DTW-Distance-Ordered Spoken Term Detection and STD-based Spoken Content Retrieval: Experiments at NTCIR-10 SpokenDoc-2

Tomoyosi Akiba, Tomoko Takigami, Teppei Ohno, and Kenta Kase
Toyohashi University of Technology
{akiba | ohnoteppei | kase}@nlp.cs.tut.ac.jp

ABSTRACT

In this paper, we report our experiments at NTCIR-10 SpokenDoc-2 task. We participated both the STD and SCR subtasks of SpokenDoc. For STD subtask, we applied novel indexing method, called metric subspace indexing, previously proposed by us. One of the distinctive advantages of the method was that it could output the detection results in increasing order of distance without using any predefined threshold for the distance. In this work, we extended the algorithm to work with the Dynamic Time Warping (DTW) distance. We also participated in the newly introduced iSTD task by using the proposed STD method. For SCR subtask, two kinds of approaches were applied. For the lecture retrieval task, STD-based SCR method was applied. It used STD for detecting the terms in the query from the spoken documents and their results were used to calculate the similarities between the query and documents according to the vector space model. For the passage retrieval task, we used IR models based on language modeling and focused on determining the optimal range of a relevant passage as the retrieval result.

Team Name

AKBL

Subtasks

Spoken Term Detection (Moderate-size Task, Large-size Task)
Spoken Content Retrieval (Lecture Retrieval Task, Passage Retrieval Task)

Keywords

metric subspace indexing, STD-based SCR

1. INTRODUCTION

In this paper, we report our experiments at NTCIR-10 SpokenDoc-2 task[3], where our previously proposed methods for STD and SCR are applied. We submitted 23 runs in total, which were three runs for the STD moderate-size task, three runs for the STD large-size task, three runs for the iSTD task, eight runs for the SCR lecture retrieval task, and six runs for the passage retrieval task.

For STD task, we applied novel indexing method, called metric subspace indexing, previously proposed by us [11]. The proposed method can be considered as using metric space indexing for approximate string matching problem, where the distance is defined between an indexing unit, e.g. a phoneme or a syllable, 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. 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. In this work, we extended the algorithm to work with the Dynamic Time Warping (DTW) distance, which results in DTW-distance-ordered detection. We also participated in the newly introduced iSTD task by using the proposed STD method.

For the SCR subtask, two kinds of approaches, the STD-based approach using vector space model and word-based approach using IR model based on language modeling, were applied to the lecture retrieval task and the passage retrieval task, respectively.

The first approach, STD-based SCR method, is based on the syllable-based speech recognition results. In the first step, a STD method is applied to the spoken documents, where each term in the given query topic is searched against the syllable sequence obtained by the speech recognition. From the detection results, we can obtain the statistics of the term frequencies for each document, to which we can apply any conventional document retrieval method. The advantage of this method is that it is not affected by the OOV terms in the query topics.

On the other hand, the second approach is rather conventional one, which uses the word-based speech recognition results, but use relevance model, which was proposed in the context of language modeling approach for IR. Furthermore, we focused on the problem of determining the range of a relevant passage that should be outputted as the retrieval results in the SpokenDoc-2 passage retrieval task.

The remainder of this paper is organized as follows. Section 2 describes our STD method used for the STD subtask and its experimental evaluation. Section 3 and 4 describes our two approaches for the SCR subtask and their evaluation. In Section 5, we conclude our experiments at SpokenDoc.

2. DTW-DISTANCE-ORDERED SPOKEN TERM DETECTION

General STD method translates speech to word/subword sequences by automatic speech recognition (ASR) at first, then searches appearances of the given query term from the word/subword sequences. Many methods dealing recognition error and Out of Vocabulary (OOV) problem have been proposed. Utilizing multiple candidates of ASR system represented by lattice or confusion network [14][16][7] is one of the approaches for recognition error. Other approach for recognition error is approximate matching which admits containing several recognition errors [13][9]. To detect OOV terms without depending on ASR vocabulary, subword unit is commonly used [14]. These approaches focused on improving search accuracy.

On the other hand, indexing method for spoken documents has been proposed for speeding up of detection process. As previous indexing method for STD, inverted files [8][4][23] and suffix array [13] have been applied, but these indexes use binary indexing, which expresses only appearance or nonappearance. Therefore, these methods set a

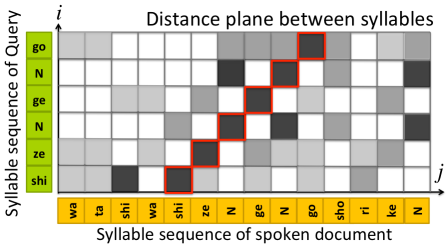

Figure 1: STD as straight line detection

threshold value preliminarily and calculated distances to filter out implausible results either during or after using the indexes for searching. Setting a threshold value is difficult problem since the optimal one depends on quality of spoken document, performance of ASR system, demand of application, etc. Depending on the threshold value, we may obtain too many detection candidates or may not get the detection results at all. Furthermore, after first detection, if more detection results are needed, the detection process has to be conducted all over again.

Kaneko and Akiba has proposed a STD method based on line detection using metric subspace indexing [11]. This method indexes a spoken document efficiently using the information of both the distance between syllables and the utterance position of the syllable and enables detection without setting any threshold values. However, the line-detection-based nature of the method suffered from the degradation of the detection performance as it failed to handle well the insertion and deletion on the spoken documents caused by speech recognition errors.

2.1 STD by Metric Subspace Indexing

In this section, we recast the work described in [11, 12].

Consider a plane in which the x and y axes correspond to the syllable sequence of the spoken document obtained by using ASR and the syllable sequence of 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 a grid point is analogous to the pixel density at an image data point. Our proposed method is divided into an indexing process and a detection process.

2.1.1 *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 follows:

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

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 query is supplied, 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.

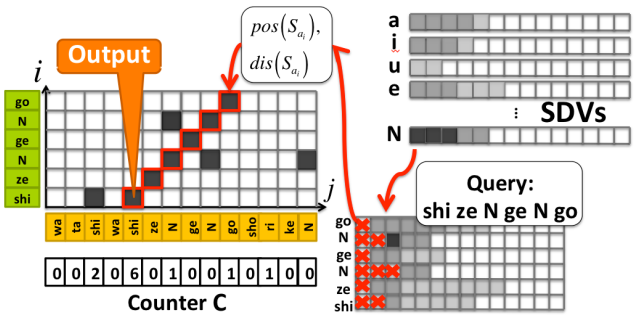

Figure 2: Detection process using metric subspace index

Furthermore, the syllable distance vector can be sorted in advance. We pair distance $D(a)_j$ with position j and make a vector $[(D(a)_0, 0), (D(a)_1, 1), \ldots, (D(a)_j, j), \ldots, (D(a)_{n-2}, n-2), (D(a)_{n-1}, n-1)]$. Then we 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 of 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 can obtain a fast STD algorithm that can output the detection results in increasing order of distance as described in the next section.

2.1.2 *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 (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 the order of their cumulative distance 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 the order of smallest to largest 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 the strict algorithm that outputs the results exactly ordered by their distances. The proof is found in [12].

2.2 Dealing with Insertion and Deletion Errors

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

$$D(a)_j = \min \begin{cases} d(a, b_{j-1}) + \gamma \\ d(a, b_j) \\ d(a, b_{j+1}) + \gamma \end{cases} \quad (3)$$

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 γ is introduced to penalize the use of the neighbor syllable.

Note that we only replace the distance measure here and the algorithm is not revised at all.

2.3 DTW Distance-Ordered Detection

In order to further improve the detection performance, we replace the cumulative distance computed by equation 1 with DTW distance in order to achieve robust detection for recognition errors. For the cumulative distance calculation in the Step 4 in the algorithm described in Section 2.1.2, we use DTW distance computed by Equation 4 and 5 instead of Equation 1.

$$\begin{aligned} W_{0,j'} &= D(a_0)_{j'} && (j - I \le j' < j + 2I) \\ W_{i',j-I} &= D(a_{i'})_{j-I} + W_{i'-1,j-I} && (0 < i' < I) \\ W_{i',j'} &= D(a_i)_j + \min\{W_{i',j'-1}, W_{i'-1,j'-1}, W_{i'-1,j'}\} \\ & && (0 < i' < I,\ j - I \le j' < j + 2I) \end{aligned} \quad (4)$$

We select the minimum cumulative distance by Equation 5 after computing the cumulative distances defined by Equation 4.

$$W_j = \min_{j-I \le j' < j+2I} W_{I-1,j'} \quad (5)$$

The change of computing the cumulative distance above is introduced by recasting Step 4 in Section 2.1.2 as follows.

4. Compute the DTW cumulative distance within a range of $[\ j - I,\ j + 2I\ )$ by Equation 4, select the minimum cumulative distance W_j by Equation 5, add a pair (j, W_j) to the candidate set U.

For this modified algorithm, when we set k = 1 at Step 3 and assume that the maximum DTW path is shorter than $2I$, we can obtain the strict DTW algorithm, which outputs the results exactly ordered by their distance computed by DTW.

2.4 Experiments

In this subsection, the evaluation results at NTCIR-10 SpokenDoc2 STD and iSTD subtask formal run are described. We submitted three runs for each of the large-size STD task, the moderate-size STD task, and the iSTD task. These runs are adjusted so as to balance the detection performance and the efficiency. In addition to the performances of these runs, we also show that of the runs by our strict algorithm, which output the detection results exactly in increasing order of the distances.

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. We use syllable HMM for our acoustic model.

2.4.1 Compared Methods

We compared the following methods. All methods utilize our proposed indexing described in subsection 2.1.1 and follow subsection 2.1.2 basically as their detection process. Therefore the difference among the following three methods is how to compute the cumulative distance between a query and a candidate.

dist-DTW (akbl-1) A algorithm based on the DTW described in Section 2.3.

altdist-LD (akbl-2) A algorithm based on the Line-Detection dealing with insertion and deletion errors described in Section 2.2.

dist-LD (akbl-3) A simplest algorithm of the three method. It is based on the Line-Detection described in Section 2.1.2.

where the parameter k for each method was selected so as to balance the detection performance and the efficiency using dryrun query set.

Furthermore, in addition to our submitted runs, we investigated the performances of the strict version of the runs that outputted the results exactly ordered by their distances. The exactly ordered results can be obtained by setting the parameter k to 1 as described in subsection 2.1.2 and subsection 2.3. We compared the following methods.

strict-dist-DTW Strict version of **dist-DTW** ($k = 1$).

strict-altdist-LD Strict version of **altdist-LD** ($k = 1$).

strict-dist-LD Strict version of **dist-LD** ($k = 1$).

For iSTD task, we introduced simple scoring to the results of STD task. Using the normalized cumulative distance d_c of the detection candidate $c \in D_q$ for a query q, we defined our iSTD score s_q as

$$s_q = 1 - \min_{c \in D_q} d_c \quad (6)$$

This score can be interpreted as the value representing "How much does the query be likely not to exist in the whole spoken document?". Then, the query terms were sorted in descending order of s_q. This scoring can be simply implemented as the post-precessing of the STD task.

2.4.2 Result on the STD task

Table 1, Figure 3 and Figure 4 show our results on the large-size STD task, while Table 2, Figure 5 and Figure 6 are on the moderate-size STD task.

We include the result of the baseline BL-1 provided by the task organizer, as it and our runs are commonly built on the REF-SYLLABLE-MATCHED transcription. The comparison between BL-1 and strict-dist-DTW simply shows the difference between the underlying DTW performances; i.e., the choice of the processing unit (BL-1 uses phoneme, while ours uses syllable), the distance metric between them (Edit-distance vs. Battacharyya-distance), the detection threshold, the path restriction adopted by the DTW, etc.

Among our proposed methods, dist-DTW (akbl-1) outperforms altdist-LD (akbl-2) and dist-LD (akbl-3). It is mainly because of the improvement at high-recall region as shown in Figure 3 and Figure 4. It can be explained that the DTW-based method (dist-DTW) can better handle the insertions and deletions, which are expected to occur more at high-recall region, than the line-detection-based methods (altdist-LD and dist-LD) do.

The comparison between the relaxed methods (dist-DTW, altdist-LD, and dist-LD) and their strict versions (strict-dist-DTW, strict-altdist-LD, and strict-dist-LD) reveals that the relaxed methods can achieve much faster detection with slightly degrading the detection performance. Even for the large-size task, it takes only about 0.06-0.07 second per query. The search time for the moderate-size task reduces to 1/16-1/27, along with the reduction of the target data size.

2.4.3 Result on the iSTD task

Table 3 shows our result on the iSTD task. The performances of our submitted three runs are similar to the baseline BL-1, which is built on the same REF-SYLLABLE-MATCHED transcription. It seems because the scoring methods for the iSTD task are similar between that of the baseline and ours.

Table 1: STD performances on the large-size task

Method (run)	micro ave.		macro ave.			index	search
	max F. [%]	spec F. [%]	max F. [%]	spec F. [%]	MAP	size [MB]	speed [s]
(BL-1)	42.32	40.71	43.91	36.70	0.500	58	560
dist-DTW (akbl-1)	39.74	33.76	39.09	37.34	0.490	17250	0.0633
altdist-LD (akbl-2)	38.11	27.56	38.99	38.53	0.452	18210	0.0719
dist-LD (akbl-3)	38.12	26.88	35.35	35.54	0.390	17250	0.0587
strict-dist-DTW	39.72	-	38.99	-	0.491	17250	7.31
strict-altdist-LD	38.11	-	38.99	-	0.451	18210	0.760
strict-dist-LD	38.12	-	35.34	-	0.389	17250	0.527

Table 2: STD performances on the moderate-size task

Method (run)	micro ave.		macro ave.			index	search
	max F. [%]	spec F. [%]	max F. [%]	spec F. [%]	MAP	size [MB]	speed [s]
(BL-1)	25.08	24.70	25.72	20.07	0.317	3.3	30.8
dist-DTW (akbl-1)	20.71	13.48	25.79	21.29	0.343	1120	0.00407
altdist-LD (akbl-2)	20.00	13.50	22.61	18.26	0.293	1120	0.00324
dist-LD (akbl-3)	19.95	13.40	21.24	18.07	0.244	1120	0.00212
strict-dist-DTW	20.71	-	25.79	-	0.348	1120	0.356
strict-altdist-LD	20.00	-	22.61	-	0.293	1120	0.0411
strict-dist-LD	19.95	-	21.24	-	0.244	1120	0.0353

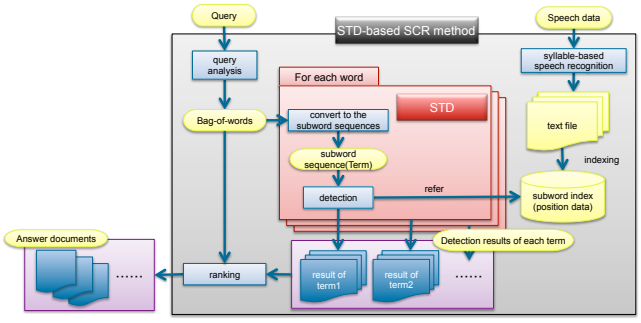

Figure 7: *STD-SCR system*

3. STD-BASED SPOKEN CONTENT RETRIEVAL

The conventional SCR methods use word-based speech recognition to obtain the transcription of the spoken documents, and then text-based document retrieval is applied to the transcription. However, the OOV words from the word-based speech recognition and misrecognized words can never be used as the clues for the document retrieval, which results in the degradation of the retrieval performance. To deal with these problems, both document and query expansion methods have been proposed [2, 19, 18]. These methods are not using OOV words or speech segments of recognition errors in the document, but these extensions of related words as clue words that are correctly recognized for order to deal with recognition errors. On the other hands, subword n-gram based SCR methods have been proposed [5, 15] . While the conventional SCR used words as a feature of a document vector, these methods used subword n-gram. However, subword n-grams are a automatically extracted without using the language knowledge, the effectiveness of such clues is limited compared to the word.

3.1 STD-based SCR

To deal with the problem of recognition errors and OOV words, we propose the STD-based approach for SCR. Figure 7 shows the configuration of our STD-based system.

First, we create automatic transcripts by applying subword-based recognition to speech data. Given query topic, the keywords are extracted from the query topics and are converted to subword sequences. In this paper, we use nouns as the keywords and syllables as the subword unit. Then, each syllable sequence is used as a search key against the syllable-based transcription of the spoken documents. From the STD results, we can obtain the keyword frequency for each document in the collection. We repeat the process for all keywords, and then we can obtain the vector of the keyword frequencies for each document. Finally, the vectors are compared with the query vector to select the retrieval results, which is equivalent to applying the conventional SCR system.

Note that the proposed method is different from the previous works that using the subword n-grams as clues [5, 15]. While they also use the subword-based recognition results, their document and query vectors are created based on the subword n-grams.

Our proposed method resembles the early approach in SCR [10, 21], which applies word-spotting for spoken documents instead of LVCSR. As the word-spotting had to be applied after a query topic is given, it was not a tractable approach for targeting a large amount of spoken documents. Thanks to the recent development of the fast STD methods, the approach now become tractable even for targeting a large amount of spoken documents, which is investigated in this work.

3.2 Retrieval Model for STD-SCR

As a method to calculate the similarity between each document and a given query topic, vector space model has been widely used in document retrieval. Unlike text retrieval, as the occurrence frequency of each word in SCR is uncertain caused by recognition errors, it can not be used as a reliable clue for document retrieval. In this work, we propose a novel retrieval model that extends the vector space model robust for uncertain clues.

There are two types of errors brought in the document vector obtained by using automatic speech recognition. One is so called *false negative*, which is an error of the word that appears in the document but not in the recognition result. The other is so called *false positive*, which is an error of the word that does not appear in the document but in the recognition result. Previous works on SCR have mainly focused on the false negative. For example, query expansion [2, 19] and document expansion [18] have been well studied in the context of SCR, which aim to compensate for the false negative by introducing the other words than the misrecognized words.

On the other hand, there have been few works that aim to compensate for the false positive. Actually, the false positive appears much less than the false negative within the conventional SCR based on the LVCSR result, so it does not affect much. However, our proposed STD-SCR tends to have many false positive as it is based on the STD detection results. In this work, we also propose the novel retrieval model designed for our proposed STD-SCR.

3.2.1 *Vector space model using word combination feature*

Two keywords contained in a query topic are related to each other, therefore, we can be considered these words are

Table 3: iSTD performances

Method (run)	Rank 100			Specified				Maximum			
	R [%]	P [%]	F [%]	R [%]	P [%]	F [%]	rank	R [%]	P [%]	F [%]	rank
(BL-1)	73.00	73.00	73.00	81.00	65.85	72.65	123	73.00	76.04	74.49	96
dist-DTW (akbl-1)	72.00	72.00	72.00	89.00	66.92	76.39	133	95.00	65.97	77.87	144
altdist-LD (akbl-2)	67.00	67.00	67.00	87.00	65.41	74.68	133	95.00	63.33	76.00	150
dist-LD (akbl-3)	68.00	68.00	68.00	90.00	65.69	75.95	137	94.00	65.28	77.05	144

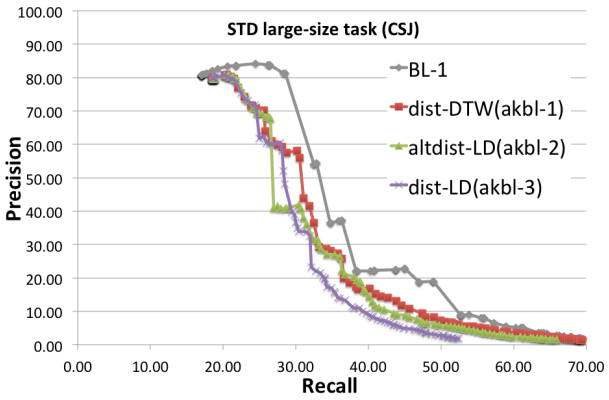

Figure 3: The recall–precision curves for the result on the large-size task (micro average)

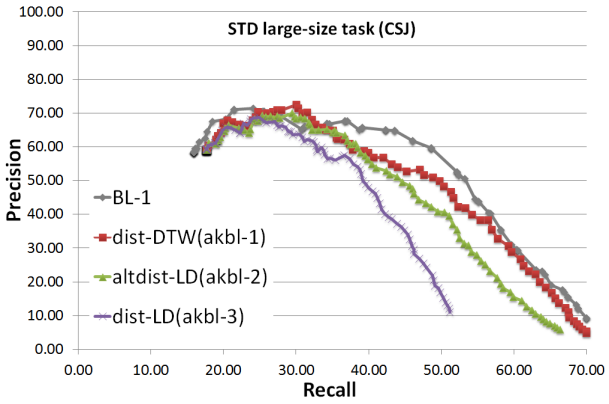

Figure 4: The recall–precision curves for the result on the large-size task (macro average)

	$v(d)$			$v_c(d)$		
	w_1	w_2	w_3	w_1w_2	w_1w_3	w_2w_3
d_1	3	0	1	0	1	0
d_2	1	2	2	1	1	1
d_3	0	0	1	0	0	0

Figure 8: *Example of document vector by extended vector space model*

more likely to appear at the same time in the document. We attempt to make use of word co-occurrence as an additional feature that is used in computing the similarity between the query and each document.

For document d, document vector $v(d)$ is computed as follow,

$$v(d) \quad = \quad [tf(w_1, d), tf(w_2, d), \dots], \tag{7}$$

where $tf(w, d)$ is frequency of word w in document d. Additionally, we also calculate co-occurrence information vector $v_c(d)$ as follows.

$$v_c(d) \quad = \quad [\delta(w_1, w_2, d), \delta(w_1, w_3, d), \dots, \delta(w_i, w_j, d), \dots] \tag{8}$$

$$\delta(w_i, w_j, d) \quad = \quad \begin{cases} 1 & (w_i \in d \ \text{ and } \ w_j \in d) \\ 0 & (\text{otherwise}) \end{cases}, \tag{9}$$

The size of the co-occurrence vector $v_c(d)$ is $|v(d)|^2 - |v(d)|$. Figure 8 shows example of document vector for query $q = w_1, w_2, w_3$.

Given query topic q, firstly, we obtain document vector $v(d)$ using word frequency that contain query topic. Than, co-occurrence vector $v_c(d)$ is computed based on $v(d)$, and extended vector $v_e(d) = [v(d), v_c(d)]$ is obtained by concatenating the $v(d)$ and $v_c(d)$. Similarly, the extended query vector $v_e(q)$ is also calculated. Finally, the similarity between $v_e(d)$ and $v_e(q)$ is calculated as same as the vector space model with TF-IDF term weighting.

3.3 Combination of CSCR and STD-SCR

The proposed method will be effective for the query including the words that are OOV or misrecognized in the spoken documents, while the conventional SCR will be effective for the query that consists of the words recognized correctly in the spoken documents. It is expected that the two systems can complement each other, so it is worth investigating the hybrid system of them.

Not only simply combining the both systems, we tried to further boost the performance by making a distinction between OOV and IV words in a given query topic. Firstly, we divide the words in a query topic q into the OOV words q_{OOV} and IV words q_{IV} by consulting the recognition dictionary of the LVCSR system used in the conventional SCR system. Then, we combine the two systems as follows.

$$\begin{aligned} sim(q, d) \quad = \quad & (1 - \alpha)\, sim_{cSCR}\,(q, d) \\ & + \alpha\{(1 - \beta)\, sim_{STD-SCR}\,(q_{IV}, d) \\ & + \beta sim_{STD-SCR}\,(q_{OOV}, d)\}, \end{aligned} \tag{10}$$

where, α and β are weighting coefficients of the linear combination, sim_{cSCR} and $sim_{STD-SCR}$ are relevance scores calculated in the conventional SCR system and the proposed STD-SCR system, respectively. Using higher α means that we prefer the STD-SCR system to the conventional SCR system. Note that $\alpha = 0$ ($\alpha = 1$) corresponds to just using only the conventional SCR (STD-SCR) system. On the other hand, using higher β means that OOV words are taken more importance than IV words when using the STD-SCR system. Note that $\beta = 0.5$ means that we make no distinction between OOV and IV words.

Table 4: Evaluation results for the lecture retrieval task

Quality of trancription	run	transcription(s)		retrieval model	MAP
		word	syllable		
MATCHED	baseline-1	√		SMART	0.268
	baseline-2	√		VSM	0.231
	AKBL-5		√	E-VSM	0.223
	AKBL-4		√	SMART	0.212
	AKBL-1	√	√	E-VSM	0.365
	AKBL-7	√	√	SMART	0.401
UNMATCHED	baseline-3	√		SMART	0.238
	baseline-4	√		VSM	0.225
	AKBL-6		√	E-VSM	0.223
	AKBL-3		√	SMART	0.208
	AKBL-2	√	√	E-VSM	0.341
	AKBL-8	√	√	SMART	0.367

3.4 Experiments

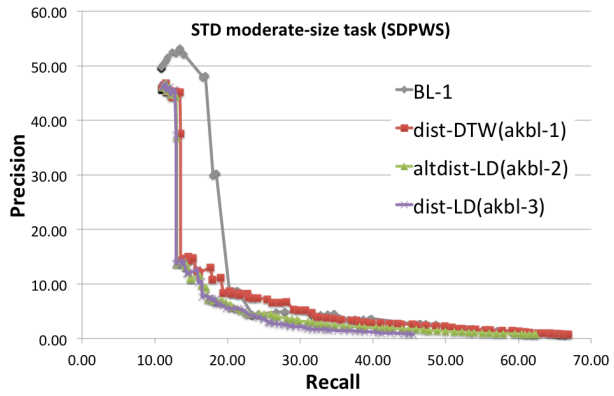

Figure 5: The recall–precision curves for the result on the moderate-size task (micro average)

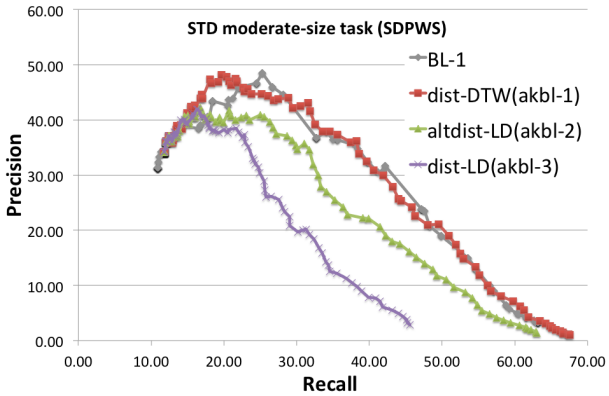

Figure 6: The recall–precision curves for the result on the moderate-size task (macro average)

We submitted eight runs for the lecture retrieval task. The four kinds of approaches were applied to either the matched or unmatched transcription.

STD-SCR system using the E-VSM (AKBL-5,AKBL-6) The proposed STD-based SCR system was used. It used no word-based transcription and was applied only on the syllable-based transcription, where AKBL-5 and AKBL-6 used matched and unmatched transcription, respectively. For the retrieval model, we used the extended vector space model that used the word-combination features, as described in Section 3.2.

STD-SCR system using SMART (AKBL-3,AKBL-4) It was the same system as that used in the AKBL-5 and 6 except that the retrieval model was replaced by the SMART retrieval method, which uses the TF–IDF term weighting with pivoted normalization [17]. AKBL-4 and AKBL-3 used matched and unmatched transcription, respectively.

Hybrid system using the E-VSM (AKBL-1,AKBL-2) We combined the conventional word-based SCR and the syllable-based STD-SCR methods, as described in Section 3.3. For the retrieval model, we used the extended vector space model that used the word-combination features, as described in Section 3.2. The run AKBL-1 was applied on the matched word-based transcription and matched syllable-based transcription, while the AKBL-2 was applied on the unmatched word-based and syllable-based transcriptions.

Hybrid system using SMART (AKBL-7,AKBL-8) It was the same system as that used in the AKBL-1 and 2 except that the retrieval model was replaced by the SMART retrieval method, which uses the TF–IDF term weighting with pivoted normalization [17]. AKBL-7 and AKBL-8 used matched and unmatched transcription, respectively.

The experimental results are summarized in Table 4. The experiment showed that the proposed STD-based SCR method performed well, even though it relied only on the degenerated syllable-based recognition result. Especially on the unmatched condition, where the word error rate was higher, the performance of the STD-based SCR was almost same as that of the conventional SCR. It also showed that the proposed extended vector space model (E-VSM) performed better than the SMART method. Furthermore, the hybrid SCR system of the STD-SCR and the conventional SCR successfully outperformed the baseline conventional SCR methods. We also found that the SMART was more effective than the E-SVM for the hybrid system.

4. DETERMINING THE RANGE OF RELEVANT PASSAGE

4.1 Methods for Passage Retrieval

The SpokenDoc passage retrieval task differs from a conventional document retrieval task in that the segments of passage are not predefined in advance. Therefore, it is required both to determine the boundary of the passage in the document and to rank them according to their relevancy to the query topic. Thus, we extended the conventional document retrieval method to that designed for the passage retrieval.

4.1.1 *Using the Neighboring Context to Index the Passage*

Passages from the same lecture may be related to each other in the passage retrieval task, whereas the target documents are considered to be independent of each other in a conventional document retrieval task. In particular, the neighboring context of a target passage should contain related information. It would seem appropriate for the passage retrieval task to use the neighboring context to index the target passage. Normally, a passage D is indexed by its own term frequencies $TF(t, D)$ of the terms $t \in D$. This can be extended to use the neighboring context for indexing. For the context $context_n(D)$, the preceding n utterances and the following n utterances are used. Therefore, we use

$$TF_{ext}(t, D) = \beta TF(t, D) + TF(t, context_n(D)) \quad (11)$$

where β is introduced to specify the relative importance of D and $context_n(D)$. In our implementation, an utterance is used for D, n and β are set to 5 respectively through the preliminary experiments. This is the same technique we used at NTCIR-9 SpokenDoc SDR task[12].

Although above approach achieved retrieving passages, its results are fixed-length segment. For example, our implement used n=7, thus retrieved passages segment length are always 15 utterances. Since relevant passages are seemed that they have various variable-length segments, we propose automatic estimating passage boundary method.

4.1.2 *Automatic Estimation of Passage Boundary*

When Given the query Q, we first retrieve relevance utterance in spoken document set using Neighboring Context to Index approach which described in section 4.1.1, and obtain pair(D, x) which include spoken document D and relevance utterance position x in D from retrieved results. Each pair is ranked according by their similarity score, and top 4000 pairs are consisting pair set $\mathbf{P}$ in our implementation.

Then, we define d_x as x-th utterance in D from each pair, and find most relevance segment boundary$(\hat{f}, \hat{t})$ about each d_x as follows

$$(\hat{f}, \hat{t}) = \underset{(f,t) \in \{(x-i, x+j) \mid 0 \le i \le l, 0 \le j \le l\}}{\operatorname{argmax}} sim(Q, d_f^t) \quad (12)$$

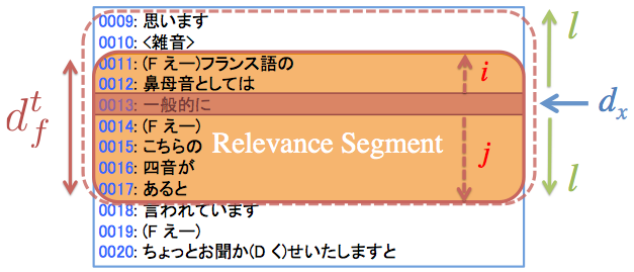

Figure 9: Automatic estimation of passage boundary

Where, $(\hat{f}, \hat{t})$ is pair of f-th and t-th utterance position in D, and d_f^t is a sequence of utterances between i-th and j-th utterances in D. As shown in Figure 9, we calculate similarity between Q and d_f^t with every possible combination of i and j, and finally we submit the most similar passage segment $[x-i, x+j]$. As a result, submitted passages have each different length. In contrast with method that no considered relevancy between Q and passage segment length described in section 4.1.1, this method also consider passage segment length and its relevancy. Therefore, we expect to obtain more relevance passage segment.

In the experiment, retrieved results are obtained using Equation(12) with every pair in $\mathbf{P}$ and ranked according by their similarity score.

4.1.3 *Penalizing Neighboring Retrieval*

In applying context indexing or automatic estimating passage boundary, neighboring passages tend to be retrieved at the same time as they share the same indexing words. This is not adequate from the perspective of retrieval systems because such systems output many redundant results. For this reason, we penalize a retrieval result that is neighbor to another result that has been output previously. In practice, the retrieved passage is discarded from the output list, if there are other results already retrieved within an utterance neighborhood of it.

As previous our run in NTCIR-9 shows[12], this approach significantly improves the performance in combination with context indexing. So we also use this approach in this round continuously.

4.2 Retrieval Models

4.2.1 *Relevance Models*

Levrenko and Croft [20] proposed relevance models as an information retrieval model. They define the relevance class R to be the subset of documents in a collection C, which are relevant to some particular information need, i.e. $R \subset C$. A relevance model is the probability distribution $P(w|R)$, where $w \in V$ is a word in a vocabulary V . $P(w|R)$ is estimated from a given query Q as follows.

$$P(w|R) = \frac{1}{P(Q)} \sum_{D \in C} P(w|D) \prod_{i=1}^{|Q|} P(q_i|D) \qquad (13)$$

where $P(Q)$ is constant with respect to Q.

Then, $P(w|R)$ is used to rank the documents $D \subset C$ by using the Kullback-Leibler divergence between th e distributions $P(w|R)$ and $P(w|D)$:

$$H(R||D) = -\sum_{w \in V} P(w|R) \log P(w|D) \qquad (14)$$

Relevance models can be seen as an implementation of pseudo relevance feedback, which is a sort of query-expansion technique using the target document collection, i.e. the query Q is expanded with the related words in the collection C through the estimation of the relevance model $P(w|R)$. In out implementation, we use only top-K documents D with $\prod_{i=i}^{|Q|} P(q_i|D)$ in Equation(13), and vocabulary V is consisted from top-800 w by $P(w|R)$.

Applying relevance models directly to our passage retrieval, specifically the context-indexing method described in Section 4.1.1 is problematic. Because context indexing uses neighboring utterances to index a document (an utterance), several neighboring documents share the same index words. This makes the estimated $P(w|R)$ inaccurate.

In order to deal with this problem, no context-expanded documents, i.e. a set of utterances, are used in the estimation of $P(w|R)$, but then context-expanded documents are ranked using $P(w|R)$. Namely, $P(w|R)$ is estimated as follows:

$$P(w|R) = \sum_{d \in c} P(w|d) \prod_{i=1}^{|Q|} P(q_i|d) \qquad (15)$$

where d and c are an utterance and a set of utterances, respectively. Then, the context-expanded documents $\hat{D} \subset \hat{C}$ are ranked by the following equation:

$$H(R||D) = -\sum_{w \in V} P(w|R) \log P(w|\hat{D}) \qquad (16)$$

In a similar way, we regard d_t^f as $\hat{D}$ for calculating $sim(Q, d_t^f)$ in Equation(12).

4.2.2 *Query Likelihood Model*

We also applied the query likelihood model[6] as our retrieval model for SCR for similarity between query and document. We use the probability $P(Q|D)$ that a query Q is constructed from a relevant document D:

$$P(Q|D) = \prod_{q \in Q} P(q|D)^{TF(q,Q)} \qquad (17)$$

$p(q|D)$ is estimated with dirichlet smoothing[22]:

$$P(q|D) = \frac{TF(q,D)}{|D| + \mu} + \frac{\mu}{|D| + \mu} \frac{TF(q|C)}{|C|} \qquad (18)$$

where $|D|$ is number of words in D, $|C|$ is number of document in C, μ is smoothing parameter. The $P(Q|D)$ is used to rank the documents $D \subset C$. In experiment, query likelihood model based similarity was obtained from Equation(17) instead of relevance models based similarity Equation(14), and the context-expanded document $\hat{D} \in \hat{C}$ or d_t^f are used as D.

4.3 Experiments

We submitted three types of runs. One detects variable length passage segments estimated automatically, as described in Section 4.1.2. The others have fixed-length segments, as described in Section 4.1.1. The latter two are the same method used in previous NTCIR-9 SpokenDoc. As the retrieval model, either Query Likelihood Model or Relevance Model is applied, so that we submitted six runs in total. All runs used neighboring penalty, as described in Section 4.1.3. All runs target the *matched* transcription.

The runs akbl-1,2, and 3 are using Relevance Model as the similarity calculation between query and passage. The run akbl-4, 5, and 6 are using Query Likelihood Model. The runs akbl-1 and 4 target variable-length segments, each of which consists of 1 to 25 utterances and are tuned against fMAP score. The akbl-2, 3, 5, and 6 target fixed-length segments, each of which consists of 15 utterances constantly. The akbl-2, and 5 are tuned against fMAP score while the akbl-3, and

Table 5: Experimental results of passage retrieval

run	uMAP	pwMAP	fMAP	retrieval model	definition of passage	tuning against
baseline-1	0.133	0.100	0.087	GETA(SMART)	fixed length	-
baseline-2	0.092	0.082	0.068	GETA(TFIDF)	fixed length	-
AKBL-1	0.102	0.088	0.063	Relevance Model	variable length	fMAP
AKBL-2	0.129	0.125	0.086	Relevance Model	fixed length	fMAP
AKBL-3	0.131	0.137	0.093	Relevance Model	fixed length	pwMAP
AKBL-4	0.131	0.123	0.087	Query Likelihood	variable length	fMAP
AKBL-5	0.122	0.132	0.083	Query Likelihood	fixed length	fMAP
AKBL-6	0.126	0.139	0.089	Query Likelihood	fixed length	pwMAP

5 are against pwMAP score. We used Spoken-Doc1 dryrun 39 queries for training these runs.

The results are shown in Table 5. It showed that there was no significant difference between the two retrieval models, i.e. Query Likelihood and Relevance Model. It also showed that the estimation of the variable length passage did not improve the performance of the passage retrieval. On the other hand, our language modeling approach performed well, compared with the baseline runs and the runs from the other SpokenDoc-2 participants.

5. CONCLUSIONS

In this paper, We investigated three methods for SpokenDoc-2, i.e. the Distance-ordered spoken term detection, the STD-based spoken content retrieval, and the automatic determination of the passage boundaries, which were applied to the STD, the SCR lecture retrieval, and the SCR passage retrieval tasks, respectively.

6. REFERENCES

[1] T. Akiba and K. Honda. Effects of query expansion for spoken document passage retrieval. In *Proceedings of International Conference on Speech Communication and Technology*, pages 2137–2140, 2011.

[2] T. Akiba and K. Honda. Effects of query expansion for spoken documnet passage retrieval. In *Proceedings of International Conference on Speech Communication and Technology*, pages 2137–2140, 2011.

[3] T. Akiba, H. Nishizaki, K. Aikawa, X. Hu, Y. Itoh, T. Kawahara, S. Nakagawa, H. Nanjo, and Y. Yamashita. Overview of the NTCIR-10 SpokenDoc-2 task. In *Proceedings of The Tenth NTCIR Workshop Meeting*, 2013.

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

[5] B. Chen, H. min Wang, and L. shan Lee. Discriminating capabilities of syllable-based features and approaches of utilizing them for voice retrieval of speech information in mandarin chinese. *IEEE Transactions on Speeh and Audio Processing*, 10:303–314, 2002.

[6] W. B. Croft and J. Lafferty. Language modeling for information retrieval. *Kluwer Academic Publishers*, 2003.

[7] 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*, pages 73–76, 2007.

[8] K. Iwami, Y. Fujii, K. Yamamoto, and S. Nakagawa. Efficient out-of-vocabulary term detection by n-gram array in deices with distance from a syllable lattices. In *Proceedings of International Conference on Acoustics Speech and Signal Processing*, pages 5664–5667, 2011.

[9] A. Jansen, K. Church, and H. Hermansky. Towards spoken term discovery at scalewith zero resources. In *Proceedings of International Conference on Speech Communication and Technology*, pages 1676–1679, 2010.

[10] G. J. F. Jones, J. T. Foote, K. S. Jones, and S. J. Young. Retrieving spoken documents by combining multiple index sources, 1996.

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

[12] T. Kaneko, T. Takigami, and T. Akiba. Std based on hough transform and sdr using std results: Experiments at ntcir-9 spokendoc. In *Proceedings of The Ninth NTCIR Workshop Meeting*, pages 264–270, 2011.

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

[14] 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*, pages 318–321, 2007.

[15] Y.-c. Pan and L.-s. Lee. Performance analysis for lattice-based speech indexing approaches using words and subword units. *Trans. Audio, Speech and Lang. Proc.*, 18(6):1562–1574, Aug. 2010.

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

[17] A. Singhal, C. Buckley, and M. Mitra. Pivoted document length normalization. In *Proceedings of ACM SIGIR*, pages 21–29, 1996.

[18] K. Sugimoto, H. Nishizaki, and Y. Sekiguchi. Effect of document expansion using web documents for spoken documents retrieval. In *Proceedings of the 2nd Asia-Pacific Signal and Information Processing Association Annual Summit and Conference*, pages 526–529, 2010.

[19] M. Terao, T. Koshinaka, S. Ando, R. Isotani, and A. Okumura. Open-vocabulary spoken-document retrieval based on query expansion using related web documents. In *Proceedings of International Conference on Speech Communication and Technology*, pages 2171–2174, 2008.

[20] V.Lavrenko and W. B. Croft. Relevance models in information retrieval. *In Language Modeling for Information Retrieval*, pages 11–56, 2003.

[21] M. Wechsler, E. Munteanu, and P. Schäuble. New techniques for open-vocabulary spoken document retrieval. In *Proceedings of the 21st annual international ACM SIGIR conference on Research and development in information retrieval*, SIGIR '98, pages 20–27, New York, NY, USA, 1998. ACM.

[22] C. Zhai and J. Lafferty. A study of smoothing methods for language models applied to information retrieval. *ACM Transactions on Information Systems (TOIS)*, 22(2):179–214, 2004.

[23] 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*, pages 415–422, 2006.