Reduction of Speech Data Posteriorgrams by Compressing Maximum-likelihood State Sequences in Query by Example

Takashi Yokota*, Kazunori Kojima*, Shi-wook Lee†, Yoshiaki Itoh*
*Iwate Prefectural University, Japan, E-mail: y-itoh@iwate-pu.ac.jp
†National Institute of Advanced Industrial Science and Technology, Japan, E-mail: s.lee@aist.go.jp

Abstract— **Spoken-term detection (STD) has recently attracted increased interest in speech-based retrieval research. STD is the task of finding sections in speech data matching a query consisting of one or more words. Query by example (QbE) using spoken queries is another important research topic in STD. Although the use of posteriorgrams (sequences of output probabilities generated by deep neural networks from speech data) is a promising approach for QbE, that method results in long retrieval times and excessive memory usage. We previously proposed a method that replaces posteriorgrams for spoken queries with sequences of state numbers for triphone hidden Markov models, omitting calculations of local distance. While that method greatly reduced retrieval times, it still required large amounts of memory to store speech data posteriorgrams. We therefore newly propose a method for reducing memory usage and retrieval times by compressing speech data posteriorgrams into sets of posterior probability vectors for each utterance, each speech document, or all speech data, rather than storing all posterior probability vectors for each frame of speech data. Evaluation experiments conducted using open test collections for the "SpokenDoc" tasks of the NTCIR-10 and NTCIR-12 workshops demonstrate memory usage reduction by the proposed method.**

Keywords— Spoken-term detection; Query by example; Average posterior probability vector

I. INTRODUCTION

The development and spread of cloud systems has increased opportunities for use of large multimedia datasets, including voice data. A retrieval function is required to find a specific scene in a large audio dataset. Studies of spoken-term detection (STD), the task of detecting a section where a search word (query) is spoken, have been actively conducted to realize such a function.

STD workshops have been held in Japan and overseas [3–5, 16–18], evaluating STD systems from viewpoints such as retrieval accuracy, retrieval time, required memory, and indexing time. With the spread of smart devices, research in recent years has extended to so-called query by example (QbE) and spoken-query STD, where queries are given by voice.

Posteriorgram matching is a representative method for realizing QbE. Posterior probabilities for each state in a triphone hidden Markov model (HMM) are generated at every frame by outputs from a deep neural network (DNN) whose inputs are features extracted from speech data and a spoken query. The posterior probabilities for all states are obtained at each frame as a posterior probability vector. Posteriorgrams consist of posterior probability vectors for speech data. While posteriorgram matching between posteriorgrams of spoken queries and speech data provides high retrieval accuracy, these methods also require long retrieval times and large amounts of memory [6]. This is because posterior provability vectors have several thousands of dimensions, and posteriorgrams require memory space for several thousands of posterior probability vectors for all frames across the entirety of speech data. Further, it requires long times to calculate inner products between two posterior probability vectors at every grid point in dynamic programming (DP) or dynamic time-warping matrixes to obtain local distances.

To reduce retrieval times, we previously proposed a method for determining the maximum likelihood state for each posterior probability vector of each frame in a posteriorgram [7]. The maximum likelihood state is associated with the state denoting the highest posterior probability among the posterior probability vector states. The method extracts a maximum likelihood state sequence (MLSS), namely, the maximum likelihood state sequence for a spoken query.

A MLSS does not require calculations for obtaining local distances, such as the inner product of two posterior probability vectors, and it refers to only the posterior probability in posteriorgrams at every grid point in a DP matrix. Use of a MLSS reduces retrieval times, but maintaining the retrieval accuracy of posteriorgram matching still consumes large amounts of memory, equivalent to the posteriorgram size.

The paper propose a novel method for reducing memory consumption by compressing speech-data posteriorgrams to an average posterior vector matrix of speech data instead of maintaining all speech-data posteriorgrams. The proposed method introduces three types of compression unit, namely, those for utterances, spoken documents, and all speech data. In the proposed method, speech data are first converted into a maximum likelihood state sequence, then a state number is assigned to each frame. Next, "category frames" indicating the same maximum likelihood state number for speech data are extracted. Posterior probability vectors are averaged for all sets of same-category frames. Posteriorgrams are then compressed to a set of averaged posterior probability matrixes for the appearing states. It is unnecessary to generate average posterior probability vectors for non-appearing states because these

states are not referred to in DP matching. Each posterior probability in the average posterior probability vector is converted to a local distance in the same manner as in the maximum likelihood state method. Average posterior probability matrixes are transformed to distance matrixes, which are constructed for each compression unit. For example, the number of distance matrixes equals the number of spoken documents when the compression unit is set to spoken documents. In DP matching, the local distance is only referred to in the distance matrixes for each compression unit of speech data. In this paper, this proposed method is called the average posterior probability vector (APPV) method. APPV reduces both retrieval times and memory usage.

The remainder of this paper is organized as follows. Section 2 describes the conventional STD system and acoustic distance. Section 3 describes the proposed acoustic distances in detail. Section 4 describes application of the "SpokenDoc" open test collections from the NTCIR-10 and NTCIR-12 workshops to evaluate retrieval accuracies in the proposed method, and Section 5 presents our conclusions.

II. Related Work

A. Posteriorgram Matching [6]

Using posteriorgram matching between a spoken query and speech data at the frame level in QbE generally provides high retrieval accuracy. Equation (1) gives the local distance between two posterior probability vectors for a spoken query and the speech data. CDP matching is performed calculating each distance for all DP grid points [11, 12].

$$Dis\left(\boldsymbol{P_q}(\boldsymbol{i}), \boldsymbol{P_d}(\boldsymbol{j})\right) = -log_{10}(dot(\boldsymbol{P_q}(\boldsymbol{i}), \boldsymbol{P_d}(j))) \quad (1)$$

Here, $\boldsymbol{P_q}$ is the posteriorgram of the spoken query, and $\boldsymbol{P_q}(\boldsymbol{i})$ is the posterior probability vector for the *i*-th frame. Similarly, $\boldsymbol{P_d}$ and $\boldsymbol{P_d}(\boldsymbol{j})$ are respectively the posteriorgram of the speech data and the posterior probability vector at the *j*-th frame. Because the inner product of $\boldsymbol{P_q}(\boldsymbol{i})$ and $\boldsymbol{P_d}(\boldsymbol{j})$ indicates similarity between the two vectors, its negative logarithm is used as the local distance. Calculation of this inner product requires a long retrieval time, because the posterior probability vector for $\boldsymbol{P_q}(\boldsymbol{i})$ and $\boldsymbol{P_d}(\boldsymbol{j})$ in Equation (1) is of several thousand dimensions. Storing speech data posteriorgrams thus also requires a large amount of memory.

B. Maximum Likelihood State Sequence

The MLSS method [7] was proposed to reduce retrieval times in posteriorgram matching. MLSS replaces posteriorgrams for spoken queries with sequences of maximum likelihood state numbers for each frame. This method does not require calculation of the inner product between posterior probability vectors for local distances, which are computed in advance. MLSS refers to only the distance matrix instead of posteriorgrams of speech data.

Figure 1 shows an image of a MLSS. Speech data are converted into posteriorgrams in advance. All posterior probabilities in posteriorgrams are converted into local distances by Eq. (2), and this distance matrix is stored instead of the speech data posteriorgrams. Spoken queries are converted to posteriorgrams (upper left in Fig. 1), and a maximum likelihood sequence (maximum likelihood state number sequence) is obtained from the posteriorgram of the spoken query (upper right in Fig. 1). Because the distance matrix is referred to only to obtain a local distance in DP matching, no calculation of local distances is required. For example, if the third frame of a query denotes state 1 in the figure, the distance of the second frame of speech data is obtained by referring to the first row in the distance matrix, circled in red.

$$Dis(\boldsymbol{i}, \boldsymbol{P_d}(\boldsymbol{j})) = -log_{10}(\boldsymbol{P_d}(i,j)) \quad (2)$$

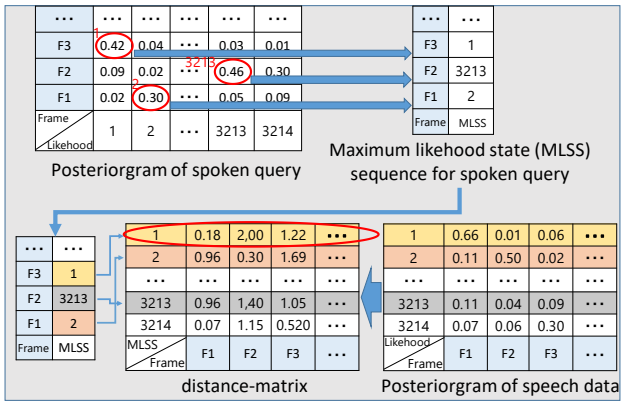

Fig. 1 Maximum likelihood state sequence image.

III. Average Posterior Probability Vector Compression

MLSS reduces retrieval times, but the retrieval accuracy is slightly deteriorated as compared with posteriorgram matching, and memory usage remains unchanged. We thus propose the average posterior probability vector (APPV) method, which reduces memory consumption by compressing speech data posteriorgrams for each speech data compression unit. As mentioned above, three types of compression unit are described in this paper: those for utterances, spoken documents, and the entirety of speech data. The following describes the proposed method in detail.

In the APPV method, the matching process is nearly identical to that under MLSS. Speech data are compressed for each compression unit. Figure 2 illustrates the process for constructing an APPV, taking all speech data as the compression unit. Speech data are converted to posteriorgrams (left part in Fig. 2), and each posteriorgram frame is assigned to the HMM state whose posterior probability indicates the highest posterior probability vector (bottom of left part). Each speech data frame is associated with the maximum likelihood state number. Because all frames are categorized by state

number, frames with same frame numbers are gathered for each speech data compression unit. Average posterior probabilities for each state are computed and average posterior probability vectors for frames with the same state number are obtained for each compression unit (right part in Fig. 2). The posteriorgram, therefore, can be compressed because posterior probability vectors in a compression unit are replaced with average posterior probability vectors. All posterior probabilities in APPV are converted into local distances by Eq. (3), and this distance matrix is stored instead of the speech data posteriorgrams. Here, $\boldsymbol{AP_d}$ is the APPV matrix of the speech data, and $\boldsymbol{AP_d(j)}$ is the average posterior probability vector for the *j*-th frame. For example, when taking a spoken document as the compression unit, the number of distance matrixes is the number of spoken documents. When the entirety of speech data is the compression unit, there is only one distance matrix. During CDP matching, local distances refer to only the distance matrixes of the average posterior probability vectors.

The required memory size is only 36 MB (4 bytes per dimension × 3,000 states × 3,000 dimensions per state) for a single distance matrix, because each state is an average posterior probability vector of 3,000 dimensions when the number of states is 3,000. Given 30 hr of speech data, the maximum likelihood sequence is 21 MB (2 bytes per dimension × 1 dimension per frame × 100 frames per sec × 30 hr × 3,600 sec per hr) to hold the state number, a total of 57 MB.

$$Dis(\boldsymbol{i}, \boldsymbol{AP_d(j)}) = -log_{10}(\boldsymbol{AP_d}(i,j)) \qquad (3)$$

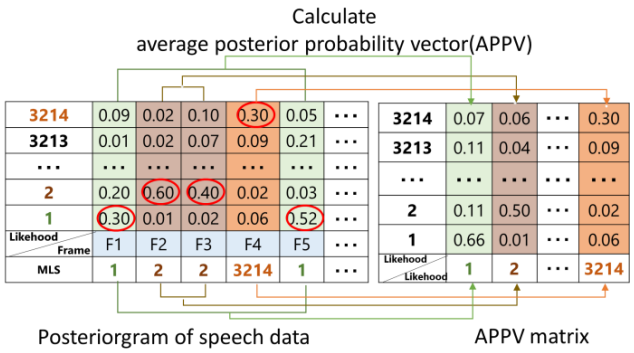

Fig. 2 Constructing an APPV.

IV. Evaluation Experiments

A. Experimental Conditions

We used 2,525 presentation speeches (about 560 hr of speech data) from the Corpus of Spontaneous Japanese [8], excluding 177 speeches, to train acoustic and language models. The input features were 83 dimensions, comprising 80 dimensions of filter bank features and 3 dimensions of pitch features. Table 1 shows the feature extraction conditions.

The ESPnet [9] shared encoder consists of 12 layers, 2,048 units, and 320 hidden units. 3,213-dimensional output probabilities corresponding to characters are generated from the output layer of the shared encoder and regarded as the feature vector. The connectionist temporal classification (CTC) of a hybrid CTC/attention-based end-to-end architecture receives the feature vector to the shared decoder and out puts, a posterior probability vector of 3,214 dimensions of characters and the blank symbols. We use posteriorgrams obtained from the CTC architecture for speech retrieval.

In preliminary experiments, we improved search accuracies by processing blanks so that their state numbers were not consecutive. Therefore, the maximum likelihood sequence is saved with blank portions with consecutive state numbers removed.

When computing retrieval times, we used an Intel Core i7-4770 CPU and 16 GB of RAM.

Table 1 Conditions for feature extraction.

Feature parameter	83 dimensions (FBANK, 80 dim + Pitch, 3 dim)
Window length	25 ms
Frame shift	10 ms
Sampling frequency	16 kHz
Number of quantization bits	16

Table 2 ESPnet learning requirements.

Learning library	PyTorch
Number of epochs	22
Batch size	8
Number of states	3,214

B. Test Sets

We used open test collections from NTCIR-10 and NTCIR-12 as test sets in the evaluation experiments. We constructed and recorded 100 spoken queries uttered by a total of ten students (five of each gender) because there were no spoken queries in the NTCIR-10 dataset. The NTCIR-12 dataset, however, provides 113 spoken queries uttered by ten persons. NTCIR-10 and NTCIR-12 respectively contain 104 Spoken Document Processing Workshop (SDPWS) presentation speeches (40,746 utterances over about 29 hr) and 98 SDPWS presentation speeches (37,782 utterances over about 29 hr). Mean average precision (MAP; the average of 10 APs × APs of the number of queries) was used as an evaluation index for retrieval accuracy.

Table 3 The two open test collections.

	NTCIR-10	NTCIR-12
Speech data	SDPWS, 104 presentations, 29 hr, 40,746 utterances	SDPWS, 98 presentations, 29 hr, 37,782 utterances
Queries	Formal run: 100 (10 persons)	Formal run: 113 (10 persons)

C. *Comparison with Former Research*

We conducted evaluation experiments to compare the proposed APPV method with MLSS under the conditions described above. In this section, all posteriorgrams for all speech data (104 SDPWS speeches and 98 SDPWS speeches) were compressed into a single average posterior probability vector matrix. Figure 3 shows the results, with bar graphs showing MAP and line graphs showing memory consumed.

MLSS retrieval accuracies for NTCIR-10 and NTCIR-12 were 81.72% and 81.54%, respectively. When compression was performed for all speech data, APPV retrieval accuracies for NTCIR-10 and NTCIR-12 were 71.66% and 70.20%, respectively, and the average MAP decreased by 10.69. However, MLSS required 30.03 GB and 28.08 GB of memory for NTCIR-10 and NTCIR-12, respectively, while APPV required only 18 MB for both. Required memory size was thus reduced by about 1/1,450. Under both MLSS and APPV, the retrieval time for both NTCIR-10 and NTCIR-12 was 0.9 sec.

The deterioration in retrieval accuracy under APPV is mainly due to the decreased amount of information in the speech data. Also, conversion errors from posteriorgrams of speech data to maximum likelihood state number sequences likely generate inaccurate average posterior probability vectors.

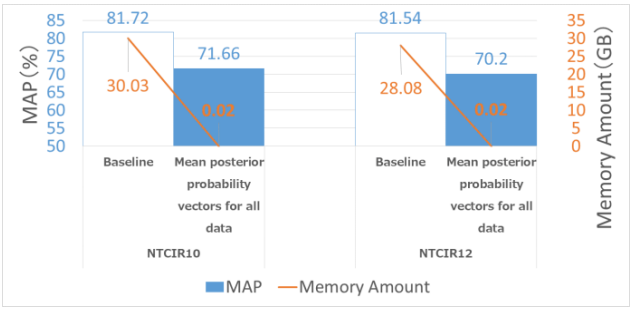

Fig. 3 Comparison of the previous and proposed methods.

D. *Evaluation Experiment According to a Compression Unit*

In Section IV.C, posteriorgrams for all speech data were compressed into one average posterior probability vector matrix. An average posterior probability vector matrix can be constructed for each presentation speech or each utterance. For example, all posteriorgrams for the NTCIR-10 SDPWS presentation speech data were compressed into a single APPV matrix. Because each posteriorgram of each presentation speech is compressed into a single APPV matrix, 104 APPV matrixes were created in this case. Similarly, each posteriorgram of each utterance is compressed into a single APPV matrix, so 40,746 matrixes are composed in this case. We also conducted experiments using the two other compression units for each presentation speech and each utterance. Figure 4 shows the experimental results with different compression units for each presentation speech and each utterance.

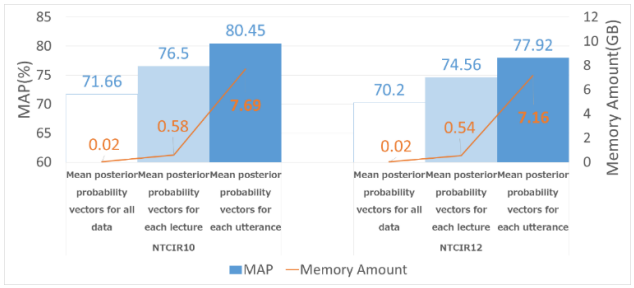

Fig. 4 Retrieval performance by compression unit for each presentation speech and each utterance.

Compared with the compression unit of all data, the required memory size (amount of information) increased in the order of each utterance, each presentation speech, and all presentation speeches, and the MAP retrieval accuracy improved with amount of information. At the compression unit of each utterance, the MAP deterioration was only 1.27 pts for NTCIR-10 and 3.62 pts for NTCIR-12, and the required amount of memory was reduced to about 1/4 for both NTCIR-10 and NTCIR-12 as compared to MLSS. Retrieval times were the same, 0.9 sec.

V. CONCLUSIONS

We proposed the APPV method, which reduces memory requirements by compressing QbE speech data posteriorgrams into average posterior probability vectors for all speech data, each presentation speech, or each utterance. Compared with MLSS on average for NTCIR-10 and NTCIR-12, the proposed method maintained the same retrieval time as that under MLSS while reducing memory consumption to about 1/1,450. However, the retrieval accuracy decreased by 10.69 pts. The retrieval accuracy at the compression unit of each utterance improved by 8.35 pts on average as compared with the case of using all speech data as the compression unit, and the required memory size was reduced to 1/4 of MLSS while suppressing MAP deterioration to within 2.5 pts. In future studies, we will investigate best clustering methods for constructing APPV.

ACKNOWLEDGMENT

This work was supported by JSPS (C), KAKENHI Grant Number 18K11358.

REFERENCES

[1] P. Motlicek, F. Valente, and PN. Garner, “English Spoken Term Detection in Multilingual Recordings,” INTERSPEECH 2010, pp.206-209, 2010.
[2] L. Mangu, H. Soltau, H.-K. Kuo, B. Kingsbury, and G. Saon, “Exploiting diversity for spoken term detection,” in Proc. ICASSP, 2013
[3] K. Kon'no, H. Saito, S. Narumi, K. Sugawara, K. Kamata, M. Kon'no, J. Takahashi and Y. Itoh, “An STD System for OOV Query Terms Integrating Multiple STD Results of Various Subword units, ” Proceedings of the 10th NTCIR Conference, 2013.
[4] National Institute of Informatics , NTCIR-12 , http://research.nii.ac.jp/ntcir/ntcir-12/index.html,
[5] Tomoyosi Akiba , Hiromitsu Nishizaki , Hiroaki Nanjo , Gareth J. F. Jones , ”Overview of the NTCIR-11 SpokenQuery&Doc Task,” Proceeding of the NTCIR-11 Conference, Tokyo, Japan, (2014)
[6] Masato Obara, Rescoring by Combination of Posteriorgram Score and Subword-Matching Score for Use in Query-by-Example, INTERSPEECH 2016, pp.1918-1922, 2016.
[7] Iwasaki Eitaro, Obara Masato, Kojima Kazunori, Lee Shi-wook, and Itoh Yoshiaki , “Query’s Maximum likelihood state sequence Using Posteriorgram in Query-by-Example,” ASJ pp.993-996, 2018-9.
[8] Tomoyosi Akiba et al, Overview of the NTCIR-10 SpokenDoc-2 Task, NTCIR-10 Workshop Meeting, pp.573-587, 2013
[9] Shinji Watanabe et al, “ESPnet: End-to-End Speech Processing Toolkit, Interspeech, ” 2018,pp2207-2211.
[10] Shinji Watanabe et al, “Hybrid CTC/attention architecture for end-to-end speech recognition, IEEE Journal of Selected Topics in Signal Processing, ” vol. 11, no. 8, pp. 1240–1253, 2017.
[11] T. Akiba, H. Nishizaki, H. Nanjo and G.J.F. Jones : Overview of NTCIR-12 Spoken&Doc Task, NTCIR-12, pp. 167-179, 2016.
[12] Timothy J. Hazen *et al.,* “Query-By-Example Spoken Term Detection Using Phonetic Posteriorgram Templates,” ASRU, 2009.
[13] Masato Obara, Munehiro Moriya, Ryota Konno, Kazunori Kojima , Kazuyo Tanaka , Shi-wook Lee and Yoshiaki Itoh, ”Acceleration for Query-by-Example Using Posteriorgram of Deep Neural Network,” Asia-Pacific Signal and Information Processing Association Annual Summit and Conference (APSIPA ASC) (2017)
[14] K. Maekawa, “Corpus of Spontaneous Japanese: Its design and evaluation Proceeding of the ISCA & IEEE Workshop on Spontaneous Speech Processing Association Annual Summit and Conference, ” APSIPA ASC, 2009
[15] Shi-wook Lee, Kazuyo Tanaka, Yoshiaki Itoh, “Effective Combination of Heterogeneous Subword-based Spoken Term Detection System,” 4 pages, IEEE Spoken Language Technology Workshop(SLT), 2014-12
[16] The Spoken Term Detection (STD) 2006 Evaluation Plan, National Institute of Standards and Technology (NIST), 2006, http://www.itl.nist.gov/iad/mig/tests/std/2006/.
[17] Draft KWS16 Keyword Search Evaluation Plan, 2016, http://www.nist.gov/itl/iad/mig/upload/KWS16-evalplan-v04.pdf
[18] The 2013 Spoken Web Search Task, MediaEval Benchmarking Initiative for Multimedia Evaluation, 2013, http://www.multimediaeval.org/mediaeval2013/sws2013.