Score Normalization using Phoneme-based Entropy for Spoken Term Detection

Hiromitsu Nishizaki* and Naoki Sawada†
* Faculty of Engineering, the Graduate School of Interdisciplinary Research,
† Department of Education, Interdisciplinary Graduate School of Medicine and Engineering,
University of Yamanashi, Kofu-shi, Yamanashi 400-8511 Japan
E-mail: {nisizaki, sawada}@alps-lab.org Tel: +81-55-220-8361

***Abstract*—This study investigates and demonstrates the effectiveness of utilizing the entropy of a query term in spoken term detection (STD) for score normalization. It is important to normalize scores of detected terms because the optimal threshold for the decision process of detected candidates is commonly set for all query terms. A query term with higher phoneme-based entropy rather than the average entropy value of a query set is probably difficult to correctly recognize using automatic speech recognition. Thus, it cannot be detected with high accuracy if the same threshold is set for all query terms. Therefore, we propose a score normalization method in which a calibrated matching score between a query term and an index made of target spoken documents is dynamically calculated using phoneme-based entropy of the query term on a dynamic time warping-based STD framework. We evaluated this framework with query entropy on an STD task. The result indicated that it worked quite well and significantly improved STD performance compared with the baseline STD system with a pooling-based evaluation framework.**

I. Introduction

Spoken term detection (STD) or open keyword search (KWS), one of speech data retrieval technologies, is designed to determine whether or not a given utterance includes a query term consisting of a word or phrase. STD research has become a hot topic in the spoken document processing research field, and the number of STD research reports is increasing in the wake of the 2006 STD evaluation organized by National Institute of Standards and Technology (NIST) [1]. In particular, recently, some kinds of test collections such as the NIST Open KWS tasks [2] and the MediaEval query-by-example search on speech task [3] have been released.

Numerous STD studies have been proposed [4], [5] and many STD systems have been developed. In recent times, combination techniques for term detection from such STD systems have been studied [6], [8]. When detection candidates from multiple STD systems are combined, the STD score calculation that determines the final output(s) of the query term should be considered. If the dynamics of the STD score of detection candidates generated by an STD system differ from the other STD systems, the combined systems are highly probable to fail. Therefore, score normalization (calibration) calculated by each STD system is very important.

Furthermore, it is important to consider score normalization when using only a single STD system. For example, the computer environment, e.g., CPU and memory capacity, should be considered [9]. All STD systems output detection candidates, each of which has a detection score for an input query term. To obtain the best STD performance for a query term, an optimal threshold that determines whether a candidate is accepted as a correct detection should be set. The optimal threshold value varies according to the query term. For example, generally, STD performance on evaluation measures, such as F-measure [10] and term-weighted value (TWV) [1], of a short query term is probable to be low because of false detected candidates (false alarms), which reduces precision. Therefore, the detection threshold should be set more strictly to reduce the number of false detections, i.e., the threshold value should be set relative to the input query term.

However, it is not necessary to set the threshold value for a query term dynamically to calibrate the STD score of each candidate. The score normalization allows us to use a common threshold value for all query terms. This study focuses on STD score normalization for a single STD system using phoneme-based entropy of a query term.

In previous studies, we proposed a confusion network (CN)-based indexing and a dynamic time warping (DTW)-based search engine [11]. The CN-based index, which we refer to as a phoneme transition network (PTN)-formed index [11], comprised 10 types of transcriptions generated by 10 different automatic speech recognition (ASR) systems, including a large vocabulary continuous speech recognition system and a phoneme ASR system. We demonstrated that the proposed method outperforms other STD technologies that participated in the 9th and 10th National Institute of Informatics Testbeds and Community for Information access Research (NTCIR-9 and NTCIR-10) project STD evaluation frameworks [10], [12]. DTW-based matching between a subword sequence of a query term and speech transcription demonstrates poor performance for speech recognition errors. Therefore, the STD performance of the DTW-based technique depends on the accuracy of subword-based transcriptions. Our proposed DTW-based approach using a PTN-formed index for STD was very robust for ASR errors.

However, this approach outputs a significant number of false detections because the structure of PTN was complex. These false detections degraded STD performance. In particular, it seemed difficult for high entropy-query terms to detect correct candidates when more stringent DTW score (cost)

thresholds[1] were set because ASR of high entropy-query terms is probably more difficult compared with lower entropy-query terms. Therefore, we expect that STD cost normalization that considers entropy of query terms in the cost calculation can uniform the threshold for accepting detected candidates and possibly improve STD performance.

Score normalization techniques for STD have been studied [6], [7], [8]. Mamou et al. [6] proposed a normalization method based on query length (duration) [13]. They also proposed a regression-based normalization method, i.e., a machine learning approach. The regression model used to calculate the STD score is trained using six types of features. Abad et al. [7] and Hout et al. [8] also investigated a linear logistic regression approach for score calibration.

In contrast to these regression model approaches, we investigated score normalization using phoneme-based entropy of a query term. The main idea of our proposed method is to normalize the STD score directly using an entropy value that depends on a query term. Our method is not a machine learning approach; therefore, it is not necessary to train or tune model parameters using much data for score normalization. The approach explored in this study was inspired by our previous study [14]. We attempted to use query entropy (**QE**) to filter falsely detected candidates in the second STD process. However, this approach did not perform well. Therefore, we propose directly embedding QE into the cost calculation of the DTW-based STD process. In this study, QE is calculated by a phoneme-based trigram language model; therefore, QE is nearly the same as the phoneme-based perplexity of a query term.

As previously mentioned, the entropy of query terms is an important factor for STD. However, to the best of our knowledge, no STD studies have considered entropy (perplexity) of a query term for an STD framework (including detection engine and score normalization) assembled in an STD engine. However, it should be noted that perplexity of a query term has been investigated [15]. Most STD studies have focused on utilizing information related to target spoken documents, e.g., acoustical similarities [16], [17], [18] and lattice-based score [15], [19], [20], [21], which are important parameters for STD. This study demonstrates the effectiveness of query entropy-based score normalization and improvement of STD performance. Our approach is quite novel, and the contribution of our study is to reveal the effectiveness of QE in a DTW-based STD framework.

We evaluated the proposed STD framework with QE on the NTCIR-10 SpokenDoc-2 moderate-size STD task. The results show that the proposed framework works quite well and significantly improves STD performance compared with a baseline STD system that did not consider QE. However, our approach does not affect the ranking of the detected candidates because the QE value is fixed for each query term. Therefore, it is particularly effective to determine an optimal threshold and STD cost (or score) normalization.

[1]Because the DTW scores equal to the distance between the query and PTN-formed index, the lower DTW score is better in the STD engine [11].

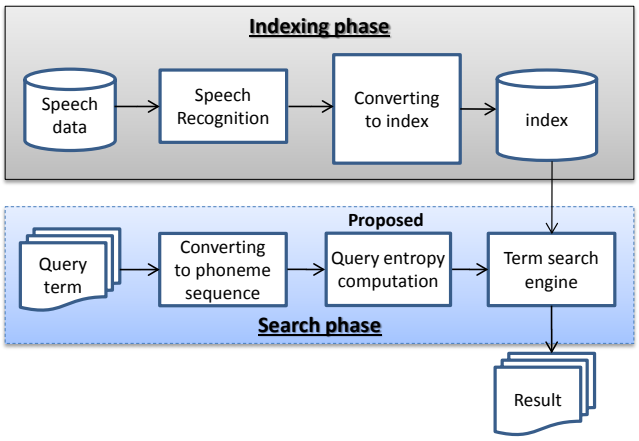

Fig. 1. Overview of our STD framework.

The remainder of this paper is organized as follows. A DTW-based STD using multiple ASR systems is described in Section II. Section III provides analysis of entropy of a query term and explains how to utilize it. Evaluation results are given in Section IV, and conclusions are presented in Section V.

II. STD USING MULTIPLE ASR SYSTEMS

This section explains the baseline STD engine using the DTW-based matching framework for calculating scores between a query term and a PTN-formed index.

Fig. 1 outlines the STD framework used in this study. In the indexing phase, target speech data is transcribed by multiple ASR systems, and their recognition outputs (word or sub-word sequences) are converted into the PTN-formed index for STD. In the search phase, the word-formed query is converted into a phoneme sequence. Next, the number of phonemes and their entropy are calculated. Finally, the phoneme-formed query is input to the term detection engine.

A. Phoneme transition network

Fig. 2 shows an example of the development of a PTN-formed index for "*cosine*" (Japanese pronunciation is /k o s a i N/) by aligning N phoneme sequences from the 1-best hypothesis of the ASR. We used 10 types of ASR systems to create the PTN-formed index. The speech was recognized by the 10 ASR systems to yield 10 hypotheses, which were then converted into phoneme sequences (Fig. 2). Next, we obtained "aligned sequences" using the dynamic programming (DP) scheme previously described [22]. Finally, the PTN was obtained by converting the aligned sequences. Here "@" in Fig. 2 indicates a null transition. Arcs between the nodes in the PTN have a few phonemes and null transitions with an occurrence probability. However, in this study, we did not consider any phoneme occurrence probabilities.

In this study, Julius ver. 4.1.3 [23], an open source decoder for ASR, is used in all systems. Acoustic models are triphone-based (Tri.) and syllable-based (Syl.) Hidden Markov Models

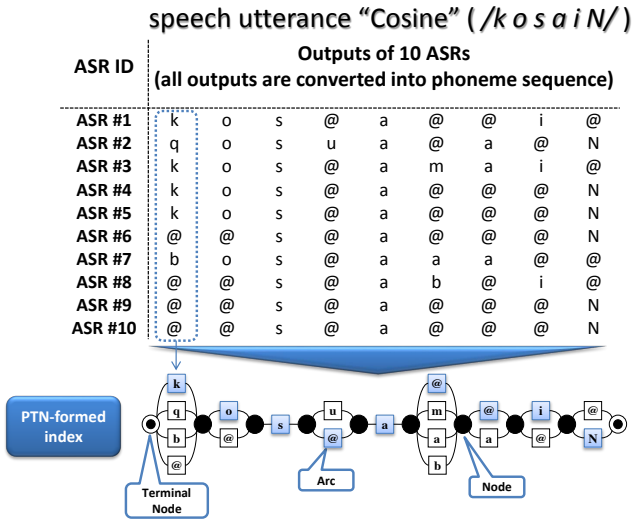

Fig. 2. Generating the PTN-formed index by performing alignment using DP and converting to PTN (quoted from [11]).

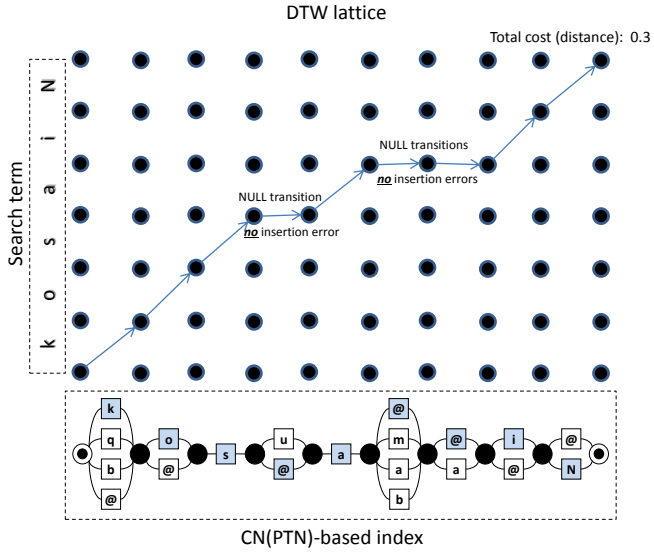

Fig. 3. Example of term search on PTN-formed index (quoted from [11]).

(HMMs), both of which are trained on spoken lectures from the Corpus of Spontaneous Japanese (CSJ) [24]. All language models are word- and character-based trigrams as follows:

- WBC : Word-based trigram where words are represented by a mix of Chinese characters and Japanese Hiragana and Katakana.
- WBH : Word-based trigram where all words are represented only by Japanese Hiragana. Words comprising Chinese characters and Japanese Katakana are converted into Hiragana sequences.
- CB : Character-based trigram where all characters are represented by Japanese Hiragana.
- BM : Character-sequence-based trigram where the unit of language modeling comprises two Japanese Hiragana characters.
- Non : No language model is used. Speech recognition without any language model is equivalent to phoneme (or syllable) recognition.

Each model was trained using CSJ transcriptions.

The training conditions of all acoustic and language models and the ASR dictionary are the same as in the STD/SDR test collections used in the NTCIR-9 [10] and NTCIR-10 [12] Workshops.

B. STD engine

The term detection engine uses the DTW-based word spotting method. Fig. 3 shows an example of the DTW framework between the search term "k o s a i N" (cosine) and the PTN-formed index. The PTN has multiple arcs between two adjoining nodes. These arcs are compared with the phoneme labels of a query term.

We use edit distance on the DTW paths as cost, and the costs for substitution, insertion, and deletion errors were set to 1.0 when the number of phonemes comprising a query term was N or greater. In contrast, the cost was set to 1.5 when the number of phonemes was less than N to avoid false term detections in query terms having fewer phonemes. This cost (=1.5) was optimized using a development query set.

Total cost $D(i,j)$ at the grid point (i,j) ($i = \{0,...,I\}$, $j = \{0,...,J\}$, where I and J are the number of the set of arcs in the index and query term, respectively) on the DTW lattice was calculated using the following equations:

$$D(i,j) = \min \begin{cases} D(i,j-1) + Del \\ D(i-1,j) + Null(i) \\ D(i-1,j-1) + \\ \quad Match(i,j) + Vot(i,j) \end{cases} \quad (1)$$

$$Match(i,j) = \begin{cases} 0.0 : Query(j) \in PTN(i) \\ 1.0 : Query(j) \notin PTN(i), J \geq N \\ 1.5 : Query(j) \notin PTN(i), J < N \end{cases} \quad (2)$$

$$Del = \begin{cases} 1.0 : J \geq N \\ 1.5 : J < N \end{cases} \quad (3)$$

$$Null(i) = \begin{cases} \frac{\alpha}{Voting(@)} : NULL \in PTN(i), J \geq N \\ \frac{\beta}{Voting(@)} : NULL \in PTN(i), J < N \\ 1.0 : NULL \notin PTN(i), J \geq N \\ 1.5 : NULL \notin PTN(i), J < N \end{cases} \quad (4)$$

where $PTN(i)$ is the set of phoneme labels of the arcs at the i-th node in the PTN and $Query(j)$ indicates the j-th phoneme label in the query term. We allowed a null transition between two nodes in the PTN-formed index with the cost defined by Eq.(4). When the query term matches null (@) in the PTN, a transition cost was set dinamically, as shown in Eq.(4). $Voting(@)$ indicates the number of ASR systems that output NULL at the same arc. We refer to this as "null voting." α and β are hyperparameters, optimized using the development set [10]. The appropriate null cost achieves increasing term detection and decreasing false detections.

"$Vot(i,j)$" in Eq.(1) is related to the false detection control parameters [11] and is calculated as follows:

$$Vot(i,j) = \begin{cases} \frac{\gamma}{Voting(p)} : \\ \exists p \in PTN(i), p = Query(j) \\ 0.0 : Query(j) \notin PTN(i) \end{cases} \quad (5)$$

We provided a parameter, $Voting(p)$, to control false detection. $Voting(p)$ is the number of ASR systems that output the same phoneme p at the same arc. Higher $Voting(p)$ values increase the reliability of phoneme p. γ, which was set to 0.5, is also a hyperparameter optimized by the development query set [10].

In advance searches for the query term, the term detection engine initializes $D(i,0) = 0$. Next, it calculates $D(i,j)$ using Eq.(1) ($i = \{0, ..., I\}$, $j = \{1, ..., J\}$). $D(i,J)$ is normalized[2] by the length of the DTW path.

After completing the calculation, the engine outputs the detection candidates, which have a normalized cost $D(i,J)$ below a threshold θ. Recall and precision rates for STD can be controlled by varying θ.

III. Entropy of Query Term

A. Computation of entropy

QE is calculated on the basis of a phoneme sequence of a query term using a phoneme-based trigram language model. The trigram model is trained using the 2,525 lecture speeches in the CSJ. Equations to compute the entropy of a query term, which is represented as a phoneme sequence $p_0, p_1, \cdots, p_{N-1}, p_N$, are expressed as follows:

$$Entropy(Q) = -log_2 \left\{ \sum_{i=0}^{N} P(p_i | p_{i-2}, p_{i-1}) \right\} \quad (6)$$

$$N\_Entropy(Q) = \frac{1}{N} Entropy(Q) \quad (7)$$

$Entropy(Q)$ is the QE of Q, and $N\_Entropy(Q)$, i.e., the normalized QE (**NQE**), is normalized by the number of phonemes (N) of Q. In this study, we use QE and NQE values for calibrating an STD cost and will compare them on the STD tasks.

When the NQE value of Q is higher than the average NQE value for a query set, it is probably difficult to recognize it correctly using a phoneme-based ASR framework. In contrast, Q with a lower NQE value may be easy to recognize. Therefore, an entropy value of a query term affects ASR performance of the query terms. In such a case, it also influences STD performance.

B. Analysis of entropy

First, we compute the QE values of the query sets of the NTCIR-10 large-size and the moderate-size task test collections. Both query sets have 100 query terms. TABLE I shows the average QE values for each query category. We classify query terms into four categories: number of phonemes (N) is equal to 10 or greater, N is less than 10, in-vocabulary (INV), and out-of-vocabulary (OOV)[3]. In fact, the average QE value for the "$N < 10$" set on the two test collections is lower than the "$N \geq 10$" set; however, the average NQE value for the "$N < 10$" set is higher. Furthermore, there are no differences between the INV and the OOV query terms on phoneme-level entropy.

TABLE I
AVERAGE QUERY ENTROPY VALUES FOR EACH QUERY CATEGORY.

category	NTCIR-10 large-size task	
	QE	NQE
ALL	42.2154	3.4614
$N \geq 10$	47.0417	3.4332
$N < 10$	28.4789	3.5419
INV	42.1666	3.4445
OOV	42.2553	3.4753
	NTCIR-10 moderate-size task	
ALL	38.6903	3.5058
$N \geq 10$	47.9468	3.4605
$N < 10$	26.9093	3.5635
INV	36.3300	3.3211
OOV	40.7834	3.6696

TABLE II
STD PERFORMANCE FOR THE HIGH-NQE SET AND THE LOW-NQE SET.

	NTCIR-10 large-size task			
	Recall	Precision	F-measure	MAP
ALL	0.399	0.741	0.470	0.677
High-NQE	0.343	0.653	0.407	0.646
Low-NQE	0.448	0.819	0.525	0.705
	NTCIR-10 moderate-size task			
ALL	0.380	0.599	0.405	0.583
High-NQE	0.275	0.486	0.304	0.518
Low-NQE	0.462	0.688	0.484	0.633

Next, we perform STD [25] for each test collection. TABLE II shows the STD performances evaluated by recall, precision, F-measure, and mean average precision (MAP) values [10]. Each query set is classified into two groups based on ALL-NQE in TABLE I. One is the high-NQE set, and the other is the low-NQE set[4]. TABLE II shows that the low-NQE terms are significantly better than the high-NQE terms. Therefore, it is expected that entropy of a query term is effective for STD.

C. Entropy-based parameter for DTW matching using QE

We attempted to generate an entropy-based parameter for DTW using QE or NQE of a query term, which is then embedded into the computation of DTW cost between the query term and a PTN-formed index.

Assume that B_{QE} and B_{NQE} are the reference values of QE and NQE. B_{QE} and B_{NQE} are calculated by averaging the QE and NQE values of all query terms in a query set, respectively. An entropy-based parameter Q is defined as $B_{\mathrm{QE}}/\mathrm{QE}$ or $B_{\mathrm{NQE}}/\mathrm{NQE}$. It is applied to Eqs. (4) and (5). In this study, we simply just multiply Q by the false detection

[2]This normalization is not a proposed technique. It is a general normalization procedure used in many STD systems.

[3]The INV or OOV query term decision is based on the WBC language model.

[4]For example, in the large-size task, a query term with NQE of 3.47 belongs to the high-NQE set.

control parameters in Eqs.(4) and (5), as follows:

$$Null(i) = Q \times \begin{cases} \frac{\alpha}{Voting(@)} : NULL \in PTN(i), J \geq N \\ \frac{\beta}{Voting(@)} : NULL \in PTN(i), J < N \\ 1.0 : NULL \notin PTN(i), J \geq N \\ 1.5 : NULL \notin PTN(i), J < N \end{cases} \quad (8)$$

$$Vot(i,j) = Q \times \begin{cases} \frac{\gamma}{Voting(p)} : \\ \exists p \in PTN(i), p = Query(j) \\ 0.0 : Query(j) \notin PTN(i) \end{cases} \quad (9)$$

In our STD system, a query with low entropy value can be detected with better STD detection score (lower STD cost) rather than one with high entropy value. The smaller a query's entropy value becomes, the larger its query-based parameter Q becomes. In contrast, The larger a query's entropy value becomes, the smaller Q gets. Therefore, the Q parameter can adjust the STD costs of detected terms for a query. In other words, using the Q parameter makes the differences between the detection costs for all the queries. Finally, we can adopt the common detection threshold θ for all the queries.

Incidentally, we tried to investigate how to apply the Q parameter to the DTW cost calculation on the development set. As the result, the Eqs.(8) and (9) got the best performance on the development set.

IV. Evaluation

A. Experimental setup

We used the moderate-size task in NTCIR-10 SpokenDoc-2 [12] as an STD task for evaluation. The evaluation speech data is the Corpus of Spoken Document Processing Workshop. It comprises recordings of the first to sixth annual Spoken Document Processing Workshop (104 oral presentations, 28.6 hours). The word error rate of the presentations is 36.9%.

The number of query terms is 100, where 47 of the query terms are INV queries included in the ASR dictionary of the WBC language model, and the other 53 queries are OOV. Note that there are 444 and 456 occurrences of INV and OOV query items in the whole presentations of SDPWS, respectively.

We used two query sets to set B_{QE} and B_{NQE} values. One is the query set from the NTCIR-10 large-size task ("open" condition), and the other is the query set used in the STD evaluation ("closed" condition). The query terms of the large-size task are quite different from the terms of the moderate-size task. Furthermore, parameters (α, β, and γ) in the DTW-based STD engine are tuned using the test collection[5] for STD in NTCIR-9 [10].

The evaluation metrics used in this study are recall, precision, F-measure, and MAP [10]. These measurements are frequently used to evaluate information retrieval performance. F-measure values for the optimal balance of recall and precision values are denoted by "maximum F-measure" in the evaluation graphs. In this study, recall, precision rates, and F-measure values are calculated as micro-averages. MAP is a macro-average measure for an STD query set.

[5]This is the development set.

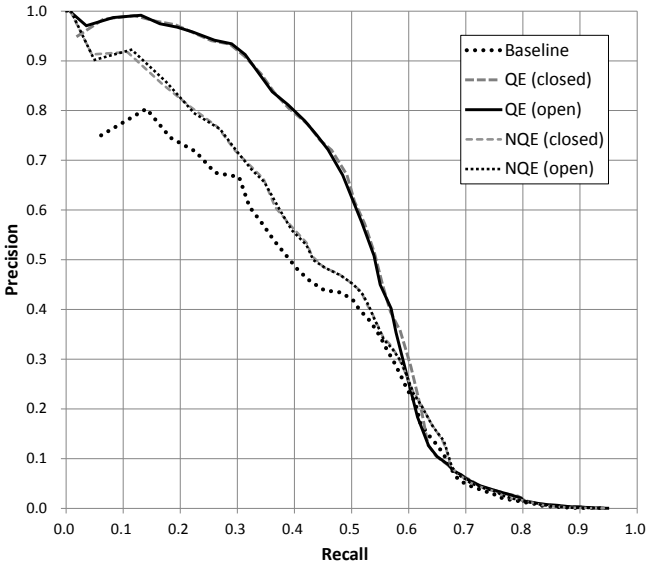

Fig. 4. Recall-precision curves of each STD system.

TABLE III
F-measure and MAP values.

	maximum F.	MAP
Baseline	0.457	0.605
QE (closed)	0.568	0.594
QE (open)	0.562	0.591
NQE (closed)	0.475	0.603
NQE (open)	0.475	0.603

The STD performance for the query sets can be illustrated by a recall-precision curve, which is plotted by changing the threshold θ value on the STD costs of detected candidates by each STD method. The θ value is the same for all query terms.

B. Experimental result

Fig. 4 shows the recall-precision curves for the baseline system without score normalization and the STD systems with score normalization using four types of entropy-based parameters. TABLE III also shows maximum F-measure and MAP values. The baseline system does not use any entropy-based parameters. We demonstrate the effectiveness of the four types of parameters for the STD task.

As shown in Fig. 4, all STD systems with score normalization by entropy-based parameters outperformed the recall-precision curve of the baseline because the score normalization method worked well. In particular, the QE-based parameters improved STD performance significantly because QE values consider the length of a query term, whereas NQE values do not. Furthermore, with regard to entropy-based parameter computation, no difference between the closed condition and the open condition was found. Thus, calculating reference entropy values (B_{QE} and B_{NQE}) using an open term set is not problematic.

Therefore, we confidently conclude that using query entropy-based parameters for score normalization in a DTW-based STD framework has the positive impact on STD per-

formance. The normalization allowed a common threshold to be set for all query terms. The micro-level evaluation that considered all detected candidates of all query terms showed improved performance.

As shown in TABLE III, the MAP values for the STD systems with entropy are the same as the baseline MAP values because the QE or NQE value is fixed for each query term. Therefore, the detected candidates ranked in lower matching cost-order are nearly the same as those output by the baseline system. In particular, the introduction of an entropy-based parameter is effective to determine the optimal threshold, which is used to narrow the candidates and normalize STD cost (or score).

V. Conclusion

We have proposed STD score normalization of the DTW-based STD method using the entropy of a query term. First, we showed that the QE value strongly affected STD performance on the STD tasks. Next, we explained how to embed QE or NQE of a query term into the DTW-based STD framework for score normalization. The experimental results for the NTCIR-10 STD task showed that both QE- or NQE-based parameters improved STD performance because the normalization technique enabled a common threshold for all query terms. In particular, the STD system with the QE-based parameter significantly outperformed the system with NQE because the number of phonemes was considered.

In future works, we plan to show that QE or NQE-based features work well in a machine learning framework for an STD task. Furthermore, we would like to demonstrate the effectiveness of the score normalization technique for a combination of multiple STD systems.

VI. Acknowledgements

This work was supported by JSPS KAKENHI Grant-in-Aid for Scientific Research (B) Grant Number 26282049 and Grant-in-Aid for Scientific Research (C) Grants Number 24500225 and 15K00254.

References

[1] The Spoken Term Detection (STD) 2006 evaluation plan (2006). http://www.itl.nist.gov/iad/mig/tests/std/2006/docs/std06-evalplan-v10.pdf (visited on 20th/June/2015).

[2] NIST, "KWS15 Keyword Seach Evaluation Plan ver.5," http://www.nist.gov/itl/iad/mig/upload/KWS15-evalplan-v05.pdf (visited on 21th/Sept./2015)

[3] X. Anguera, L.J. Rodriguez-Fuentes, I. Szöke, A. Buzo, F. Metze, and M. Penagarikano, "Query-by-Example Spoken Term Detection Evaluation on Low-resource Languages," in *Proceedings of the International Workshop on Spoken Language Technologies for Underresourced Languages (SLTU)*, pp.24–31, 2014.

[4] D. Vergyri, I. Shafran, A. Stolcke, R. R. Gadde, M. Akbacak, B. Roark, and W. Wang, "The SRI/OGI 2006 spoken term detection system," in *Proceedings of the 8th Annual Conference of the International Speech Communication Association (INTERSPEECH2007)*, 2007, pp. 2393–2396.

[5] S. Meng, J. Shao, R. P. Yu, J. Liu, and F. Seide, "Addressing the out-of-vocabulary problem for large-scale chinese spoken term detection," in *Proceedings of the 9th Annual Conference of the International Speech Communication Association (INTERSPEECH2008)*, 2008, pp. 2146–2149.

[6] J. Mamou, J. Cui, X. Cui, M. J. F. Gales, B. Kingsbury, K. Knill, L. Mangu, D. Nolden, M. Picheny, B. Ramabhadran, R. Schlüter, A. Sethy, P. C. Woodland, "System Combination and Score Normalization for Spoken Term Detection," in *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP2013)*, 2013, pp. 8272–8276.

[7] A. Abad, L. J. Rodriguez-Fuentes, M. Penagarikano, A. Varona, G. Bordel, "On the Calibration and Fusion of Heterogeneous Spoken Term Detection Systems," in *Proceedings of the 14th Annual Conference of the International Speech Communication Association (INTERSPEECH2013)*, 2013, pp. 20–24.

[8] J. van Hout, L. Ferrer, D. Vergyri, N. Scheffer, Y. Lei, V. Mitra, S. Wegmann, "Calibration and Multiple System Fusion for Spoken Term Detection Using Linear Logistic Regression," in *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP2014)*, 2014, pp. 7188–7192.

[9] D. R. H. Miller, M. Kleber, C. Kao, O. Kimball, T. Colthurst, S. A. Lowe, R. M. Schwartz, H. Gish, "Rapid and Accurate Spoken Term Detection," in *Proceedings of the 8th Annual Conference of the International Speech Communication Association (INTERSPEECH2007)*, 2007, pp. 314–317.

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

[11] S. Natori, Y. Furuya, H. Nishizaki, and Y. Sekiguchi, "Spoken term detection using phoneme transition network from multiple speech recognizers' outputs," *Journal of Information Processing*, vol. 21, no. 2, pp. 176–185, 2013.

[12] T. Akiba, H. Nishizaki, K. Aikawa, X. Hu, Y. Itoh, T. Kawahara, S. Nakagawa, H. Nanjo, and Y. Yamanashita, "Overview of the NTCIR-10 SpokenDoc-2 Task," in *Proceedings of the 10th NTCIR Conference*, 2013, pp. 573–587.

[13] J. Mamou, B. Ramabhadran, O. Siohan, "Vocabulary independent spoken term detection," in *Proceedings of the 30th Annual International ACM SIGIR conference*, vol.23, 2007, pp. 615–622.

[14] S. Natori, Y. Furuya, H. Nishizak, and Y. Sekiguchi, "Entropy-based false detection filtering in spoken term detection tasks," in *Proceedings of the 5th Asia-Pacific Signal and Information Processing Association Annual Summit and Conference (APSIPA ASC 2013)*, 2013, pp. 1–7.

[15] D. Wang, S. King, J. rankel, R. Vipperla, N. Evans, and R. Troncy, "Direct posterior confidence for out-of-vocabulary spoken term detection," *ACM Transactions on Information Systems*, vol. 30, no. 3, pp. 16:1–16:34, 2012.

[16] H. Wang, C.-C. Leung, T. Lee, B. Ma, and H. Li, "An Acoustic Segment Modeling Approach to Query-by-Examble Spoken Term Detection," in *Proceedings of The IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP2012)*, 2012, pp. 5157–5160.

[17] S. Narumi, K. Konno, T. Nakano, Y. Itoh, K. Kojima, M. Ishigame, K. Tanaka, and S. wook Lee, "Intensive acoustic models constructed by integrating low-occurrence models for spoken term detection," in *Proceedings of the 14th Annual Conference of the International Speech Communication Association (INTERSPEECH2013)*, 2013, pp. 25–28.

[18] S.-R. Shiang, P.-W. Chou, and L.-C. Yu, "Spoken Term Detection and Spoken Content Retrieval: Evaluations on NTCIR-11 SpokenQuery&Doc Task," in *Proceedings of the 11th NTCIR Conference*, 2014, pp. 371–375.

[19] S. Meng, P. Yu, F. Seide, and J. Liu, "A study of lattice-based spoken term detection for chinese spontaneous speech," in *Proceedings of the IEEE Workshop on Automatic Speech Recognition and Understanding (ASRU)*, 2007, pp. 635–640.

[20] D. Can and M. Saraçlar, "Lattice Indexing for Spoken Term Detection," *IEEE Trans. on Audio, Speech and Language Processing*, vol. 19, no. 8, pp. 2338–2347, 2011.

[21] H. yi Lee, Y. Zhang, E. Chuangsuwanich, and J. Glass, "Graph-based re-ranking using acoustic feature similarity between search results for spoken term detection on low-resource languages," in *Proceedings of the 15th Annual Conference of the International Speech Communication Association (INTERSPEECH2014)*, 2014, pp. 2479–2483.

[22] J. G. Fiscus, "A Post-processing System to Yield Reduced Word Error Rates: Recognizer Output Voting Error Reduction (ROVER)," in *Proceedings of the 1997 IEEE Workshop on Automatic Speech Recognition adn Understanding (ASRU'97)*, 1997, pp. 347–354.

[23] A. Lee and T. Kawahara, "Recent Development of Open-Source Speech Recognition Engine Julius," in *Proceedings of the 1st Asia-Pacific Signal and Information Processing Association Annual Summit and Conference (APSIPA ASC2009)*, 2009, pp. 1–6.
[24] K. Maekawa, "Corpus of Spontaneous Japanese: Its Design and Evaluation," in *Proceedings of the ISCA & IEEE Workshop on Spontaneous Speech Processing and Recognition*, 2003, pp. 1–8.
[25] S. Natori, Y. Furuya, H. Nishizaki, and Y. Sekiguchi, "Spoken Term Detection Using Phoneme Transition Network from Multiple Speech Recognizers' Outputs," *Journal of Information Processing*, vol. 21, no. 2, pp. 176–185, 2013.