AN ACOUSTIC SEGMENT MODELING APPROACH TO QUERY-BY-EXAMPLE SPOKEN TERM DETECTION

Haipeng Wang[†]*, Cheung-Chi Leung*[*]*, Tan Lee*[†]*, Bin Ma*[*]*, Haizhou Li*[*]

[†]Department of Electronic Engineering, The Chinese University of Hong Kong, Hong Kong
[*]Institute for Infocomm Research, A*STAR, Singapore

[†]{hpwang,tanlee}@ee.cuhk.edu.hk [*]{ccleung,mabin,hli}@i2r.a-star.edu.sg

ABSTRACT

The framework of posteriorgram-based template matching has been shown to be successful for query-by-example spoken term detection (STD). This framework employs a tokenizer to convert query examples and test utterances into frame-level posteriorgrams, and applies dynamic time warping to match the query posteriorgrams with test posteriorgrams to locate possible occurrences of the query term. It is not trivial to design a reliable tokenizer due to heterogeneous test conditions and the limitation of training resources. This paper presents a study of using acoustic segment models (ASMs) as the tokenizer. ASMs can be obtained following an unsupervised iterative procedure without any training transcriptions. The STD performance of the ASM tokenizer is evaluated on Fisher Corpus with comparison to three alternative tokenizers. Experimental results show that the ASM tokenizer outperforms a conventional GMM tokenizer and a language-mismatched phoneme recognizer. In addition, the performance is significantly improved by applying unsupervised speaker normalization techniques.

Index Terms— Spoken term detection, query-by-example, acoustic segment model, posteriorgram-based template matching

1. INTRODUCTION

Spoken term detection (STD) refers to the task of automatically locating the occurrences of a specified query term in a large audio archive. The query term may contain a single word or a sequence of words. It can be given in the form of orthographic representations or query utterance examples [1]. The latter case is known as query-by-example (QbyE) STD. STD technology is useful in various applications, such as multimedia information retrieval, personal entertainment, surveillance and security.

State-of-the-art STD systems [2] usually employ sophisticated large vocabulary continuous speech recognition (LVCSR) engines to convert speech utterances into textual representations, such as one-best sequences or lattices. The detection of a query term then becomes a problem of string matching. While this kind of systems provide high detection accuracy, a reliable and robust LVCSR system is not always available. It requires a substantial effort of software development and a large amount of transcribed speech and text data, which can be satisfied in only a few tasks with plenty of resources. Moreover, the detection performance is largely affected by the vocabulary coverage. If a query term contains out-of-vocabulary (OOV) words, the detection would fail.

Recently, a posteriorgram-based template matching framework was proposed for QbyE STD [3]. This framework represents speech segments by phoneme posteriorgrams, and matches query posteriorgrams with test posteriorgrams using the conventional dynamic time warping (DTW) method, which has been used in template-based speech recognition. This framework totally gets rid of the limitation of vocabulary coverage, and thereby OOV is no longer an issue in this framework. However training a phoneme recognizer requires at least hours of labeled speech data. This is not feasible for many languages or dialects for which resources have never been developed. In these cases, an intuitive approach is to utilize a phoneme recognizer of another resource-rich language. This approach is based on the assumption that the acoustic representations of some phonemes are similar across languages. However the performance of this approach inevitably suffers from the mismatches in language and speaking styles. Recently several studies have contributed to unsupervised STD, which emphasizes on unsupervised training of the tokenizer without requiring any training transcriptions. For example, Gaussian mixture model (GMM) is proposed to replace the phoneme recognizer in [4]. With unsupervised training, the mismatch between the training data and test data can be significantly alleviated because even the test data can be involved in the training process if necessary.

In this paper, we investigate the use of acoustic segment models (ASMs) as the tokenizer in the posteriorgram-based template matching framework. ASMs are a set of self-organized sound units which are intended to cover the overall speech characteristics of available training data [5]. ASMs are obtained by an unsupervised iterative training procedure without training transcriptions. Compared with GMM tokenizers, ASM tokenizers have two advantages. First, GMM training assumes independence among different speech frames, while ASM training groups similar neighboring frames into small segments and then establish models for these segments. In other words, ASM training takes advantage of the temporal information of speech, which is ignored in GMM training. The usefulness of temporal structure in speech processing has been proved and exploited [6]. Second, GMM tokenizers represent each sound unit by a single Gaussian component, while ASM tokenizers represent each sound unit by an HMM, which is more accurate and flexible for modeling the distribution of speech data. Similar ideas of applying self-organized sound units for STD have been proposed in [7] and [8]. Our work differentiates from them by incorporating an ASM tokenizer into the posteriorgram-based template matching framework. Within this framework, we made comparisons between the ASM tokenizer and other tokenizers including a GMM tokenizer, a well-trained phoneme recognizer, as well as a language-mismatched phoneme recognizer. In addition, our experimental results show that the performance of the ASM tokenizer can be significantly improved by applying unsupervised speaker normalization techniques, and that a further performance improvement can be obtained by combining the ASM tokenizer and the phoneme recognizers.

The rest of this paper is organized as follows. Section 2 briefly introduces the posteriorgram-based template matching framework for QbyE STD. Section 3 presents the details of training ASM and using the ASM tokenizer in the template matching framework. Experimental setup and results are presented in Section 4. We conclude and discuss the future work in Section 5.

2. POSTERIORGRAM-BASED TEMPLATE MATCHING FRAMEWORK

Figure 1 depicts the general framework of posteriorgram-based template matching for QbyE STD. A tokenizer is first obtained from training data. If the training data is given with training transcriptions, this tokenizer is referred to as *supervised*, otherwise as *unsupervised*. Using this tokenizer, query examples and test utterances are converted into posteriorgrams. DTW is then applied to scan through the test posteriorgrams and determine the best-matching region, which has the smallest distortion with respect to the query posteriorgrams.

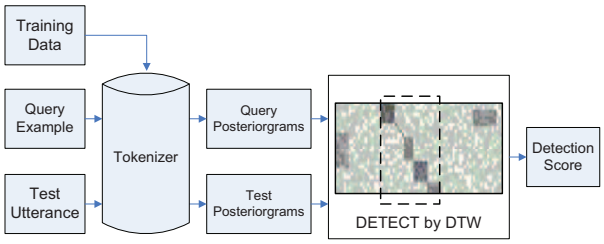

Fig. 1. Posteriorgram-based Template Matching Framework

2.1. Posteriorgram Representation

Given an observation feature vector o_t, its posteriorgram PG_{o_t} reflects the posterior distribution over a set of K predefined classes $\{C_1, C_2, ..., C_K\}$:

$$PG_{o_t} = [p(C_1|o_t), p(C_2|o_t), ..., p(C_K|o_t)]^T \quad (1)$$

where $p(C_i|o_t)$ is the posterior probability of the i^{th} class. The so-called *class* here could be any kind of sound unit, such as phoneme, Gaussian component, as well as ASM unit. Posteriorgrams can also be utilized as a kind of features, known as posterior features. Compared to standard spectral features, posteriorgrams are considered to be more robust towards session and speaker variabilities [9]. The posterior probabilities can be directly calculated according to Bayes' theorem using likelihood estimation, or be modeled by discriminative classifiers, such as MLP. Alternatively, if using an HMM based tokenizer, posterior probabilities can be derived from the decoding lattices.

2.2. Detection by Dynamic Time Warping

Dynamic Time Warping (DTW) has been widely applied to solve temporal sequence matching problems. Given two sequences of feature vectors, DTW finds the alignment path which minimizes the global warping distortion score by dynamic programming. For keyword detection, some variants of DTW are usually used to determine which region in the test utterance provides the minimum distortion from the query example. We use segmental DTW [4] in our implementation of STD system. In this study, three computationally-efficient distance metrics are considered as the DTW local distances between two posteriorgrams u and v:

- Inner Product Distance

$$D_{IP} = -\log(u^T v) \quad (2)$$

  Inner Product Distance measures the probability that u and v are generated from the same underlying sound unit [3].

- Cosine Distance

$$D_{Cos} = -\log(\frac{u^T v}{|u||v|}) \quad (3)$$

  As a geometry metric, Cosine distance measures the orthogonality of u and v.

- Bhattacharyya Distance

$$D_{Bhatt} = -\log(\sum_{k=1}^{K} \sqrt{u_k v_k}) \quad (4)$$

  Bhattacharyya distance measures the contribution of individual elements by the square-root operator.

Note that all of the above distance metrics are defined in the log probability scale.

3. ACOUSTIC SEGMENT MODELING APPROACH

The approach of acoustic segment modeling aims to establish a speech tokenizer in an unsupervised manner, and apply the ASM tokenizer for QbyE STD. We present the details of ASM training in three steps, and then introduce our approach of using the ASM tokenizer in the posteriorgram-based template matching framework.

3.1. Initial Segmentation

Given the observation sequence $O = [o_1, o_2, ..., o_T]$, consecutive observations are grouped into segments according to a criterion that minimizes the following distortion:

$$D(O, S) = \sum_{s=1}^{S} \sum_{i=b_{s-1}+1}^{b_s} d(o_i, c_s), \quad (5)$$

where c_s is the centroid of the s^{th} segment, S is the number of segments, $d(o_i, c_s)$ denotes the local distortion between o_i and c_s, and $b_{s-1}+1$ and b_s are the beginning and ending observation indices of the s^{th} segment. In this study, the local distortion $d(o_i, c_s)$ is set to be Euclidean distance, and the agglomerative clustering method [10] is adopted to determine the segment boundaries. Because the number of segments is not known beforehand, a threshold is pre-set for the total distortion. In addition, a segment duration constraint is employed to prevent a segment from lasting for too long or too short. The distortion threshold and duration constraint are adjusted to balance the precision and the recall rates of the identified segment boundaries on the development set.

3.2. Segment Labeling

After initial segmentation for all training utterances, a label is assigned to each resulted speech segment. In previous studies, this was usually done by vector quantization (VQ) [11] or K-means clustering [5]. Our work uses a different strategy based on GMM tokenization. A GMM is first trained from all training data. The number of Gaussian components is set to be the desired number of ASM units, which is empirically determined. For each segment, we label it with the index of the Gaussian component which provides the

highest likelihood of the speech segment. We design this strategy for several reasons. First, each Gaussian component can be interpreted as a broad acoustic class [12], which is in line with the aim of ASM training. Second, GMM has been experimentally proven to be useful as an alternative to phoneme recognizer [4, 13].

3.3. Iterative HMM Training

The initialized segment boundaries and labels are regarded as initial training transcriptions, with which ASMs are trained using an iterative procedure described as follows:

1. Train an HMM model for each ASM unit based on the initial transcriptions;
2. Use the existing HMMs to decode all training utterances. The resulted ASM unit sequences are taken as new transcriptions.
3. Re-train the HMMs with the new transcriptions.
4. Repeat Step 2 and Step 3 until convergence.

This procedure jointly optimizes segmentation and model estimation. In our experiments, it converges typically with only a few iterations. The convergence criteria can be specified based on the estimated likelihood of training data or recognition accuracy as in [5]. Since our focus is on STD, we determine the iteration number according to STD performance on the development set.

3.4. Applying ASM for Query-by-Example STD

We apply ASMs as the tokenizer in the posteriorgram-based template matching framework. ASMs share similar structure as conventional acoustic models in a phoneme recognizer. To transcribe a speech utterance into posteriorgrams using the ASM tokenizer, we first decode the utterance into ASM unit lattice, and then compute frame-level posteriorgrams from the ASM unit lattice. Since the formation of ASMs only depends on acoustic similarities, it would be quite sensitive to non-linguistic factors, such as the characteristics of speakers or the influence of environment. We alleviate this problem by conventional speaker normalization techniques. Similar to unsupervised speaker adaptation methods in speech recognition, the decoding sequence of each utterance is used as the *true* transcription. With these transcriptions and ASMs, we can perform two feature-level speaker normalization techniques: VTLN and CMLLR [14]. Since we assume no prior knowledge about the speaker information of the test archive, both VTLN and CMLLR are applied on a per utterance basis. For VTLN, the warping factor is determined by a maximum-likelihood grid search from 0.80 to 1.25. For CMLLR, a global transform is estimated for each utterance.

4. EXPERIMENTS

4.1. Experimental Setup

Within the framework of posteriorgram-based template matching, we evaluate the QbyE STD performance of the ASM tokenizer with comparison to three alternative tokenizers. The test archive, which is chosen from a part of Fisher English Corpus[1], contains 4638 utterances with the duration ranging from 5 seconds to 15 seconds for each utterance. 50 words are selected as query terms. For each query term, 10 spoken examples are manually extracted from another part of Fisher Corpus. The length of the query terms is from 4 phonemes to 10 phonemes. The prior probabilities of the query terms occurring in a test utterance vary from 0.30% to 1.32%. Similar to the experimental configuration of [4], we use the TIMIT Corpus[2] as the development set to adjust system parameters.

Three evaluation metrics are used for evaluation: 1) average precision of top 10 hits (P@10); 2) average precision of top N hits (P@N), where N is the number of target utterances containing the query term in the test set; 3) mean average precision (MAP).

4.2. Alternative Tokenizers

We first explore the QbyE STD performances of three alternative tokenizers. One alternative is a BUT-style [15] English (EN) phoneme recognizer trained from Switchboard Cellular Phase 1 Corpus, which is well matched to the test condition. The second alternative is the Hungarian (HU) phoneme recognizer developed by BUT. The Hungarian phoneme recognizer shares the same architecture as the English phoneme recognizer, but is language-mismatched with the test condition. The third alternative is a GMM tokenizer trained with 10-hour Fisher Data which has no overlap with the test data. 39-dimensional MFCCs processed by voice activity detection (VAD) and utterance-based cepstral mean substraction (CMS) are used as features to train the GMM tokenizer. The number of Gaussian components which is tuned on the development set, is set to 60.

Table 1 shows the corresponding results with the three DTW local distance metrics. It is not surprising that the well-trained English phoneme recognizer performs the best among the three tokenizers. Although the Hungarian phoneme recognizer is also well-trained with enough training resources, the mismatch in language causes significant degradation of detection performances, and relative drops of 47.3%, 47.7% and 44.4% in MAP can be observed with the three DTW distance metrics. On the other hand, the *unsupervised* GMM tokenizer provides similar performances as the Hungarian phoneme recognizer. This indicates that the usefulness of the GMM tokenizer is also limited. It is seen that Bhattacharyya distance shows the best overall performances among the three DTW distance metrics.

Table 1. Performances of Alternative Tokenizers

	Tokenizers	P@N	P@10	MAP
D_{IP}	EN	0.632	0.826	0.674
	HU	0.364	0.490	0.355
	GMM	0.373	0.510	0.359
D_{Cos}	EN	0.637	0.810	0.664
	HU	0.355	0.492	0.347
	GMM	0.354	0.498	0.342
D_{Bhatt}	EN	**0.661**	**0.838**	**0.700**
	HU	0.394	0.520	0.389
	GMM	0.375	0.508	0.358

4.3. ASM Tokenizer

We use the same training data of the GMM tokenizer to train the ASM tokenizer. The number of ASM units is set to 60, which is the same as the number of Gaussian components in the GMM tokenizer. Each ASM unit is modeled by a one-state HMM of 8 Gaussian mixtures. We did not see obvious improvement using 3-state HMM or more Gaussian mixtures on the development set. Table 2 shows the QbyE STD performances of the ASM tokenizer. With Bhattacharyya distance, the ASM tokenizer achieves relative improvements of 8.27% in P@N, 11.4% in P@10, and 12.9% in MAP

[1] *Fisher English Training Speech Part 1 Speech Corpus*, LDC

[2] *TIMIT Acoustic-Phonetic Continuous Speech Corpus*, LDC

compared to the GMM tokenizer. This demonstrates the advantage of the ASM tokenizer.

We then examine the performances after applying the speaker normalization techniques mentioned in Section 3.4. From Table 2, we observe that the speaker normalization techniques improve the detection performances substantially. With Bhattacharyya distance, VTLN contributes to a relative 21.8% improvement in MAP, and CMLLR contributes to a relative 10.9% improvement in MAP. These demonstrate the great effectiveness of the speaker normalization for unsupervised spoken term detection. It is also noted that VTLN performs consistently better than CMLLR in our experiments. This is probably because they are applied on a per utterance basis, and several test utterances are not long enough for reliable CMLLR transform estimation.

Table 2. Performances of ASM Tokenizers

	Tokenizers	P@N	P@10	MAP
D_{IP}	ASM	0.403	0.556	0.390
	ASM VTLN	0.458	0.662	0.479
	ASM CMLLR	0.424	0.612	0.425
D_{Cos}	ASM	0.402	0.556	0.392
	ASM VTLN	0.451	0.658	0.481
	ASM CMLLR	0.425	0.596	0.424
D_{Bhatt}	ASM	0.406	0.566	0.404
	ASM VTLN	**0.474**	**0.666**	**0.492**
	ASM CMLLR	0.442	0.624	0.448

The detection performance is further examined as a function of the number of query examples and the results are shown in Figure 2. As can be seen, increasing query examples improves the performance consistently, but when the number of query examples goes beyond 6, the improvement becomes less significant. Figure 2 also shows the performances of the combination of the ASM tokenizer and the Hungarian phoneme recognizer. The combination is done by linear score fusion with equal weights. It is promising that the combination leads to a MAP of 0.591, which is a 9.9% absolute improvement to the performance of ASM with VTLN normalization. This indicates the complementariness between the *unsupervised* tokenizer and the langauge-mismatched *supervised* tokenizer for STD. It is observed that the combination of the ASM tokenizer and the English phoneme recognizer also leads to an obvious improvement on

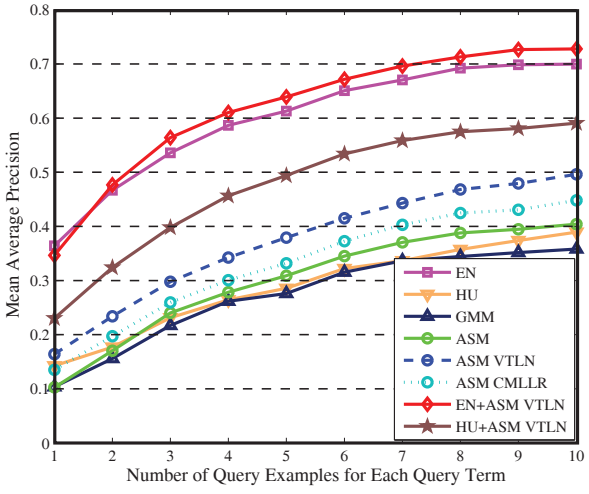

Fig. 2. STD performances with different numbers of query examples. D_{Bhatt} is used as the DTW local distance metric.

5. CONCLUSIONS AND FUTURE WORK

This paper investigates the use of an ASM tokenizer in the framework of posteriorgram-based template matching for QbyE STD task, and evaluates its performance compared to a GMM tokenizer and two phoneme recognizers. It is shown that the ASM tokenizer outperforms the language-mismatched phoneme recognizer and the GMM tokenizer. The performance is further improved by applying speaker normalization techniques. The ASM tokenizer can provide a further performance improvement when it is combined with the two phoneme recognizers. Future work includes: 1) designing more sophisticated clustering techniques for ASM training; 2) a complete and task-oriented theoretical analysis on the ASM tokenizer.

6. ACKNOWLEDGEMENT

This research is partially supported by the General Research Fund (Ref: CUHK 414010) from the Hong Kong Research Grants Council.

7. REFERENCES

[1] W. Shen, C.M. White, and T.J. Hazen, "A comparison of query-by-example methods for spoken term detection," *Proc. INTERSPEECH*, pp. 2143–2146, 2009.

[2] C. Chelba, T.J. Hazen, and M. Saraçlar, "Retrieval and browsing of spoken content," *IEEE Signal Processing Magazine*, vol. 25, no. 3, pp. 39–49, 2008.

[3] T.J. Hazen, W. Shen, and C. White, "Query-by-example spoken term detection using phonetic posteriorgram templates," *Proc. ASRU*, pp. 421–426, 2009.

[4] Y. Zhang and J.R. Glass, "Unsupervised spoken keyword spotting via segmental DTW on Gaussian posteriorgrams," *Proc. ASRU*, pp. 398–403, 2009.

[5] H. Li, B. Ma, and C.H. Lee, "A vector space modeling approach to spoken language identification," *IEEE Trans. ASLP*, vol. 15, no. 1, pp. 271–284, 2007.

[6] M. Ostendorf, V.V. Digalakis, and O.A. Kimball, "From HMM's to segment models: A unified view of stochastic modeling for speech recognition," *IEEE Trans. SAP*, vol. 4, no. 5, pp. 360–378, 1996.

[7] C. Chan and L. Lee, "Unsupervised hidden markov modeling of spoken queries for spoken term detection without speech recognition," *Proc. INTERSPEECH*, pp. 2141–2144, 2011.

[8] M. Huijbregts, M. McLaren, and D.V. Leeuwen, "Unsupervised acoustic sub-word unit detection for query-by-example spoken term detection," *Proc. ICASSP*, pp. 4436–4439, 2011.

[9] G. Aradilla, J. Vepa, and H. Bourlard, "Using posterior-based features in template matching for speech recognition," *Proc. INTERSPEECH*, pp. 1186–1189, 2006.

[10] Y. Qiao, N. Shimomura, and N. Minematsu, "Unsupervised optimal phoneme segmentation: objectives, algorithm and comparisons," *Proc. ICASSP*, pp. 3989–3992, 2008.

[11] C.H. Lee, F.K. Soong, and B.H. Juang, "A segment model based approach to speech recognition," *Proc. ICASSP*, pp. 501–541, 1988.

[12] D.A. Reynolds and R.C. Rose, "Robust text-independent speaker identification using Gaussian mixture speaker models," *IEEE Trans. SAP*, vol. 3, no. 1, pp. 72–83, 1995.

[13] P.A. Torres-Carrasquillo, D.A. Reynolds, and JR Deller Jr, "Language identification using Gaussian mixture model tokenization," *Proc. ICSLP*, pp. 757–760, 2002.

[14] M.J.F. Gales, "Maximum likelihood linear transformations for HMM-based speech recognition," *Computer Speech and Language*, vol. 12, pp. 75–98, 1998.

[15] P. Schwarz, P. Matejka, and J. Cernocky, "Hierarchical structures of neural networks for phoneme recognition," *Proc. ICASSP*, pp. 325–328, 2006.