Query-by-Example Spoken Term Detection For OOV Terms

Carolina Parada [#1], Abhinav Sethy [*2], Bhuvana Ramabhadran [*3]

[#] *Center for Language and Speech Processing, Johns Hopkins University*
3400 North Charles Street, Baltimore MD 21210, USA
[1] carolinap@jhu.edu

[*] *IBM T.J. Watson Research Center*
Yorktown Heights, N.Y. 10568, USA
[2] asethy@us.ibm.com
[3] bhuvana@us.ibm.com

Abstract—**The goal of Spoken Term Detection (STD) technology is to allow open vocabulary search over large collections of speech content. In this paper, we address cases where search term(s) of interest (queries) are acoustic examples. This is provided either by identifying a region of interest in a speech stream or by speaking the query term. Queries often relate to named-entities and foreign words, which typically have poor coverage in the vocabulary of Large Vocabulary Continuous Speech Recognition (LVCSR) systems. Throughout this paper, we focus on query-by-example search for such out-of-vocabulary (OOV) query terms. We build upon a finite state transducer (FST) based search and indexing system [1] to address the query by example search for OOV terms by representing both the query and the index as phonetic lattices from the output of an LVCSR system. We provide results comparing different representations and generation mechanisms for both queries and indexes built with word and combined word and subword units [2]. We also present a two-pass method which uses query-by-example search using the best hit identified in an initial pass to augment the STD search results. The results demonstrate that query-by-example search can yield a significantly better performance, measured using Actual Term-Weighted Value (ATWV), of 0.479 when compared to a baseline ATWV of 0.325 that uses reference pronunciations for OOVs. Further improvements can be obtained with the proposed two pass approach and filtering using the expected unigram counts from the LVCSR system's lexicon.**

I. Introduction

The fast-growing availability of recorded speech calls for efficient and scalable solutions to index and search this data. Spoken Term Detection (STD) is a key technology aimed at open-vocabulary search over large collections of spoken documents. A common approach to STD is to employ a large vocabulary continuous speech recognition (LVCSR) system to obtain word lattices and extend classical Information Retrieval techniques to word lattices. Such approaches have been shown to be very accurate for well-resourced tasks [3], [1].

A significant challenge in the STD task is the search for queries containing OOV terms. As queries often relate to named-entities and foreign words, they have typically poor coverage in the LVCSR system's vocabulary, and hence searching through word lattices will not return any results. Common approaches to overcome this problem consist of searching sub-word lattices ([4], [1], [2] among others). Such approaches assume that at query time an orthographic representation of the term can be converted to a sensible phonetic representation. This is typically done using grapheme to phoneme conversion algorithms, which are not always available and may not work accurately for all query terms, particularly, terms in foreign languages.

In this paper, we focus on spoken term detection of OOV queries only. In particular, we examine an alternative interface for phonetic search of OOV terms, namely query-by-example (referred to hereafter as QbyE). We envision an application, where the user provides query samples, either via speech cuts (audio snipets) corresponding to the query or by speaking the query (speech to speech retrieval). These audio snippets become the query, and the system must search the pool of data to retrieve audio samples that resemble the query sample.

QbyE search for OOV's can be considered as an extension to the well-know query expansion method proposed in text-based information retrieval (IR) methods. In classical IR, query expansion is based on expanding the query with additional words using relevance feedback methods, synonyms of query terms, various morphological forms of the query terms and spelling corrections. In QbyE, the rich representation of the query as the phonetic lattice from the output of an LVCSR system, provides additional information over a textual representation of the query.

Query-by-example has been used previously in several audio applications such as: sound classification [5], music retrieval [6], [7], as well as in spoken document retrieval [8], but has received only limited attention in the Spoken Term Detection community. A QbyE approach to STD was considered by [9]. The authors employ a query-by-example approach to STD, treating the problem as a string-distance comparison between the confusion network of the query sample and test utterances.

The approach presented in this paper, exploits the flexibility of a weighted finite state transducer (WFST) based indexing system [10], [4], which allows us to use the lattice representation of the audio sample directly as a query to the search system. We compare the performance of the STD system when the query is represented using a lattice excision to that of a spoken query. Lattice excision corresponds to the case when the user selects a portion of the audio from an existing index and requests retrieval of similar examples. We also present a two-pass method which uses QbyE search using the best hit identified in an initial pass to augment the STD system results. This can be used to improve STD performance using textual queries.

This paper is organized as follows. In Section II, we describe our WFST based indexing and retreival system for QbyE. Section III describes the corpora and LVCSR systems used in all our experiments. In Section IV we present baseline results with text-based queries. QbyE results for different query representation and generation schemes on word and hybrid indexes are presented and analyzed in Section V. In Section VI, we present a two pass QbyE strategy and a method to reduce false alarms using the expected unigram counts from the LVCSR system's lexicon. We conclude with a summary of our findings and directions for future work (Section VII).

II. Search, Indexing and Query generation system

In this section we describe the overall framework of our STD system and the methods used in our experiments. We assume that the audio to be indexed has been processed with an LVCSR system and the corresponding word or sub-word lattices are available. Phonetic lattices are subsequently derived and used to build the indexes used in all of our experiments. All silences and hesitations in the lattices are converted to <epsilon> arcs.

A. Pre-Processing

Prior to creating the index, the phonetic lattices are preprocessed into weighted finite state transducers (WFST), and the timing information is pushed onto the output label of each arc in the lattice. An additional normalization step (achieved by weight pushing in the log-semiring) [11] converts the weights into the desired posterior probabilities.

In essence, each arc in the resulting WFST representing the lattice is a 5-tuple (p, i, o, w, q) where $p \in Q$ is the start state, $q \in Q$ is the end state, $i \in \Sigma$ is the input label (phone), $o \in \Re$ is the output label (start-time associated with state p), and $w \in \Re^+$ is (neg log of) posterior probability associated with i. Q is a finite set of states and Σ is the input alphabet.

B. WFST-based Indexing and Retrieval

The general indexation of weighted automata provides an efficient means of indexing the pre-processed lattices. The algorithm described in [10] creates a full index represented as a WFST, which maps each substring x to the set of indexes in the automata in which x appears. Here, the weight of each path gives the within utterance expected counts of the substring corresponding to that path. The algorithm presented in [10] is optimal for search: the search is linear in the size of the query string and the number of indexes of the weighted automata in which it appears.

At search time, the query is represented as a weighted acceptor and using a single composition operation [11] with the index we can retrieve the automata (lattice) containing it. Note the flexibility of this framework, since it allows us to search for any weighted finite state acceptor as query, an advantage we exploit.

The construction described above only retrieves the lattice indexes. However it can be used as the first pass in a two-pass STD retrieval system as described in [12]. Essentially, once the lattice indexes have been identified (in the first pass), the second pass loads the relevant lattices and extract the time marks corresponding to the query. Alternatively, the index can be modified to perform 1-pass retrieval [4], improving search times at the cost of a larger index and with comparable performance. We implemented a 2-pass FST-based indexing system using the OpenFst toolkit [13], and achieve comparable performance to that reported by [4] as shown in Table I.

C. Evaluation

To evaluate performance, we present results in terms of the NIST 2006 STD Evaluation criteria [14]: "Actual Term-Weighted Value". Such measure requires a binary decision for each instance returned by the system. In this work, we use the *Term Specific Threshold* introduced by [3] to make a binary decision, since it achieved higher performance in our experiments than a global threshold.

D. Query generation

As mentioned in Section I, we need audio samples of queries. Our method uses the lattices corresponding to the audio exemplars as queries to the WFST-based STD system described in II-B. We examine two methods of query generation:

- *Lattice-cuts*: excised speech cuts containing words of interest within a larger utterance. This simulates an application in which the user is listening to some audio, highlights a snippet, and request the system for similar examples.
- *Isolated-decode*: queries spoken in isolation. This represents a speech to speech retrieval application.

Given a lattice (preprocessed as in Section II-A), and the time-marks of interest, the *lattice-cut* queries are generated as follows:

1) Let $S \notin \Sigma$ be a special symbol that denotes the region of interest, say $S = in\_seg$.
2) Let C be the FST shown in Figure 1, where rho is a special symbol that consumes any symbol other than in_seg in this case. This FST essentially maps all symbols other than in_seg to ϵ (eps in Figure 1).
3) For each transition (p, i, o, w, q): if the arc is in the desired interval, it is replaced by (p, i, S, w, q). Otherwise, it is replace with (p, i, ϵ, w, q), resulting in FST L.
4) A lattice-cut is defined using standard WFST operators as: rm-epsilon(project-output($L \circ C$)).

Since the queries are represented by their phonetic lattices, this approach will yield many false alarms, particularly for short query words, which are likely to be substrings of longer

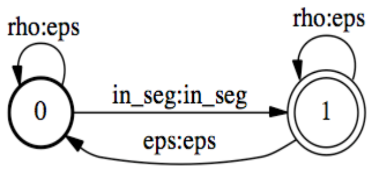

Fig. 1. FST used to cut lattice segments marked as: in_seg

words. To control this we prefilter the query lattices with a *minimum-length* filter (via composition), which ensures all paths in the query lattice have a minimum length of N. In our experiments we used $N = 4$. The minimum length filter FST is a simple FSA that only accepts paths of length N or more.

III. Corpora and ASR System

For our experiments we use an 100 Hour spoken term detection corpus especially designed to emphasize OOV content [4]. The 1290 OOVs in the corpus were selected with a minimum of 5 acoustic instances per word, and common English words were filtered out to obtain meaningful OOVs (e.g. NATALIE, PUTIN, QAEDA, HOLLOWAY), excluding short (less than 4 phones) queries.

The LVCSR system was built using the IBM Speech Recognition Toolkit [15] with acoustic models trained on the 300 hours of HUB4 data with utterances containing OOV words excluded. The excluded utterances (around 100 hours) were used as the test set for the STD experiments. We will refer to this set as **OOVCORP**. The language model for the LVCSR system was trained on 400M words from various text sources with a vocabulary of 83K words. The LVCSR system's WER on a standard RT04 BN test set was 19.4%. In addition to a word based ASR system we will also present results using a hybrid ASR system which uses a combination of word/subword units [16], [17]. Combined word/subword systems have been shown to improve STD performance, especially for the case of OOVs [2]. Our hybrid system's lexicon has 83K words and 20K fragments derived using [16]. The 1290 queries are OOVs to both the word and the hybrid systems.

IV. STD with text queries

In order to establish a comparative baseline for our QbyE results, we first present some experiments using textual queries. First, the textual queries are converted to their phonetic representation using the reference pronunciations of the OOV queries, which we refer to as *reflex*. Second, the queries are represented using the pronunciations obtained form the *letter-to-sound* system described in [18], which we refer to as *L2S*. Table I presents the *reflex* results, which achieve comparable performance to those presented by [4] on the same data set when a word-based lattice index is used.

Table II presents the L2S results obtained for different number of pronunciations and weighting schemes. In general, we found that the phonetic index obtained from the hybrid lattices always outperforms that obtained using the word lattices. For the L2S system, performance peaks at 6 pronunciations, when weighted with the normalized L2S scores. Weighted query terms consistently performs better than unweighted representations.

TABLE I
Reflex results

Data	P(FA)	P(miss)	ATWV
Word 1best	0.00002	0.757	0.227
Word lattices	0.00004	0.638	0.325
Hybrid lattices	0.00002	0.639	0.342

TABLE II
N-Best L2S pronunciations

Data	L2S Model	# Best	P(FA)	P(miss)	ATWV
Word Lattices	Unweighted	3	0.00002	0.675	0.304
	Weighted	6	0.00002	0.674	0.305
Hybrid Lattices	Unweighted	3	0.00002	0.641	0.339
	Weighted	6	0.00002	0.639	0.341

V. Query By Example Experiments

In order to conduct QbyE experiments we need a set of audio samples to serve as queries. We select one instance of each query term to represent the query. The test set is composed of all instances for all query terms, which is decoded using the LVCSR systems decribed in Section III and indexed using the system described in Section II. Thus out of the 23K OOV instances in the OOVCORP corpus, 1290 are used as queries.

A. *Selecting query instances*

The phone error rate (PER) for the query has a significant impact on the performance of the retrieval system. To study the degree to which this affects the QbyE STD performance, we first create transducers for all the instances of the OOV terms using the lattice-cut method described in Section II. Then for each query term, we consider all instances of the query and compute the PER of the best path in the cut lattice against the reference pronunciation (reflex). We select for each OOV term one instance for three cases: best PER, worst PER, and a random instance. Thus we generate a best case query list, a worst case query list and a randomly chosen query list. These will hence, be refered to as **bestq**, **worstq** and the **randomq** sets, and reflect practical situations.

B. *Query Transducers*

In the general case the query transducer which can be generated using either lattice-cuts or isolated decodes (Section II) is similar to LVCSR lattices used for generating the index. A rich lattice representation allows the recovery of more instances of the query term but at the same time can lead to a large number of false alarms. On the other hand, a sparse 1-best representation does not allow for any fuzziness on the query side thus decreasing both the number of hits as well as false alarms. We present results which compare the different lattice and n-best representations of the query (for n=1,3,5) in terms of their hit rate, FA rate and ATWV score.

C. Lattice Cuts with Word and Hybrid Index

We present our first results on indexes built from the word and hybrid LVCSR systems using queries generated by lattice-cuts. For the word system, the results in Table III show the ATWV for the **bestq**, **worstq** and **randomq** query sets with the query transducers being represented as 1,3,5-best and pruned lattices. The corresponding hybrid system results can be seen in Table IV.

The first observation is that in general the hybrid index performs better than the word index. This supports the results in [16] which show that the hybrid system has a better phone error rate especially for OOV regions. The second interesting observation is that the optimal choice of the transducer representation depends on the fidelity of the query decode itself. We can see that for the **bestq** set, the one-best representation provides the best results whereas for the **worstq** set, 3-best and 5-best representations provide better results. In the **randomq** case all representations have no noticeable difference in their performances. It can also be seen that for the random case the QbyE results compare well with the reflex results in Table I. Pruned-lattice query representations perform similar to 5-best multi-path query representations.

An analysis of the QbyE results across the different transducer representations shows that as expected, the false alarms increase as the query representation allows for more paths while the misses are reduced. For the **worstq** set which as a high miss rate the decrease in misses offsets the increase in false alarms, whereas for the **bestq** set the reverse is true. We will present an approach to address the increase in false alarms in Section VI-A.

One caveat in all the ATWV numbers presented in the QbyE search experiments is the inclusion of the sample query in the score computation. The ATWV values are slightly different when the samples are excluded but it does not change the key messages presented in this paper. To keep the test sets comparable to the prior work reported in the literature [4], [19], we leave the results in Tables III-VIII with the sample included. It also serves as a realistic number in a two-pass STD system, where the best hit from the first-pass search is selected as the query.

D. Isolated Decodes

The query transducers used for lattice-cuts are generated by excising a region of a larger lattice. An alternative approach is to decode the spoken query corresponding to the query term with an LVCSR system. One possible advantage of this approach is that the *isolated* decode might provide a better phonetic match to the word pronunciation in contrast to excising a region from a lattice where language model scores can strongly influence the choice of words. For the isolated query decode experiments we compare two systems to generate the query: a unigram word loop decoder which contains the terms in question in its vocabulary and a phonetic decoder. Table V provides a comparison between the performance of the two approaches. Interestingly, the isolated decode based queries give worse results than the lattice cuts. Table VI shows the average phone error rate for the **bestq**, **randomq**, and **worstq** sets. We can see that even though the lattice cut has a higher PER, the fact that the queries generated by the lattice cuts closely model the behavior of the LVCSR system used to build the index, helps to boost the results.

TABLE III
QbyE Lattice Cut using a word-based index

Cut type	Q Nbest	P(FA)	P(miss)	ATWV
bestq	1-best	0.00004	0.492	0.466
	3-best	0.00008	0.478	0.445
	5-best	0.00009	0.475	0.435
	pruned	0.00009	0.473	0.440
worstq	1-best	0.00007	0.800	0.133
	3-best	0.00010	0.745	0.153
	5-best	0.00012	0.730	0.154
	pruned	0.00012	0.732	0.151
randomq	1-best	0.00005	0.606	0.339
	3-best	0.00009	0.573	0.336
	5-best	0.00010	0.567	0.333
	pruned	0.00010	0.571	0.328

TABLE IV
QbyE Lattice Cut using a hybrid-based index

Cut type	Q type	P(FA)	P(miss)	ATWV
bestq	1-best	0.00004	0.482	0.479
	3-best	0.00007	0.471	0.455
	5-best	0.00009	0.472	0.443
	pruned	0.00009	0.466	0.448
worstq	1-best	0.00006	0.796	0.140
	3-best	0.00010	0.740	0.160
	5-best	0.00011	0.723	0.164
	pruned	0.00012	0.730	0.148
randomq	1-best	0.00005	0.612	0.338
	3-best	0.00008	0.569	0.348
	5-best	0.00010	0.558	0.345
	pruned	0.00010	0.583	0.313

TABLE V
QbyE local decode (best cut) query nbest paths (word-index)

Query Decoding	Q Nbest	P(FA)	P(Miss)	ATWV
Word unigram	1	0.00004	0.630	0.330
	3	0.00010	0.562	0.338
Phonetic	1	0.00006	0.737	0.200
	3	0.00015	0.693	0.159

TABLE VI
Phone Error Rate for different query-generation methods

Cut Type	Best	Random	Worst
Lattice Cut	0.24	0.48	0.8
Word Local Decode	0.1	0.3	0.5
Phone Local Decode	0.2	0.4	0.7

E. 1-Best Index

We next consider the case where the indexation is done over 1-best word transcripts. The motivation for using 1-best indexes comes from the reduction in memory and disk requirements, and allows for faster retreival and indexation. We found that QbyE performs reasonably well for 1-best indexes as well. The results are presented in Table VII. The higher n-best representations of the query seem to provide more gains for the compact one best index especially for the randomq and worstq sets as compared to the denser lattice based indexes.

The overall trend in our results indicates that multi-path query representations lead to higher false alarms. The decrease in misses offsets the loss in most cases. In the next section we introduce an approach that allows us to back-off to representations of queries with fewer paths. We also present a more general 2-pass QbyE scenario which can enhanced the performance of textual queries.

TABLE VII
QBYE USING A 1-BEST (WORD) INDEX

Cut type	Q Nbest	P(FA)	P(Miss)	ATWV
bestq	1	0.00003	0.604	0.361
	3	0.00007	0.569	0.362
	5	0.00008	0.556	0.360
worstq	1	0.00006	0.828	0.116
	3	0.00010	0.773	0.132
	5	0.00011	0.755	0.136
randomq	1	0.00005	0.691	0.264
	3	0.00008	0.635	0.280
	5	0.00010	0.616	0.286

VI. QBYE IMPROVEMENTS

A. Reducing False Alarms

One common problem with our QbyE experiments with multi-path query representations is the increase in false alarms. This parallels the increase in false alarms for textual queries when using L2S systems or web-based pronunciations [4], [19]. We address this problem by identifying queries where the FA rate is expected to be high for the denser queries. For such queries we back-off to a 1-best representation. We consider the problem of identifying queries with potentially high false alarm rate by finding possible matches for the phone sequence representing the query in terms of the vocabulary of the decoder. We compose the query transducer with a inverted dictionary transducer and a unigram language model. The score of the best path of the composed transducer serves as an indicator of whether the query is a common phone sequence. The transducer allows us to detect both word subsequences and common multi-word sequences. N-best queries with the score higher than a certain threshold are replaced with 1-best queries as back off in order ro reduce false alarms. Table VIII shows that the query exclusion can siginificantly improve performance for higher n-bests.

Figure 2 shows the distribution of misses and false alarms using word indexes built from 1-best and lattices. We can

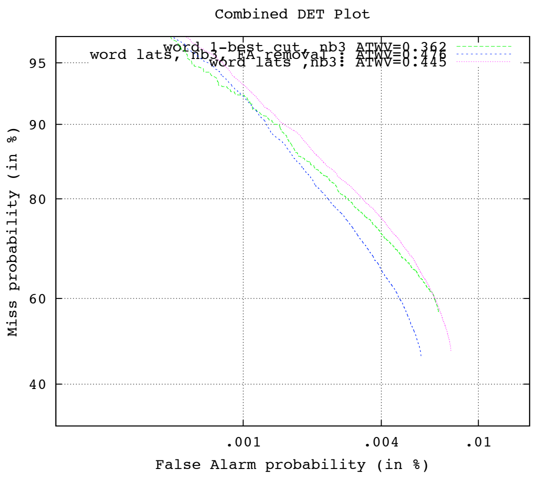

Fig. 2. DET curve comparing performance of word lattice, 1-best index, and word-lattice with exclusion for query 3-best

observe that false alarms can be reduced with the approach presented in this section. The DET plots were generated using a global threshold for the query terms. The compact one best index performs close to the lattice-index when a global threshold is employed.

TABLE VIII
QUERY EXCLUSION FOR A RANDOMLY-CHOSEN QUERY (WORD INDEX)

Q Nbest	Exclusion	P(FA)	P(Miss)	ATWV
1	No	0.00005	0.606	0.339
3	No	0.00009	0.573	0.336
	Yes	0.00008	0.565	0.360
5	No	0.00010	0.567	0.333
	Yes	0.00009	0.554	0.360

B. Two pass spoken term detection

In this section, we investigate a novel application of the query-by-example approach, namely to serve as a "query expansion" technique to improve performance for textual query retrieval. We describe a two pass approach which combines a search using textual queries and a query-by-example searching approach. Specifically, given a textual query, we retrieve the relevant instances using the baseline system described in Section IV, and use the highest scoring hit for each query as input to the QbyE system described in Section V. The second-pass serves as a means to identify richer representations of query terms incorporating phonetic confusions.

As shown in Table IX a slight improvement was obtained when merging the results from the 1st-pass and 2nd-pass compared to the baseline textual query retrieval. The merged result is obtained by simply combining the output of 1st-pass and 2nd-pass systems into a single set of instances and updating the Terms Specific Threshold accordingly to re-evaluate the binary decisions. Currently we don't weight the confidence scores obtained from first pass and second pass differently, and doing so may improve the results even further.

TABLE IX
2 PASS STD RESULTS

Data	Pron model	Q nbest	1-pass	2-pass	merged
Word	Reflex	1	0.325	0.288	0.336
		3	0.325	0.290	0.334
	L2S best-6	1	0.305	0.263	0.311
		3	0.305	0.266	0.311
Hybrid	Reflex	1	0.342	0.274	0.349
		3	0.342	0.287	0.349
	L2S best-6	1	0.341	0.278	0.344
		3	0.341	0.289	0.343

VII. CONCLUSION

In this paper, we have presented a WFST-based, query-by-example STD system and evaluated its performance on retrieving OOV queries. The key messages we wish to highlight are:

- Phone indexes derived from a hybrid (combination of word and sub-word units) LVCSR system are better than phone indexes built from a word-based LVCSR system. This is consistent with the performance of hybrid systems in OOV detection and phone recognition experiments reported in the literature.
- Queries represented using samples from the index (lattice cuts) yield better STD performance, i.e. the LVCSR system used to build the index is also used to generate a query representation.
- Increased N-best representation of queries do not translate to significant improvements in STD performance when using a lattice index. A transducer that models phonetic confusability (including insertions and deletions) can be incorporated into our framework to reduce misses, but these are implicitly captured via our multi-path queries.
- Addressing false alarm rates associated with multi-path queries can significantly improve performance over using the one-best representation of the query term. We present a method that selects a query representation using the expected counts from an unigram LM.
- Finally, QbyE can enhance the performance of text-based queries when using a two-pass approach to refine search results from a first pass based on textual queries.

REFERENCES

[1] M. Saraclar and R. W. Sproat, "Lattice-based search for spoken utterance retrieval," in *HLT-NAACL*, 2004.

[2] J. Mamou, B. Ramabhadran, and O. Siohan, "Vocabulary independent spoken term detection," in *SIGIR '07: Proceedings of the 30th annual international ACM SIGIR conference on Research and development in information retrieval*. New York, NY, USA: ACM, 2007, pp. 615–622.

[3] D. Miller, M. Kleber, C. lin Kao, and O. Kimball, "Rapid and accurate spoken term detection," in *INTERSPEECH*, 2007.

[4] D. Can, E. Cooper, A. Sethy, C. White, B. Ramabhadran, and M. Saraclar, "Effect of pronounciations on OOV queries in spoken term detection," *Acoustics, Speech, and Signal Processing, IEEE International Conference on*, vol. 0, pp. 3957–3960, 2009.

[5] T. Zhang and C.-C. J. Kuo, "Hierarchical classification of audio data for archiving and retrieving," in *ICASSP '99: Proceedings of the Acoustics, Speech, and Signal Processing, 1999. on 1999 IEEE International Conference*. Washington, DC, USA: IEEE Computer Society, 1999, pp. 3001–3004.

[6] G. Tzanetakis, A. Ermolinskiy, and P. Cook, "Pitch histograms in audio and symbolic music information retrieval," in *Proceedings of the Third International Conference on Music Information Retrieval: ISMIR*, 2002, pp. 31–38.

[7] W.-H. Tsai and H.-M. Wang, "A query-by-example framework to retrieval music documents by singer," in *ICME '04, 2004 IEEE International Conference on Multimedia and Expo, 2004*, 2004, pp. 1863–1866.

[8] T. K. Chia, K. C. Sim, H. Li, and H. T. Ng, "A lattice-based approach to query-by-example spoken document retrieval," in *SIGIR '08: Proceedings of the 31st annual international ACM SIGIR conference on Research and development in information retrieval*. New York, NY, USA: ACM, 2008, pp. 363–370.

[9] W. Shen, C. White, and T. Hazen, "A comparison of query-by-example methods for spoken term detection," in *INTERSPEECH*, 2009.

[10] C. Allauzen, M. Mohri, and Murat, "General indexation of weighted automata - application to spoken utterance retrieval," in *In Proceedings of the Workshop on Interdisciplinary Approaches to Speech Indexing and Retrieval (HLT/NAACL 2004*, 2004, pp. 33–40.

[11] M. Mohri, F. Pereira, O. Pereira, and M. Riley, "Weighted automata in text and speech processing," in *In ECAI-96 Workshop*. John Wiley and Sons, 1996, pp. 46–50.

[12] S. Parlak and M. Saraclar, "Spoken term detection for turkish broadcast news," in *ICASSP: Proceedings of the Acoustics, Speech, and Signal Processing, 2008*, 2008.

[13] C. Allauzen, M. Riley, J. Schalkwyk, W. Skut, and M. Mohri, "Openfst: A general and efficient weighted finite-state transducer library," in *CIAA*, 2007, pp. 11–23.

[14] [Online]. Available: http://www.nist.gov/speech/tests/std/

[15] H. Soltau, B. Kingsbury, L. Mangu, D. Povey, G. Saon, and G. Zweig, "The ibm 2004 conversational telephony system for rich transcription," in *ICASSP*, 2005.

[16] A. Rastrow, A. Sethy, B. Ramabhadran, and F. Jelinek, "Towards using hybrid, word, and fragment units for vocabulary independent LVCSR systems," *INTERSPEECH*, 2009.

[17] A. Rastrow, A. Sethy, and B. Ramabhadran, "A new method for OOV detection using hybrid word/fragment system," *Acoustics, Speech, and Signal Processing, IEEE International Conference on*, pp. 3953–3956, 2009.

[18] Stanley F. Chen, "Conditional and joint models for grapheme-to-phoneme conversion," in *Eurospeech*, 2003, pp. 2033–2036.

[19] E. Cooper, A. Ghoshal, M. Jansche, S. Khudanpur, B. Ramabhadran, M. Riley, M. Saraclar, A. Sethy, M. Ulinski, and C. White, "Web derived pronunciations for spoken term detection," in *SIGIR*, 2009.