A robust/fast spoken term detection method based on a syllable $n$-gram index with a distance metric

Seiichi Nakagawa, Keisuke Iwami*, Yasuhisa Fujii, Kazumasa Yamamoto

Department of Computer Science and Engineering, Toyohashi University of Technology, 1-1 Hibarigaoka, Tempaku-cho, Toyohashi, Aichi 441-8580, Japan

Received 5 December 2011; received in revised form 29 November 2012; accepted 2 December 2012

Available online 22 December 2012

Abstract

For spoken document retrieval, it is crucial to consider Out-of-vocabulary (OOV) and the mis-recognition of spoken words. Consequently, sub-word unit based recognition and retrieval methods have been proposed. This paper describes a Japanese spoken term detection method for spoken documents that robustly considers OOV words and mis-recognition. To solve the problem of OOV keywords, we use individual syllables as the sub-word unit in continuous speech recognition. To address OOV words, recognition errors, and high-speed retrieval, we propose a distant $n$-gram indexing/retrieval method that incorporates a distance metric in a syllable lattice. When applied to syllable sequences, our proposed method outperformed a conventional DTW method between syllable sequences and was about 100 times faster. The retrieval results show that we can detect OOV words in a database containing 44 h of audio in less than 10 m sec per query with an $F$-measure of 0.54.

© 2012 Elsevier B.V. All rights reserved.

Keywords: Spoken term detection; Syllable recognition; $N$-gram; Distant $n$-gram; Out-of-Vocabulary; Mis-recognition

1. Introduction

The wide availability on the web of such multimedia data as audio continues to grow. Information can be found using an existing textual search engine if the target data are comprised of such textual information as transcriptions of broadcast news or newspapers. However, efficient robust spoken document retrieval (SDR) or spoken term detection (STD) methods have not yet to be established, since system designers face specific problems, such as recognition errors and Out-of-vocabulary (OOV) terms. The SDR task, which seeks suitable documents or passages based on the query, is usually performed using STD results. The aim of this research is to develop a robust and efficient STD method.

The standard STD method uses a textual search on Large Vocabulary Continuous Speech Recognizer (LVCSR) transcripts. However, certain OOV terms exist that are unregistered in the dictionary of the speech recognizer. Consequently, detecting OOV terms is impossible using existing textual search engines because the words are not output in the LVCSR’s recognition results. The advantage of using a sub-word unit based speech recognition system is that it can ignore grammatical constraints and recognize OOV terms (Ng, 1998). Therefore some hybrid retrieval systems have been proposed. Mamou et al. (2007) used a hybrid speech recognition system of word and phone-units. Akbacak et al. (2008) used word and graphone-units. Thus, to deal with OOV terms, we use a sequence of syllables corresponding to an OOV term, which is recognized as a syllable lattice and is the place where it is searched for. For German, a retrieval method has been proposed based on the weighted Levenshtein distance between syllables (with words containing only one syllable at a ratio of 1/2) (Larson and Eickeler, 2003). In Chinese (with 440 syllables in total), the syllable unit is often used as the basic unit for recognition/retrieval (Wang, 2000). In addition, based on elastic matching between two syllable sequences, other retrieval
methods have been suggested for dealing with recognition errors (Wechsler et al., 1998). In Japanese, Kanda et al. (2008) proposed a hierarchical dynamic time warping (DTW) matching method between phoneme sequences, where a coarse matching process is followed by fine matching. However, their method still consumes a great deal of computation time and memory storage.

A phoneme-based n-gram has been proposed in various retrieval methods, usually with a bag of words or partial exact matching (Allauzen et al., 2004; Saraclar and Sproat, 2004). For document retrieval, Chen et al. (2000) used such skipped (distant) bigrams as sₙ₋₁, sₙ₋₂, sₙ for syllable sequence sₙ₋₁, sₙ₋₂, sₙ for an SDR task. Phoneme recognition errors such as substitution mistakes have not been explicitly considered (Ng et al., 2000; Dharianipragada and Roukos, 2002) in OOV term retrieval. Mamou et al. (2007) proposed fuzzy matching between a query and an n-gram of phones indexed in an inverted index from recognition results. Recently, Chaudhari and Picheny (2012) proposed a similarity measure for the matching by using unigram, bigram and trigram phone sequence confusions. Their method consumes much computation time (Chaudhari and Picheny, 2012). Allauzen et al. (2004), Can et al. (2009) and Parada et al. (2009) used a weighted finite state transducer (WFST) based indexing system, that allows the lattice representation of recognized results to be used and maps each substring to the set of indexes in the automata. However, this approach cannot treat insertion/deletion errors, instead, they used a query expansion technique for them. Akbacak et al. (2008) used hybrid recognition systems containing both words and sub-word (graphones) units to generate hybrid lattice indexes. They made n-gram indexes of graphones from a graphone lattice. In Chinese, the performance of character- or syllable-based spoken document/term retrieval is comparable with that of a word-based one (Meng et al., 2000) even for in-word vocabulary (IV). This finding results from a special property of Chinese (in which almost all words consist of only one or two syllables).

The Japanese language consists of about 116 syllables, and thus a syllable unit is suitable for the retrieval of spoken OOV words, where we note some different definitions of the number of Japanese syllables. These depend on how to represent the pronunciation of loan words and how to discriminate between syllables and moras. We prepared a syllable lattice or a confusion network that is made from syllable units for sub-word units and used an n-gram array to retrieve the OOV terms. Katsurada et al. (2009) constructed a table using the suffix arrays of syllable recognition results and suggested a way of exploring the table while allowing for recognition errors during the search (Katsurada et al., 2009). Their method performs a DTW method, but cannot deal with a syllable lattice or confusion network.

However, for IV words, precision based on sub-word units deteriorates more than compared with the textual searching of a word unit. In this paper, we perform a text search using the results of a conventional LVCSR and a sub-word sequence search using the results of sub-word sequence recognition, if the query is an IV term. On the other hand, if the query is an OOV term, we search using the recognition results of the sub-word sequence prepared beforehand (Saraclar and Sproat, 2004). We use syllables as the sub-word units, since Japanese consists of 116 syllables.

The n-gram information of syllables in a syllable lattice or confusion network is maintained by a data structure called an n-gram array, which is added by index information based on an inverted file. Although such approaches have been proposed (Mamou and Ramabhadran, 2008; Saito et al., 2012), all methods did not take into consideration of the connectivity of retrieved n-grams or recognition errors.

We deal with three kinds of recognition errors in the syllable lattice or confusion network. First, to handle substitution errors, we use a trigram array that considers the m-best and dummy syllables in the syllable lattice. To tackle insertion errors, we create an n-gram array that permits a one-distance n-gram. Finally, to address deletion errors, we search for edited queries from which one syllable has been deleted. Of course, the text retrieval for IV terms depends on the performance of the LVCSR system’s performance.

Since the false detection rate increases when dealing with recognition errors, we must prune the detection candidates. Typically, as in the DTW method, a string is used to elastically match pruning candidates. However, since DTW processing is more time consuming than index-based searches, we use an n-gram array incorporating distance metrics that can account for recognition errors. The processing time is significantly improved using this method. When applied to syllable sequences, the proposed method outperformed a conventional DTW method between syllable sequences and was about 100 times faster. The retrieval results show that we can detect OOV words with an F-measure of 0.54 in a database containing 44 h of audio in less than 10 m sec per query.

The main contribution of this work is our proposal of a new fast/robust STD technique based on an n-gram index with distance that considers recognition errors and the connectivity of adjacent trigrams, and experimentally shows its effectiveness. Although our method is language independent, the order of n-gram depends on the suitable size of the sub-word units. In English, we believe that the phoneme or grapheme-unit is suitable for the 3 to 5-grams. We could try a 4-gram for 116 Japanese syllable units. It is possible to implement our method for 3-gram of 440 Chinese syllable units. Recently, various retrieval systems based on combinations of systems have been proposed and have improved performance (Natori et al., 2010; Parada et al., 2010). Nevertheless, we focus on a retrieval method based on a single system in this paper.

The remainder of this paper is organized as follows. In Section 2, we give an overview of our retrieval system
and, in Section 3, present the n-gram array index method that incorporates a distance metric and considers recognition errors. Evaluation results are given in Section 4 and a conclusion in Section 5.

2. System overview

2.1. Spoken term detection procedure for IV and OOV

We propose an STD system for OOV words using a syllable lattice or confusion network with mis-recognized syllables using an n-gram of syllables for the STD, particularly for OOV terms. A flow chart of the proposed method is illustrated in Fig. 1. Our proposed method is composed of the following six steps:

1. The spoken document is recognized by an LVCSR for IV words and by a continuous syllable recognition system for dealing with OOV and mis-recognized words.

2. The syllable lattice or confusion network is generated by a continuous syllable recognition system implemented by the SPOJUS++ in-house LVCSR system (Fujii et al., 2011). The system memorizes the m-best syllables at every syllable boundary of the best syllable sequence. The m-best syllables are represented by a lattice structure as shown in Fig. 2, which is a sausage-type confusion network. We call this a syllable lattice hereafter.

3. From the syllable lattice, an n-gram index table is generated from the inverted file, which consists of position, index (n-gram) and distance. The distance is calculated on the basis of substitution and insertion errors. A word index table is also generated from the LVCSR result.

4. A query consisting of IV words is retrieved using a standard text search technique from the LVCSR results. A search for the OOV terms or the mis-recognized words in the IV terms by the LVCSR is conducted by searching in the index table. If the length of a query in the syllables is larger than n in the n-gram, the query sequence is divided into parts with length n. Each part is retrieved from the index table, and the retrieved results for these plural parts are merged by considering the connectivity between adjacent n-grams.

5. To handle the mis-recognition errors of the LVCSR, the system also searches for IV terms using the same method as for OOV term detection.

6. Finally, the system combines the retrieval results of the text and syllable sequence searches.

2.2. Implementation of OOV word retrieval method using an n-gram with distance

The n-gram information of syllables is maintained in a data structure called an n-gram array that consists of index and syllable distance information for each n-gram. Fig. 3 shows how a trigram array is arranged. First, the appearance positions of the syllables in a recognized syllable lattice for a spoken document are located. Then an n-gram of the syllable is constructed at every appearance position. Next, the n-gram is sorted in lexical order so that it can be searched for quickly using a binary search algorithm. The search process for an n-gram array includes three steps. First, a query is converted into a syllable sequence. Second, an n-gram of the query is constructed. Finally, the n-gram in a query is retrieved from the n-gram array. A query consisting of more than n + 1 syllables is retrieved using a combination of n-grams. A query consisting of less than 2n syllables but more than n + 1 syllables is separated into two n-grams for the first and second halves. Thus, the query is retrieved from the n-gram array twice. The retrieved results are merged by considering whether the position at which the detection result occurred in the first...
and second halves is the same. Similarly, a query with less than \(3n\) syllables but more than \(2n + 1\) syllables is retrieved by a sequence of syllables by dividing the query into three parts (Fig. 4). For example, when a query consists of six syllables, “i mi ka i se ki” in Fig. 4, the query’s syllable sequence is divided into two 3-gram; “i mi ka” and “i se ki.” If the first term, “i mi ka,” is detected at \(s_1 \sim t_1\) with a distance less than a threshold, that is, index position \(= s_1\), and the second term, “i se ki,” is detected at \(t_1 + 1 \sim u_1\) with a distance less than a threshold, that is, index position \(= t_1 + 1\), then “i mi ka i se ki” is detected at \(s_1 \sim u_1\). For a query consisting of five syllables, “ke i ta i so” in Fig. 4, the query sequence is divided into two 3-gram; “ke i ta” and “ta i so”. If the first term “ke i ta” is detected at \(s_2 \sim t_2\) and the second term “ta i so” is detected at \(t_2 \sim u_2\), then “ke i ta i so” is detected at \(s_2 \sim u_2\). For a query consisting of 4 syllables “ni N ge N”, the syllable sequence in the query is divided into two 3-gram; “ni N ge” and “N ge N”. If the first term is detected at \(s_3 \sim t_3\), and second term is detected at \(s_3 + 1 \sim u_3\), or \(t_3 - 1 \sim u_3\), then “ni N ge N” is detected at \(s_3 \sim u_3\). This merging process will also be described in Section 3.5.
2.3. OOV word retrieval by DTW for sub-word sequences (baseline)

Substitution, insertion and deletion errors occur in sub-word based automatic speech recognizers. The word detector has to find the candidate positions from the recognized sub-word sequence. We usually call this word spotting. We use a syllable as the sub-word unit, where each Japanese syllable consists of a consonant and a vowel, or a single vowel. Each word is expressed as a concatenation of syllables. For word spotting, the recurrence equation for DTW (between the sub-word sequence \( A = a_1a_2 \ldots a_i \) for a document and a query sub-word sequence \( B = b_1b_2 \ldots b_j \)) is given below (Iwami et al., 2010; Iwami et al., 2011). Let \( g(i,j) \) be the intermediate matching distance, and \( B(i,j) \) be the starting frame of the matching, that is, \( b_ib_{i+1} \ldots b_j \) is matched with \( a_ia_{i+1} \ldots a_i \), where \( i = B(i,j) \).

\[
g(i,j) = \begin{cases} 
(1) & g(i-1,j-1) + d(i,j) \\
(2) & g(i-2,j-1) + (d(i-1,j)+\text{Ins})/2 \quad (i \neq 0, 1) \\
& \text{o}r \quad g(i-2,j-1) + (d(i-1,j)+d(i,j))/2 \\
(3) & g(i-1,j-2) + d(i,j-1) + \text{Del} \quad (i \neq 0) \\
& \text{o}r \quad g(i-1,j-2) + d(i,j-1) + d(i,j)
\end{cases}
\]

\[
B(i,j) = \begin{cases} 
(1) & B(i-1,j-1) \\
(2) & B(i-2,j-1) \\
(3) & B(i-1,j-2)
\end{cases}
\]

In each expression, its label corresponds to the restricting condition for DTW in Fig. 5.

1. query matches the recognition result or substitution error
2. insertion error (Ins)
3. deletion error (Del).

We calculate the distance between a query and the retrieval result using the edit or Bhattacharyya distance (Iwami et al., 2010) between syllables as local distance “\( d \)”. In Japanese, almost all syllables consist of a consonant followed by a vowel, for example, /ka/, /gi/, /ze/, /mo/, /ky/, and /yo/, where /ky/ is regarded as a consonant. The irregular syllables consist of a single vowel-syllable (/a/, /i/, /u/, /e/, /o/) and a nasal syllabic sound (/N/). In Japanese, there are 116 syllables, including syllables for pronouncing such loaned words as “file [fa i ru]”. Distance “\( d \)” between syllables is defined as follows:

(a) ternary distance for two syllables (Syllable – Edit)-substitution:-

\[
d(a,b) = \begin{cases} 
0 & \text{if } a = b \\
1 & \text{if vowel or consonant is incorrect (e.g. /ka/ \( \rightarrow /a/ \), /ka/ \( \rightarrow /ko/ \))} \\
2 & \text{if both a vowel and a consonant are incorrect (e.g. /ka/ \( \rightarrow /go/ \))}
\end{cases}
\]

(b) binary distance for two phonemes (Phoneme – Edit)-

Both the recognized and query syllable sequences are converted to corresponding phoneme sequences. Then the distance between two phonemes is defined as follows:-substitution-

\[
d(a,b) = \begin{cases} 
0 & \text{if } a = b \\
1 & \text{if phoneme is incorrect}
\end{cases}
\]

(c) Bhattacharyya distance between two syllable-based HMMs-substitution:-

\[
BD(P_a, P_b) = \frac{1}{8} \left( \mu_a - \mu_b \right)^T \left( \frac{\Sigma_a + \Sigma_b}{2} \right)^{-1} \left( \mu_a - \mu_b \right) + \frac{1}{2} \log \left( \frac{1}{\left( |\Sigma_a| |\Sigma_b| \right)^{1/2}} \right)
\]

Fig. 5. Constrained conditions for DTW (1): substitution, (2): insertion, (3): deletion.
When using the $k$th candidate in a lattice, the distance is defined as follows:

$$d(a, b) \leftarrow \min_k \{d(a, b_k) + k \cdot c\}$$  \hspace{1cm} (10)

where $b_k$ and $k \cdot c$ denote the $k$th syllable candidate and the penalty distance.

The normalized DTW distance is calculated as:

$$g(i, J)/J$$

for the part of $\{B(i, J) \sim i\}$.  \hspace{1cm} (11)

If the normalized DTW distance is less than a given threshold, the position at which the query appears can be considered as the retrieval result. If it exceeds the threshold, it is rejected.

3. Solving mis-recognized sub-words in OOV detection

We must consider automatic speech mis-recognition such as substitution, insertion, and deletion errors. Our proposed retrieval method addresses three kinds of mis-recognition errors in syllable recognition.

3.1. Substitution errors

To handle substitution errors, we use an $n$-gram array constructed from the $m$-best of the syllable lattice (Iwami et al., 2010). An $n$-gram array is constructed using a combination of syllables in the $m$-best syllable lattice. Thus, for any one position in the lattice, there are $m^n$ kinds of $n$-grams. For example, even if the recognition result of the 1-best, “fu e ki e he N ka N”, has recognition errors, we can search for query “fu u ri e he N ka N (“Fourier Transform” in English)”, if a correct syllable is included in the $m$-best. We used an HMM based Bhattacharrya distance (Iwami et al., 2010) as the local distance between the first syllable and subsequent candidates (Fig. 6). The distance was defined as 0 for the first candidate. The distance is calculated as the distance between “fu u ri” in the target trigram and “fu e ki” in the 1-best trigram, where the distance is $d(e, u) + d(ki, ri)$ (Fig. 6).

Even if we use syllable lattices, some substitution errors will not be contained in the lattices. Therefore, we introduce dummy syllables and represent them by “*” to match any syllable that is not contained in the $m$-best recognition results. For example, if the recognition result of the $m$-best does not include “C”, the original method can not search for query “ABC”. In this case, since an index including the dummy syllable has $n$-gram AB*, we can retrieve query “ABC”. This improves the recall rate. On the other hand, the method has the potential for decreasing the precision rate. This problem is addressed by increasing the distance between “*” and any other syllable, where only one dummy syllable is allowed in a trigram. We used a large constant value for the local distance, such as $d(e, *)$ in Fig. 6. Note that this approach is different from the one distant bigram index method (Section 3.8). In the implementation, the dummy syllable is inserted into the last row in the syllable lattice, that is, $(m+1)$-th candidate as shown in Fig. 2.

3.2. Insertion errors

To address the insertion errors, we create an $n$-gram array that permits a one-distant $n$-gram. By taking into account the gap between appearance positions, we can deal with this error. For example, even if the recognition result, “fu ku u ri e he N ka N”, includes insertion error “ku”, we can search for the query “fu u ri e he N ka N”, if an $n$-gram array is allowed that considers a one-distant $n$-gram. Therefore, we can deal with one insertion error within each $n$-gram. The trigram of “fu u ri” is constructed as a skipped trigram from “fu ku u ri”, when “ku” is regarded as insertion error (Fig. 7).

3.3. Deletion errors

To handle deletion errors, we search for the query as above while allowing for the possibility that one syllable in it may have been deleted. Even if recognition result, “fu u e he N ka N”, has a deletion error, we can search for query “fu u ri e he N ka N”, if a syllable (‘ri’) in the query has been deleted (Fig. 8).

In Japanese, no big difference is found between orthography in a lexicon and pronunciation, so transliteral problems do not exist, but they do in English. Mamou and Ramabhadran (2008) treats this problem using query expansion. Of course, our approach can also be regarded as the expansion of the phonetic representation for a query because it considers deletion errors in ASR.

3.4. Reduction in index size

The size of the index increases dramatically when dealing with mis-recognized syllables. In $n = 3/m = 5$, we must
consider $5^3$ trigrams at every position in the lattice. If we consider insertion errors, this becomes $4 \times 5^3$ trigrams. If we include dummy syllables, the index increases even more. To address this problem, we used a condition where only the restricted patterns of substitution error actions are allowed in a trigram when constructing the index. Below, we discuss the experiments using each of the following conditions. In each condition, we use only a single dummy syllable.

- **reduced-1** two of the three syllables in a trigram are used as the recognition results from the $1 \sim 3$-best. The remaining one is used as the recognition result from the $1 \sim 5$-best+dummy.
- **reduced-2** two of the three syllables in a trigram are used as the recognition results from the $1 \sim 5$-best and $1 \sim 5$-best+dummy. The remaining one is used only as the $1$-best recognition result.
- **reduced-3** one of the three syllables in a trigram is used as the recognition results from the $1 \sim 5$-best+dummy. Another is used as the $1 \sim 3$-best, and the final one is used only as the $1$ best.

Table 1 lists the possible trigrams at every position in a syllable lattice and their coverage rates under the above conditions (refer to syllable recognition rates given in Table 4), where “coverage rate” means the probability that the correct trigram index is registered in the index table. For example, for the 5-best, the coverage rate is calculated as $0.91^3$, and for the 5-best+dummy (only 1 dummy syllable in a trigram), it is calculated as $0.91^3 + (1.0 - 0.91) \times 0.91^2 \times 3$.

### 3.5. Decision maker

The computation time for fine string matching, as for DTW, is linear with the number of detection candidates. To solve the problem more efficiently, we define a new distance metric to represent the number of allowed errors. N-gram arrays using a distance metric enable fast pruning of unreliable candidates without elastic string matching. The syllable distance in a substitution error is calculated using the Bhattacharyya distance between two HMM-based syllables from the 1-best recognition result (refer to Eqs. (7) and (8)). In other words, the syllable in the 1-best result always has a zero distance (see the column labeled “distance” in Fig. 6). Next, the syllable distance for an insertion error is represented by 1 (refer to the column labeled “insert” in Fig. 7). Finally, the syllable distance for a deletion error equals the number of syllables that were eliminated in the query (refer to the column labeled “deletion” in Fig. 8). In this study, the deletion error distance is either 0 or 1, since we allow up to one deletion error per n-gram.

The merging process for two detected adjacent n-grams is simple. Fig. 9 illustrates the merging process of general cases. For example, for the query “ABCDEF”, two trigrams “ABC” and “DFE” are searched for. If “ABC” is found at index position $s$ and at part of $(s \sim t)$, the index position of “DEF” should be $t + 1$. For query “ABCDEF”, its syllable sequence is divided into two trigrams: “ABC” and “CDE”. If “ABC” is found at index position $s$ and at part of $(s \sim t)$, the index position of “CDE” should be $t$. For query “ABCD”, the sequence is divided into “ABC” and “BCD”. If “ABC” is found at index position $s$ and at part of $(s \sim t)$, the index position of “BCD” should be $t - 1$ or $s + 1$. If these connectivity conditions for adjacent trigrams are satisfied, these detected trigrams are merged. We should notice, however, that the distance for substitution is doubly added to the distance score after merging and the average distance per syllable is not correctly calculated for such a case of “C” in Fig. 9(b). To simplify the merging process (or reduce the processing time and memory size), we regarded ABC and CDE as independent tri-
grams each other. Of course, we should revise the merging process to improve the retrieval performance. It is one of future works (Sakamoto et al., 2013).

The retrieval threshold is defined in accordance with the query length in syllables. After merging the detected n-grams, the average distance per syllable is calculated, and if the averaged distance is smaller than the adaptive threshold, the corresponding location is returned as the query position.

3.6. Memory and search time

The memory required by our proposed method corresponds to the amount of memory used by the index for the trigram array. Although our method can be expanded to any n-gram array, if we use higher order of n, the index size will become larger. The suitable order depends on the number of sub-word units. The amount of memory in a trigram array is estimated by Eq. (12):

\[
M = M_1 \times S_1 + M_2 \times S_2
\]

\[
S_1 = \text{memory size of } \{\text{trigram type + number of entries}\}
\]

\[
S_2 = \text{memory size of } \{\text{position + insertion distance + substitution distance}\}
\]

Here, \(M_1\) is the different number of trigrams, \(M_2 (\gg M_1)\) is the number of trigram entries in the index, \(S_1\) is the amount of memory for a particular kind of trigram, and \(S_2\) is the amount of memory for a trigram entry (refer to Fig. 10 and Table 2). We used two different data structures for representing indices, that is, “simple” and “compressed”, respectively.

Three syllables are included in a trigram, and each syllable consists of one or two characters (or phonemes). We assume that spoken documents are spoken at six syllables per second considering the 3-best in the lattice. Using the simple expression given above, a trigram array of about 2 MB is created for a spoken document of one hour for the 3-best. Using the 5-best, the size increases to 12 MB. Additionally, the index size increases 4 fold when considering insertion errors, for example, ABC, ABD, ACD and BCD for ABCD. For this simple expression, the memory size per hour is about 48 MB for the 5-best. If we implement the compressed expression given in Table 2, the required memory size is 17 MB per hour. In our experiment, we used 44 h of spoken documents. For them, the amount of required memory size becomes 17 MB \times 44 = 756 MB (Fig. 14). If we introduce a dummy syllable into the trigram index, the number of possible trigrams at every position increases from 125 to 200 without considering the insertion errors or from 500 to 800 by considering them, that is, 8/5 times. The required memory size becomes 756 MB \times 8/5 = 1210 MB (Fig. 14). We can reduce the memory size by about half using a pruning method (Iwami et al., 2010).

If a 4-gram based index method is adopted (where the 4-gram is used for retrieving a word consisting of the 5-best), the number of indices increases significantly and the amount of memory required becomes enormous. When retrieval is carried out with a 4-gram, only one insertion error per four syllables can be tolerated. By considering the required amount of memory, speech recognition performance, retrieval performance, search speed, and so on, we must determine whether a 3 or 4-gram is better.

Since the proposed method uses a binary search, the computational complexity is \(O(\log_2 M_1)\) for a single search of a trigram. Then the algorithm sequentially checks the corresponding trigram entries at different positions of \(M_2 / M_1\) on average and performs post-processing for the connectivity. Actually, our method divides a query into a number of parts, each of which contains three syllables, and performs the search multiple times. For example, when the query is composed of seven syllables, our method searches for the query in three parts from a trigram array (Fig. 4). If there are \(M_1\) types of indexes, the computation of the binary search is \(k \log_2 M_1\), where \(k\) is the number of divisions of the query. The retrieved results are merged into a final result while considering the connectivity of the adjacent retrieved results. Therefore, the computational complexity depends largely on the query length and the number of retrieved results (i.e., proportional to the duration of the speech document).

Additionally, the number of n-gram searches is proportional to the number of deletion actions. For example, if
the query is composed of six syllables, deletion actions are activated six times. On the other hand, a search from a word inverted index based on the LVCSR results for IV words is much faster than an n-gram array, because the word inverted index size is relatively small and we do not need to consider any recognition errors.

3.7. IV word retrieval by combining LVCSR and syllable recognition results

We use continuous syllable recognition results for OOV queries, but for IV queries, we combine the LVCSR and syllable recognition results (Fig. 1). Our proposed method allows searching for the following different types of queries:

- queries containing only IV terms, where both the LVCSR and syllable recognition results are used,
- queries containing only OOV terms, where the syllable recognition results are used, and
- queries containing both IV and OOV terms, where the syllable recognition results are used. For example, “MEIKEN LASSIE” is a compound noun consisting of “MEIKEN” (well bred dog) from the IV words and “LASSIE” (Lassie; name of dog) from the OOV words.

A search for an IV term includes three stages. First, a query of an IV term is retrieved using an inverted index that considers the word confusion network of the LVCSR result. We can accurately detect the IV term using the LVCSR result, if the word is correctly recognized. Second, for word mis-recognition, our approach retrieves the same IV term using the n-gram array created from the syllable recognition results. Finally, the system combines the two results using the OR operator. Although false alarms may occur in the retrieved results when using an n-gram array, it is robust for mis-recognition and improves the recall rate.

3.8. Comparison with other approaches

(a) DTW approach

The most standard retrieval method for a sub-word/lattice sequence is the DTW-based approach. Similar to the DTW approach, our proposed approach recovers three recognition errors: substitution, insertion, and deletion. Both approaches can
define any distance metric for the three types of errors. However, even if the path constraint of the DTW approach is limited to a $1/2 \sim 2$ slope, the constraint is still weak when matching two syllable sequences. On the other hand, our proposed approach limits the number of insertion and deletion sequences. Roughly speaking, the slope of the proposed approach is $2/3 \sim 4/3$. Therefore, we expect that our proposed method will achieve a better precision rate, but conversely, a worse recall rate than the DTW approach. This result will be described in the experimental results in the next section. One advantage of the DTW approach is that it requires no memory in advance, but a disadvantage is its huge computation cost. Our proposed approach requires much memory for the index table, but it is very fast and the retrieval performances are comparable. Katsurada et al. (2009) approximately implemented the DTW approach using a suffix array for a recognized syllable sequence, a tree search technique that permits substitution, insertion and deletion errors, and a pruning technique. Although their method is very fast, it is difficult to extend to a syllable lattice or confusion network. Mamou et al., 2007 proposed a fuzzy matching approach that resembles DTW using $n$-gram phones indexed in an inverted index. However, their approach requires fuzzy matching between $n$-gram phone sequences of a recognition result and a phone sequence in a query. It is difficult to check exactly the connectivity of adjacent phones over $n$-grams. Therefore, Saito et al. (2012) performed a term-based second stage DTW for the candidate parts retrieved from the $n$-gram-based first stage DTW results. Their method still consumes computation time.

(b) $n$-gram and distant $n$-gram

The standard method based on a “bag of words” of an $n$-gram & distant $n$-gram (Chen et al., 2000; Allauzen et al., 2004) resembles our proposed method. A trigram with an insertion action corresponds to a 1 distant trigram or a skip trigram, and a trigram with a dummy syllable corresponds to a bigram and a 1 distant or skip bigram. While our method uses a trigram with a distance metric, of course, the standard and $n$-gram/distant $n$-gram approaches can also include an embedded distance metric. However, these $n$-gram & distant $n$-gram approaches were usually applied to SDR tasks, not STD tasks, because it is difficult to retrieve exactly the locations of spoken terms by these approaches. The difficulty is caused by the uncertainty of the order or connectivity of $n$-grams. On the other hand, our method considers the order or connectivity of the $n$-gram/distant $n$-gram and solves the above problem. Therefore, our approach gives a comparable performance to the DTW approach.

4. Evaluation and results

4.1. Experimental setup

We used the 44 h of core data from the Corpus of Spontaneous Japanese (CSJ) as experimental data (Itoh and Nishizaki, 2010) and SPOJUS++ (Fujii et al., 2011), which were developed in our laboratory as LVCSR. The context-dependent syllable-based HMMs were trained on 2707 lectures within the CSJ corpus excluding the core data. In our experiments, the query set was divided into 19 IV and 31 OOV queries. We used the query set for a formal run in NTCIR9 (Akiba et al., 2011). In almost all the OOV queries, only 3 ~ 5 times appear in 44 h. So the detection task is relatively difficult.

An LVCSR with 28 k words and continuous syllable recognition was performed by 928 context-dependent syllable-based HMMs. We used a left-to-right HMM that consisted of four states with self loops and four Gaussians with full covariance matrices per state. Table 3 summarizes the speech analysis conditions. We used word based 3-gram and syllable-based 4-gram, as the language models. The cutoff value in the language model was set to four, which means that if a word appeared fewer than three times in the training corpus, it was regarded as an OOV term.

The syllable recognition rate, the LVCSR’s word recognition rate, and the syllable recognition rate converted from the recognized words are summarized in Table 4. The 1-best result for continuous syllable recognition is slightly worse than that for the syllable recognition rate converted from the LVCSR result.

We implemented the proposed method on a machine with the following specifications: Xeon 2.93 GHz, 6 core CPU (although we only used a single core), and 74 GB memory.

4.2. Preliminary comparison of DTW methods -baseline-

As a baseline, we used DTW with an asymmetric path weight Eq. (1), and the Bhattacharyya distance as the local distance, which was derived from the 5-best of the syllable lattice (Iwami et al., 2010). In addition, we compared various local syllable and phoneme-based edit distances (1-best, 5-best) for the DTW. Fig. 11 illustrates the retrieval

<table>
<thead>
<tr>
<th>Conditions for acoustic analysis of input speech.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Sampling rate</strong></td>
</tr>
<tr>
<td><strong>Pre-emphasis</strong></td>
</tr>
<tr>
<td><strong>Analysis window</strong></td>
</tr>
<tr>
<td><strong>Analysis frame length</strong></td>
</tr>
<tr>
<td><strong>Analysis frame shift</strong></td>
</tr>
<tr>
<td><strong>Feature parameter</strong></td>
</tr>
<tr>
<td></td>
</tr>
</tbody>
</table>
Table 4
Recognition results (%).

<table>
<thead>
<tr>
<th>Output</th>
<th>Del</th>
<th>Ins</th>
<th>Subs</th>
<th>Corr</th>
<th>Acc</th>
</tr>
</thead>
<tbody>
<tr>
<td>Syllable (1-best)</td>
<td>3.9</td>
<td>3.6</td>
<td>12.5</td>
<td>83.6</td>
<td>80.0</td>
</tr>
<tr>
<td>Syllable (3-best)</td>
<td>3.9</td>
<td>2.2</td>
<td>6.9</td>
<td>89.1</td>
<td>86.9</td>
</tr>
<tr>
<td>Syllable (5-best)</td>
<td>4.1</td>
<td>1.9</td>
<td>4.9</td>
<td>91.0</td>
<td>89.1</td>
</tr>
<tr>
<td>Converted syllable from LVCSR</td>
<td>4.1</td>
<td>2.3</td>
<td>12.5</td>
<td>83.3</td>
<td>81.1</td>
</tr>
<tr>
<td>WORD from LVCSR (1-best)</td>
<td>5.4</td>
<td>4.6</td>
<td>22.7</td>
<td>71.9</td>
<td>67.3</td>
</tr>
</tbody>
</table>

4.3. In-vocabulary term detection

Table 5 gives the IV term detection results (with the maximum $F$-measure). The column labeled “DTW” means the performance on syllable (lattice) sequences using the Bhattacharyya distance as the local distance. “LVCSR” represents the performance using the LVCSR (confusion network) results, and “3-gram” represents the performance using the 3-gram array from a recognized syllable (lattice) sequence. “LVCSR+DTW” and “LVCSR+3-gram” denote the combination of the LVCSR and DTW-based approaches, and the LVCSR and 3-gram approaches, respectively. Fig. 12 illustrates the comparative results for IV term detection.

When considering the balance between precision and recall, there is no difference for IV retrieval between the proposed method and DTW. As discussed in Section 3.8, our proposed trigram-based method outperforms the DTW method with regards to the precision rate for the condition of the same low recall rate at the range of 0.4 ~ 0.6, but worse for the recall rate for the condition of the same low precision rate at range of 0.0 ~ 0.5. Retrieval from the LVCSR results yields the best result of all three methods with an $F$-measure of 0.67. Additionally, by combining the LVCSR and a 3-gram array, we increased Recall from 0.44 to 0.68 and the $F$-measure from 0.59 to 0.77. The “3-gram+dummy” method improved the recall rate more than the “3-gram” method. However, the performance of the “3-gram+LVCSR+dummy” resembled the “LVCSR+3-gram” method ($F=0.77$). Although the combination of “LVCSR+DTW” showed the best condition of the same low recall rate at the range of 0.4 ~ 0.6, but worse for the recall rate for the condition of the same low precision rate at range of 0.0 ~ 0.5. Retrieval from the LVCSR results yields the best result of all three methods with an $F$-measure of 0.67. Additionally, by combining the LVCSR and a 3-gram array, we increased Recall from 0.44 to 0.68 and the $F$-measure from 0.59 to 0.77. The “3-gram+dummy” method improved the recall rate more than the “3-gram” method. However, the performance of the “3-gram+LVCSR+dummy” resembled the “LVCSR+3-gram” method ($F=0.77$). Although the combination of “LVCSR+DTW” showed the best
performance ($F=0.78$), the computation of the DTW-based approach is very expensive.

### 4.4 Out-of-vocabulary term detection

Table 6(a) shows the OOV term detection performance by addressing various recognition errors (with the maximum $F$-measure), and Fig. 13 illustrates the comparative results for OOV term detection. (1), (2), and (3) in Table 6 denote substitution, insertion, and deletion error processing described in Section 3, respectively. With respect to precision and recall, a “No processing” action means the result without considering the errors, that is, exact matching using only the first candidate. We also conducted an OOV retrieval experiment using only the 1-best/5-best syllable sequence and dummy syllables (Table 6(b)). The “1-best+(2)+(3)+dummy” in Table 6(b) gives the results when only using the first candidate with a dummy syllable. The process (1) for this case is not applied, because there is no alternative candidate for substitution errors. Although the dummy syllable method in Table 6(b) improved the retrieval performance more than “(1)+(2)+(3)” in Table 6(a) and DTW in Table 6(c).
The substitution error processing action for recognition errors outperformed the other error processing actions; the next best was the deletion error processing action. The substitution action using the 5-best was better than that using the 3-best. We obtained an F-measure of 0.52 for OOV term detection by combining all error processing actions. Finally, we obtained an F-measure of 0.54 by introducing “dummy syllables”. Although the value was lower than that for IV term detection, our proposed method remarkably outperformed the DTW-based method (F-measure of 0.48). However, as discussed in Section 3.8, although the precision rate using the proposed trigram method exceeded that for DTW on the same low recall rate at the range of 0.2 ~ 0.53, the recall rate was worse on the same low precision rate at the range of 0.0 ~ 0.18 (see Fig. 13). Note that the retrieval performance depends on a set of query terms and syllable recognition results. We obtained the DTW-based baseline result using other speech recognition results from the organizer of the NTCIR workshop (Akiba et al., 2011) for the same query set and documents, which is based on the edit distance between phoneme sequences. They reported F-measures of 0.573 for IV and 0.521 for OOV. Our proposed method outperformed our own baseline results and those of the organizer.
In Fig. 13, “3-gram+dummy” denotes the dummy syllable method described in Section 3.1 using the 3-gram index with dummy syllables. The “3-gram+dummy” method yields a more improved overall performance compared with the original “3-gram” method, especially its recall rate. The $F$-measure increased by an absolute 2% from 0.52 to 0.54.

Fig. 14 illustrates the comparative results for various index reduction methods. In Fig. 14, “dummy+reduce-1”, “dummy+reduce-2”, and “dummy+reduce-3” denote, respectively, the three conditions discussed in Section 3.4. Although the overall performance resembles the “3-gram+dummy” and “3-gram+dummy+reduce-2”, the index size was significantly reduced. In fact, the size of the index decreased from 1.3 GB to 689 MB.

Fig. 15 shows that our proposed current method (3-gram+dummy+LVCSR) is comparable with DTW+LVCSR for IV+OOV queries ($F$-measure=0.65). The result of the “3-gram+LVCSR” method (with $F$-measure=0.645, old version) was submitted as a formal run task on NTCIR9 (Akiba et al., 2011), which is not shown in Fig. 15. From the viewpoint of retrieval performance, we confirmed that our approach worked well.

4.5. Retrieval time

We also experimentally compared the average search times of the DTW and 3-gram methods. Fig. 16 compares them for average search time per query for a varying
number of syllables. The average search time using DTW was 600 ms, but the original trigram array method took about 1 ms for each query. DTW has a drawback; its processing time is proportional to the number of spoken documents. On the other hand, even if the number of spoken documents increases, the query can be searched for rapidly using a trigram array with the original trigram method. This difference becomes significant as the number of spoken documents increases.

Fig. 17 compares the original 3-gram and 3-gram+dummy syllable approaches with respect to the average search time per query for a varying number of syllables. The retrieval time using dummy syllables was about 6 ms slower than the original 3-gram method and about 100 times faster than DTW, because the number of retrievals was increased using the query or the trigram index containing dummy syllables. This problem may be due to our unoptimized algorithm. We will solve this problem in future work.

5. Conclusion

In this paper we described a Japanese spoken term detection method for spoken documents that is robust in considering OOV and mis-recognized words. To solve the problem of OOV keywords and mis-recognized words, the method uses individual syllables as sub-word units in continuous speech recognition and a 3-gram sequence of syllables as the retrieval unit. We proposed a 3-gram indexing/retrieval method that incorporates a distance metric in a syllable lattice to handle OOV words, recognition errors, and high speed retrieval. We significantly reduced false alarms using trigram arrays with error distance. We applied this method to an academic lecture presentation database that consisted of 44 h of speech, and obtained an F-measure of 0.54 for the OOV terms in the transcription of spoken documents with a syllable recognition rate of 91% for the 5-best candidates and a retrieval time for each query of only a few milliseconds. The proposed retrieval method outperformed the traditional DTW approach when applied to syllable sequences, and was about 600 times faster than DTW. Additionally, we improved the recall rate by introducing a dummy syllable symbol in the syllable lattice. To solve the problem of large index size, we introduced a constraint of substitution actions when constructing the index. By applying these functions, we successfully implemented a very robust/fast OOV term retrieval method.

Improving the retrieval performance remains important future work. One way of improving retrieval accuracy is to strictly define the insertion, deletion, and substitution error distance metric such as the metric between a dummy syllable and a syllable in a query, where the dummy syllable can be replaced with the first candidate syllable (Iwami et al., 2013). We should also implement the merging process strictly as described in Section 3.5. For example, a combination of trigram, bigram and unigram index is suitable to simplify the merging process (Sakamoto et al., 2013). Another way is to improve the syllable recognition rate by combining the results of several decoders (Nishizaki and Nakagawa, 2002). Finally, we could use the syllable’s likelihood obtained from the decoder, instead of the syllable distance to improve the retrieval accuracy.

References


