and Technology*, 2009.

[9] T. Kaneko and T. Akiba, "Metric subspace indexing for fast spoken term detection," in *Proceedings of International Conference on Speech Communication and Technology*, 2010, pp. 689–692.

[10] Y. Pan, H. Chang, B. Chen, and L. Lee, "Subword-based position specific posterior lattices (S-PSPL) for indexing speech information," in *Proceedings of International Conference on Speech Communication and Technology*, 2007, pp. 318–321.

[11] M. Saraclar and R. Sproat, "Lattice-based search for spoken utterance retrieval," in *Proceedings of Human Language Technology Conference*, 2004.

[12] T. Hori, L. Hetherington, T. J. Hazen, and J. R. Glass, "Open-vocabulary spoken utterance retrieval using confusion networks," in *Proceedings of International Conference on Acoustic, Speech, and Signal Processing*, 2007, pp. 73–76.

[13] A. Jansen, K. Church, and H. Hermansky, "Towards spoken term discovery at scale with zero resources," in *Proceedings of International Conference on Speech Communication and Technology*, 2010, pp. 1676–1679.

[14] K. Maekawa, H. Koiso, S. Furui, and H. Isahara, "Spontaneous speech corpus of Japanese," in *Proceedings of International Conference on Language Resources and Evaluation*, 2000, pp. 947–952.