Query-By-Example Spoken Term Detection Using Phonetic Posteriorgram Templates[1]

Timothy J. Hazen*, Wade Shen*, and Christopher White#

* *MIT Lincoln Laboratory*
Lexington, Massachusetts, USA

Johns Hopkins University
Baltimore, Maryland, USA

***Abstract*—This paper examines a query-by-example approach to spoken term detection in audio files. The approach is designed for low-resource situations in which limited or no in-domain training material is available and accurate word-based speech recognition capability is unavailable. Instead of using word or phone strings as search terms, the user presents the system with audio snippets of desired search terms to act as the queries. Query and test materials are represented using phonetic posteriorgrams obtained from a phonetic recognition system. Query matches in the test data are located using a modified dynamic time warping search between query templates and test utterances. Experiments using this approach are presented using data from the Fisher corpus.**

I. Introduction

In recent years, spoken term detection for spoken audio data has received increasing attention in the research and development communities [3]. Systems employing a large vocabulary continuous speech recognition (LVCSR) approach are common and have been shown to be very accurate for a variety of well-resourced tasks [6], [10]. However, concerns over the computational requirements and vocabulary coverage of LVCSR systems have been raised, leading some researchers to focus on systems that employ a phonetic approach to spoken term detection [8], [14]. In fact, for some tasks a phonetic approach may be the only feasible approach. This is particularly true when the available training data for learning a vocabulary and language model is severely limited, thus impeding the development of an LVCSR system that can provide adequate lexical coverage. In this case, phonetic modeling is needed to combat the out-of-vocabulary word problem.

Another difficult scenario involves audio search for data impoverished languages or accents. In this case, it may not be possible to adequately train language specific acoustic models and the system may need to rely on a cross-language or language-independent modeling approach. If there are phonetic differences between the phonetic recognition system and the language or accent of the test data, it can also be assumed that an accurate lexical dictionary mapping words to the recognizer's phonetic units may not be available.

In this paper, we focus on spoken term detection for the situations discussed above where standard techniques for spoken term detection are inadequate. For these cases we explore a *query-by-example* approach to spoken term detection in which the user has found some data of interest within their data pool (through random browsing or some other method) and they wish to find more data like it in their data pool. In the query-by-example approach the user selects audio snippets containing a keyword (or key-phrase) of interest. These audio snippets then become the *queries* for the user's search. The system must then search the pool of test audio for segments that closely resemble these query examples.

Query-by-example search has been applied in a variety of audio applications including sound classification [15] and music retrieval [13], but audio-based query-by-example retrieval of speech has received little attention in the speech community. However, the speech community does have a rich history of applying template matching techniques to the speech recognition problem. Thus, the approach that we employ in this work borrows heavily from the basic ideas of template-based speech recognition using dynamic time warping [7], [9]. Most of the early speech-based template matching work relied on direct acoustic similarity measures when matching templates to test data. However, acoustic similarity measures can suffer from mismatches in speaker and/or environment. Alternatively, some recent work has examined the use of symbolic features within a template matching approach [1], [2]. Our work similarly uses a phonetically-based symbolic representation within a template-based approach to query-by-example spoken term detection. In our previous work, we have explored the query-by-example problem using a hidden Markov modeling approach based on phonetic confusion networks [12]. In this work we examine a template matching approach based on a phonetic posteriorgram representation.

II. Query-by-Example Using Posteriorgrams

A. Phonetic Posteriorgram Representation

Many spoken term detection systems rely on a network or lattice representation of phonetic recognition hypotheses for

[1]This work was sponsored by the Department of Defense under Air Force Contract FA8721-05-C-0002. Opinions, interpretations, conclusions, and recommendations are those of the authors and are not necessarily endorsed by the United States Government

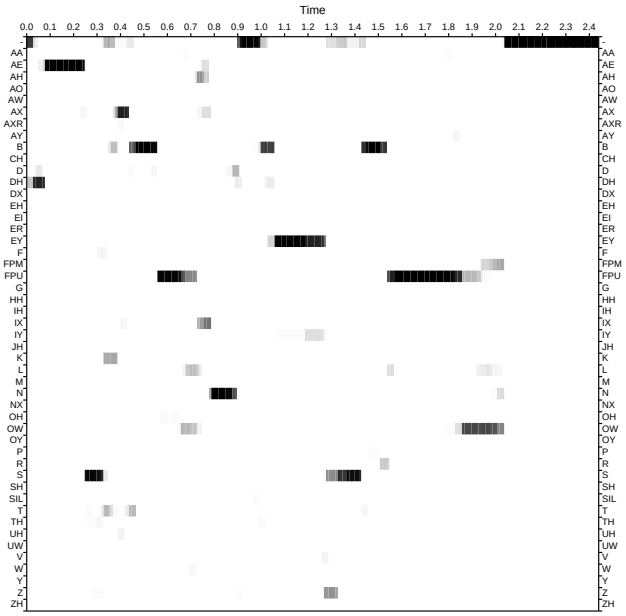

Fig. 1. An example posteriorgram representation for the spoken phrase "basketball and baseball".

capturing speech recognition output information. When using lattice representations it is typically assumed that the phonetic string of a query term is known and can be found within a lattice using standard search techniques.

In our query-by-example scenario, we wish to find similarity between the query example and matching segments in test utterances even though the underlying phonetic content of the query is unknown. Our approach uses a representation that is often referred to as a phonetic posteriorgram, which is a time-vs.-class matrix representing the posterior probability of each phonetic class for each specific time frame. Posteriorgrams can be computed directly from the frame-based acoustic likelihood scores for each phonetic class at each time frame. Alternatively, a full phonetic recognizer, using both acoustic and phonetic language model scores, can be run to generate a lattice, and the posteriorgram can be computed directly from this lattice.

Figure 1 shows a posteriorgram for an audio segment containing the spoken words "basketball and baseball". The horizontal axis represents time in seconds and the vertical axis represents the individual phonetic classes. The level of darkness in the figure's posteriorgram signifies the posterior probability of the class at a given time; posterior probabilities near 1 are black and posterior probabilities near 0 are white.

B. Similarity Matching of Posteriorgrams

To locate audio segments that are similar to a query sample using posteriorgrams, we first define a measure for comparing individual posterior distributions. Let us represent the posteriorgram for a speech segment Q as a series of vectors containing phonetic posterior probabilities for N frames in the speech segment as:

$$Q = \{\vec{q}_1, \ldots, \vec{q}_N\} \tag{1}$$

We will use Q to refer to the posteriorgram for a query segment, and X to refer to a posteriorgram for a test utterance containing M frames. The goal is thus to determine if there is a similarity match between Q and any region in X.

In work by Aradilla *et al* [1], [2], the Kullback-Leibler divergence metric has been used as a similarity measure between posterior distributions. However, the underlying goal is to identify speech regions in a test utterance that match the phonetic content of the query. Divergence measures may capture the similarity between distributions but they do not model the likelihood that two posterior distribution estimates could have resulted from the same underlying phonetic event.

Given two posterior distributions $\vec{q}$ and $\vec{x}$, the probability that these distributions resulted from the same underlying phonetic event is easily represented by their dot product:

$$P\left(\text{phone}\{\vec{q}\} = \text{phone}\{\vec{x}\}\right) = \vec{q} \cdot \vec{x}. \tag{2}$$

We can reinterpret this probability as a distance-like measure by converting it into the log probability space as follows:

$$D(\vec{q}, \vec{x}) = -\log(\vec{q} \cdot \vec{x}) \tag{3}$$

Here, values close to zero represent strong similarity between $\vec{q}$ and $\vec{x}$ while large positive values represent dissimilarity. In practice, this expression could fail in the situation where many of the values of $\vec{q}$ and $\vec{x}$ are zero, leading to $\vec{q} \cdot \vec{x} = 0$ and hence $D(\vec{q}, \vec{x}) = \infty$. To compensate, we can smooth each posteriorgram distribution as follows:

$$\vec{q}\,' = (1 - \lambda)\vec{q} + \lambda\vec{u} \tag{4}$$

Here $\vec{u}$ is a vector representing a uniform probability distribution and $\lambda > 0$ assures a non-zero probability for all phonetic posteriors in $\vec{q}\,'$. This smoothing can be applied to the posteriorgrams for both the query and test material.

To compare the posteriorgrams of a query example and a test utterance, we compute the similarity measure between the individual posterior distributions for all N frames in the query against the individual posterior distributions for all M frames in test utterances. This results in an $N \times M$ similarity matrix.

C. Dynamic Time Warping Search

When comparing a query posteriorgram against a test posteriorgram, our goal is to find a region of time in the test posteriorgram with high similarity to the query sample posteriorgram. For example, Figure 2 shows a posteriorgram similarity matrices between a test utterance (along the x-axis) and a query term (along the y-axis). The dark regions represent strong similarity between frames of the test utterance and frames of the query example, while the light regions represent dissimilarity. Ideally, a match between a query segment and a segment of a test utterance would be represented by an upper-left to lower-right diagonal sequence of highly similar regions (or blocks) within the similarity matrix. The matrix in Figure 2 shows an example of a valid match between the query and a test utterance with the green line representing the best matching alignment of the query sample against its matching region in the test utterance.

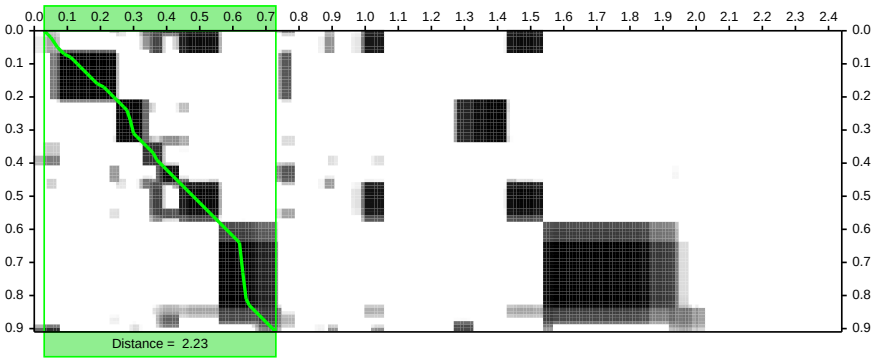

Fig. 2. An example posteriorgram similarity matrices between a query lasting 0.9 seconds (along the y-axis) and a test utterance lasting 2.4 seconds (along the x-axis). The matrix shows the superimposed results of a DTW search for a valid match of the query within the test utterance.

In order to search for a well matched path between a query example and a test utterance, as exemplified by the path found in Figure 2, we use a modified dynamic time warping (DTW) search. The DTW search employs two primary constraints. First the DTW search accumulates similarity scores along path extensions in the search space. A path which has progressed in the DTW search to index i in the query and index j in the test can be extended to index $i+n$ in the query and index $j+m$ in the test, subject to the constraint that either $n = 1$ or $m = 1$. In other words, the search disallows simultaneous multiframe path extensions in both the query and test segments.

Our second constraint is to favor path extensions with similar durations by scaling the similarity score along individual extensions of a hypothesized path by an alignment slope factor defined as:

$$\gamma = \max(n, m) \tag{5}$$

This alignment slope factor is further exponentially weighted by a factor φ which is designed to control the strength of the alignment slope constraint.

With these constraints we express the score for any path extension in the DTW where $m = 1$ as:

$$S_{ext}(\vec{q}_i \rightarrow \vec{q}_{i+n}, \vec{x}_j \rightarrow \vec{x}_{j+1}) = \gamma^{\varphi} \sum_{k=1}^{n} D(\vec{q}_{i+k}, \vec{x}_{j+1}) \tag{6}$$

Similarly, the score for any path extension where $n = 1$ is:

$$S_{ext}(\vec{q}_i \rightarrow \vec{q}_{i+1}, \vec{x}_j \rightarrow \vec{x}_{j+m}) = \frac{\gamma^{\varphi}}{m} \sum_{k=1}^{m} D(\vec{q}_{i+1}, \vec{x}_{j+k}) \tag{7}$$

Within these expressions the distance score for the path extension is normalized by the number of frames m absorbed by the test side of the extension. This insures that the total score of any final path receives equal contribution from each query frame regardless of the total number of test frames absorbed in the path. Also note the contribution of the duration constraint variable φ; when $\varphi = 0$, no diagonal alignment constraint is enforced, while larger values of φ will force the search to strongly favor perfectly diagonal path alignments (i.e., one for one matching of query and test frames).

The final score $S(X|Q)$ for any full path through the posteriorgram is the sum of the scores of the full set of path extensions taken in that path during the search, normalized by the total number of frames in the query. The DTW search finds the minimum scoring path through the similarity matrix.

D. Using Multiple Queries

Our query-by-example approach can also be used when multiple examples of a query term are available. There are two basic approaches that can be taken to combine multiple templates. The first approach would be to combine the templates through some process into a single template combining the characteristics of the multiple queries. A second simpler, but more computationally expensive approach, is to use all available query templates to generate scores, and to then combine the scores from these templates. In this work we use the second approach for combining scores. We leave an examination of the first approach (i.e. combining queries into a single template) for future work.

Our system currently employs a flexible score "averaging" expression for score fusion. We consider the case where N_Q queries (labeled $Q_1, \ldots, Q_{N_Q}$) are used to generate N_Q different scores for an input utterance X. The total score for X for the fusion of query scores can be computed using this expression:

$$S(X|Q_1, \ldots, Q_{N_Q}) = -\frac{1}{\alpha} \log \frac{1}{N_Q} \sum_{i}^{N_Q} \exp(-\alpha S(X|Q_i)) \tag{8}$$

The value of α in this expression determines the relative contribution of the scores within the averaging that is performed. When $\alpha = 0$ the fusion expression computes the average of the scores in the *log probability* space. A value of $\alpha = 1$ represents an average of the scores in the *probability* space and $\alpha = \infty$ returns the maximum score.

E. User-Driven Relevance Feedback

Because our approach allows for the combination of results from multiple queries, it becomes straightforward to implement user-driven relevance feedback. After returning a ranked list of potential utterance *hits* to a user, the user can listen to an individual utterance and provide feedback to the system indicating whether the desired word was or was not present

TABLE I
THE COLLECTION OF QUERY TERMS USED IN THE EVALUATION WITH THEIR OCCURRENCE COUNTS IN THE EVALUATION DATA.

age (11)	money (21)	married (21)	basically (12)
war (12)	always (36)	business (22)	different (20)
down (36)	couple (15)	children (18)	important (13)
food (18)	family (31)	happened (12)	sometimes (14)
nice (23)	pretty (40)	problems (11)	definitely (29)
agree (15)	school (12)	remember (16)	especially (17)
funny (14)	exactly (15)	supposed (15)	everything (27)
never (42)	talking (14)	thinking (18)	government (17)
t. v. (11)	parents (20)	together (14)	understand (14)
years (30)	started (14)	whatever (21)	interesting (15)

in that utterance. Newly observed positive examples can be scored against the remaining utterances in the returned list. The new scores can then be factored into the previously computed scores and the list can be reordered. This process can be repeated after each new observation of the desired word in the returned data, hopefully improving the accuracy of the rankings in the remaining unseen portion of the data being searched by the user.

III. EXPERIMENTAL RESULTS

A. Phonetic Recognition

For our initial evaluations we evaluate using the output of a phonetic recognition system developed at the Brno University of Technology (BUT) [11]. The recognizer is trained on only 10 hours of English data from the Switchboard cellular corpus [4]. The recognizer generates a lattice of phonetic hypotheses which is then post-processed into a posteriorgram. Pruning is employed during recognition such that only a sparse subset of phones in the posteriorgram yield a non-zero posterior score at any frame. This serves to reduce the memory requirements of the posteriorgram representation. During the DTW search the smoothing operation in Equation 4 is employed to approximate the probability mass lost due to the pruning of the recognition results.

B. Experimental Data

We evaluate our system on a set of 36 conversations contained in the Fisher English development test set from the NIST Spoken Term Detection Evaluation in 2006 [5]. The evaluation data was automatically segmented into 3501 utterances of 8 seconds duration or less.

For query terms, we have selected 40 words that have occurrence counts within the evaluation set of between 11 and 42. For these terms, the prior likelihood of observing a term in one of the 3501 test utterances varies between 0.003 and 0.011. In other words, these selected terms are relatively rare in comparison to the most common English words, but are not so exceedingly rare that we have only a few examples of each for our experiments.

Table I shows the full collection of query terms with the number of times they appear in the evaluation data. To serve as the query examples of these terms we have excised posteriorgram representations of examples of these terms from conversations in the Fisher English Phase 1 corpus [4]. Five randomly selected examples of each query term are used for our experiments with their start and end times being determined from independently generated forced alignments.

C. Evaluation Metrics

For our evaluation, we examine three different evaluation metrics: (1) the average precision of the top ten utterance hits returned by a search (P@10), (2) the average precision of the top N search hits (P@N), where N is the number of occurrences of the term in the evaluation data, and (3) the average detection equal error rate (EER) where the rate of missed detections is equivalent to the rate of false alarms on a standard detection error trade-off curve. Our evaluation is conducted on a per utterance basis and not on a per term basis, i.e., a correct hit occurs if a returned utterance contains the desired search term and a false alarm occurs if a returned utterance does not contain the search term. In this evaluation, long and short utterances are treated as equivalent, and the resulting EER scores will be higher that those reported using the term-based NIST STD evaluation tools [5]. It is also worth noting that the P@N measure represents the point on the precision-recall curve where precision and recall are equal.

D. Primary Experimental Results

In our primary experiments, we evaluated our posteriorgram DTW (PG-DTW) system under three different query-by-example evaluation cases (described below). In each of these three cases, we have set the posteriorgram floor variable from Equation 4 to $\lambda = 10^{-5}$ (based upon preliminary experiments on an independent development test set). We also evaluate using two different settings for the duration constraint variable. In the first setting, the duration constraint is completely ignored by setting $\varphi = 0$. In the second setting, the duration constraint is employed by setting $\varphi = 1$.

In our first evaluation case, we assume that only one query example is available for each search. Under this assumption we evaluate each of the 40 terms using each of the five examples of that term yielding 200 different trial searches.

In our second evaluation case, we assume that we have five query examples available for each search term. We fuse the results from the five query examples for each term into a single search result to yield a total of 40 different term searches. For these experiments we set the fusion variable to $\alpha = 0$ such that our fused scores are a simple average of the scores.

Our third evaluation case is the oracle case where we construct the queries directly from a lexical dictionary entry. Because we do not have any duration information available to accompany our dictionary entry, each phoneme receives a single frame in the query posteriorgram and the PG-DTW duration constraint is set to $\varphi = 0$. Each phoneme in the dictionary entry receives a weight of 1 in the query posteriorgram. When alternate pronunciations are provided for a phoneme slot, each alternate phoneme is given a weight of 1.

Table II shows query-by-example retrieval results that compare our new PG-DTW approach against our pre-existing

TABLE II
QUERY-BY-EXAMPLE TERM DETECTION RESULTS USING DIFFERENT QUERY CONSTRAINTS FOR TWO DIFFERENT MODELING APPROACHES.

System	Queries	Duration Scale	P@10	P@N	EER (%)
DHMM	1 Example	N/A	.351	.279	20.1
PG-DTW	1 Example	$\varphi = 0$	.335	.269	18.2
PG-DTW	1 Example	$\varphi = 1$	.363	.293	17.1
DHMM	5 Examples	N/A	.610	.456	11.2
PG-DTW	5 Examples	$\varphi = 0$	.570	.447	12.8
PG-DTW	5 Examples	$\varphi = 1$	.633	.528	10.4
DHMM	Dictionary	N/A	.712	.548	9.6
PG-DTW	Dictionary	$\varphi = 0$	.708	.560	10.5

TABLE III
RESULTS FOR THE PG-DTW SYSTEM USING FIVE QUERY EXAMPLES BROKEN DOWN BY THE PHONETIC LENGTHS OF THE QUERY TERMS.

# Phones	P@10	P@N	EER (%)
2-4	0.533	0.462	13.7
5-6	0.633	0.514	9.0
7-9	0.706	0.588	9.0

discrete hidden Markov model (DHMM) approach (as described in [12]). The DHMM models phonetic sequences using a segment-based or *column-based* confusion network representation of the lattice rather than using the fine-grained frame-based posteriorgram. A column-based HMM containing one state per confusion network column is constructed for each query term. The emitting observation function for each state is a phonetic unigram distribution based on the phonetic posteriors for the corresponding column in the confusion network. As such the DHMM can be viewed as a compact approximation of the full posteriorgram used by the DTW approach. The DHMM accumulates segment-level scores while PG-DTW accumulates frame-level scores. Thus, the PG-DTW scoring mechanism inherently gives more weight to the longer phonemes in the query than the shorter ones, while the DHMM weights all phonetic segments equally regardless of length. In its current form, the DHMM approach does not account for phoneme durations while PG-DTW incorporates a duration constraint through the variable φ.

In comparing the query-by-example systems, the PG-DTW system with a duration constraint gives the best performance for both the 1-query and 5-query conditions. Removing the duration constraint harms the performance of the PG-DTW system by a modest amount in both experimental cases. The DHMM performs slightly worse than the PG-DTW system that employs the duration constraint, but better than the PG-DTW system with no duration constraint.

For all systems, the single query example results suffer from poor precision. The best system (the PG-DTW system using the duration constraint) achieves an average precision at 10 of only 0.36, while the precision at N is less than 0.3 for all three systems. When using 5 query examples, the results are significantly better with a top 10 precision of 0.63 for the PG-DTW system using a duration constraint. By comparison, the oracle systems using the known dictionary entry both achieve a precision at 10 of 0.71. When examining the systems using the EER measure, the 5-example PG-DTW system using the duration constraint actually performs marginally better than the oracle PG-DTW system, though it is still worse than the oracle DHMM system. To be fair, though, the oracle systems do not use any duration information.

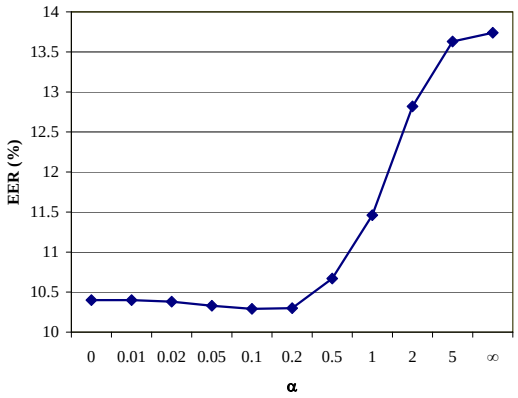

Fig. 3. Effect of score fusion parameter α on EER of PG-DTW term detection using five query examples.

Table III shows the 5-example PG-DTW results (with the duration constraint) sorted into bins of short, medium, and long query terms based on their phonetic counts. As expected, better performance is observed on the longer terms.

E. Experimental Results with Variable Score Fusion

Figure 3 shows the results for the PG-DTW system using 5 query examples as the fusion parameter α in Equation 8 is varied. While there is a slight advantage to setting the value to $\alpha = 0.2$ for this test set, this advantage is minimal and may not hold for other test sets. Because performance degrades significantly for $\alpha > 0.2$, using a setting of $\alpha = 0$ appears to be the wisest solution for score fusion using the method presented in Equation 8.

F. Experimental Results Using Relevance Feedback

When performing query searches in an interactive mode, one would expect improved performance by employing user-driven relevance feedback. We simulate this scenario experimentally using the following procedure.

1) Using a set of query examples, compute the scores for all queries against all unexamined test utterances and return the ranked list of test utterance candidates.
2) Examine the highest ranked previously unexamined utterance from the ranked list and determine if it is a positive example of the desired search term. If the utterance contains a positive example of the term, add the example into the query set and return to step 1; otherwise, repeat step 2.

We repeat the procedure above until 10 test utterances have been examined by the simulated user. For our evaluation, we compute P@10, P@N, and EER for the ranked list after each new candidate utterance has been examined. To simulate actual usage, each examined utterance from the ranked list that retains its rank position in the list, and only unexamined utterances are reranked. Thus, if 10 utterances have been examined

TABLE IV
IMPROVEMENTS OBSERVED IN QUERY-BY-EXAMPLE SPOKEN TERM DETECTION WHEN USING RELEVANCE FEEDBACK TO RESCORE AND RESORT THE RANKED LIST OF CANDIDATE HITS VIA THE INCORPORATION OF NEW POSITIVE EXAMPLES OBSERVED IN THE RANKED LIST.

# of Candidates Examined	P@10	P@N	EER (%)
0	0.633	0.528	10.4
1	0.643	0.514	10.4
2	0.653	0.526	10.4
3	0.665	0.535	10.2
4	0.665	0.541	10.1
5	0.663	0.545	10.0
6	0.665	0.546	9.9
7	0.658	0.548	9.9
8	0.663	0.548	9.9
9	0.663	0.547	9.8

during the relevance feedback process, P@10 represents the precision of the actual 10 utterances examined.

Table IV shows the improvements in performance that are obtained when employing relevance feedback to an initial system using five query examples as up to nine test utterances returned by the system are examined. We see continued improvements in performance as additional positive examples are discovered and added into the example query set for each term. After feedback from the first nine examined utterances has been provided, the P@10 and P@N metrics move closer to the results obtained from the oracle pronunciation system, while the EER obtained by the query-by-example system (9.8%) is now significantly better than the EER of the oracle PG-DTW pronunciation system (10.5%).

IV. CONCLUSION

In this paper we have presented a query-by-example approach to the spoken term detection problem for situations where data resources and knowledge are limited and word-based recognition is unavailable. In this approach phonetic posteriorgram templates are constructed from audio examples and then compared against test utterances using a dynamic time warping approach. Our experiments have verified the viability of this approach. The accuracy of our new PG-DTW approach compares favorably with both our previous DHMM query-by-example system and with systems which have access to the known dictionary pronunciations of the search terms.

One aspect of our approach that has not yet been discussed is its computational needs. While the offline recognition costs are similar to other phonetic retrieval systems, the online search and retrieval costs could be prohibitively expensive for low-latency online searches of large corpora. Currently, the DTW search time increases linearly with both the amount of data to be searched and the number of query examples available for search. The DTW search time is not insubstantial and is significantly higher than the index look-up approaches employed by standard search engines. While there are methods we can employ to improve the computational efficiency of our system, we envision that the practical application of this technique is to serve as a search refinement tool and not as the initial retrieval mechanism, i.e., this technique could be used to rescore and reranked a small subset of results containing the most probable candidate hits (on the order of hundreds to thousands) returned by a standard phonetic indexing and retrieval mechanism. This could allow for improved accuracy with only modest added latency from with the DTW rescoring.

In future work we will examine ways to improve the accuracy and efficiency of our system. In particular we believe the computation can be reduced by combining multiple query templates into a single query template, and by merging similar sequential frames in a posteriorgram into a single multi-frame segment. In essence, we seek to build a system whose computational requirements are more in line with those of our DHMM approach but which also provides a mechanism for modeling the durational information of the observed terms. We will also extend this approach to cases where labeled training data in the target language is not available by employing cross-language phonetic recognition, potentially with unsupervised adaptation to unlabeled target language data.

V. ACKNOWLEDGMENTS

The authors would like to thank Fred Richardson for his help training and running the BUT phonetic recognizer.

REFERENCES

[1] G. Aradilla, J. Vepa and H. Bourlard. "Using posterior-based features in template matching for speech recognition," in *Proc. Interspeech*, Pittsburgh, Sep. 2006.
[2] G. Aradilla, H. Bourlard, and M. Magimai-Doss. "Posterior features applied to speech recognition tasks with user-defined vocabulary," in *Proc. ICASSP*, Taipei, May 2009.
[3] C. Chelba, T. Hazen and M. Saraçlar, "Retrieval and browsing of spoken content," *IEEE Signal Processing Magazine*, vol. 24, no. 3, pp. 39–49, May 2008.
[4] C. Cieri, D. Miller, and K. Walker, "From Switchboard to Fisher: Telephone collection protocols, their uses and yields," in *Proc. Interspeech*, Geneva, Sep. 2003.
[5] J. Fiscus, J. Ajot, J. Garofolo, and G. Doddington, "Results of the 2006 Spoken Term Detection Evaluation," in *Proc. 2007 SIGIR Workshop on Searching Spontaneous Conversational Speech*, Amsterdam, July 2007.
[6] D. Miller, *et al*, "Rapid and accurate spoken term detection," in *Proc. Interspeech*, Antwerp, Belgium, 2007.
[7] H. Ney, "The use of a one-stage dynamic programming algorithm for connected word recognition," *IEEE Trans. on Acoustics, Speech and Signal Proc.*, vol. 32, no. 2, pp. 263-271, April 1984.
[8] K. Ng, "Subword-based approaches for spoken document retrieval," Ph.D. dissertation, Massachusetts Institute of Technology, 2000.
[9] L. Rabiner, A. Rosenberg, and S. Levinson, "Considerations in dynamic time warping algorithms for discrete word recognition," *IEEE Trans. on Acoustics, Speech and Signal Proc.*, vol. 26, no. 6, pp. 575-582, December 1978.
[10] M. Saraçlar and R. Sproat, "Lattice-based search for spoken utterance retrieval," in *Proc. HLT-NAACL*, Boston, 2004.
[11] P. Schwarz, P. Matějka, and J. Černocký, "Towards lower error rates in phoneme recognition," in *Proc. Int. Conf. on Text, Speech and Dialogue*, Brno, Czech Republic, Sep. 2004.
[12] W. Shen, C. White, and T. Hazen, "A comparison of query-by-example methods for spoken term detection," in *Proc. Interspeech*, Brighton, England, Sep. 2009.
[13] G. Tzanetakis, A. Ermolinsky, and P. Cook, "Pitch histograms in audio and symbolic music information retrieval," *Journal of New Music Research*, vol. 32, no. 2, pp. 143-152, June 2003.
[14] P. Yu, K. Chen, C. Ma, and F. Seide, "Vocabulary-independent indexing of spontaneous speech," *IEEE Trans. Speech Audio Processing*, vol. 13, no. 5, pp. 635–643, Sept. 2005.
[15] T. Zhang and C. Kuo, "Hierarchical classification of audio data for archiving and retrieval," in *Proc. ICASSP*, Phoenix, March 1999.