Fusion of LVCSR and Posteriorgram Based Keyword Search

Leda Sarı, Batuhan Gündoğdu, Murat Saraçlar

Boğaziçi University, Bebek, Istanbul, 34342, Turkey

{leda.sari, batuhan.gundogdu, murat.saraclar}@boun.edu.tr

Abstract

The aim of this work is to improve the performance of an existing KWS system by merging the search results produced by two additional KWS systems. The existing baseline system is based on large vocabulary continuous speech recognition (LVCSR) and weighted finite state transducers (WFST). The first proposed KWS system is based on searching a symbolic WFST index which is generated by quantizing the posteriorgram representation of the audio. The second proposed KWS system is based on subsequence dynamic time warping (sDTW) algorithm which is commonly used in the query-by-example spoken term detection (QbE-STD) tasks. We also investigate using average posteriorgrams for query generation. Experimental results show that when combined with the existing KWS system, the proposed systems improve the performance of the KWS system especially for the out-of-vocabulary (OOV) queries.

Index Terms: keyword search, posteriorgram, indexation, subsequence dynamic time warping

10.21437/Interspeech.2015-259

1. Introduction

The aim of keyword search (KWS) systems is to find the written queries within an unsegmented and untranscribed audio data. The main goal is successfully detecting the time interval of the occurrence of the queries which are sequences of one or more words in a large amount of data along with a decision score. One of the most common approaches for KWS is applying text retrieval methods to the outputs of a large vocabulary continuous speech recognition (LVCSR) system [1, 2]. However, for the languages having a limited amount of transcribed audio, most of the keywords become out-of-vocabulary (OOV) queries for which we cannot obtain an LVSCR output. Therefore, the performance of the KWS system is low for such queries. In general, subword lattices are used to overcome the OOV problem [3]. Another method in dealing with OOV queries is to use a confusion model (CM) to expand the queries and allow searching for queries similar to the original ones [4, 5]. In this study, phone posteriorgrams, which are sequences of posterior probability vectors, are used for data representation. The reason for using posteriorgrams is to represent the data in a speaker-independent statistical form while still retaining phonetic information [6]. This representation is widely used in query-by-example (QbE) spoken term detection applications [6, 7, 8]. For instance, in [7], they are used along with dynamic time warping (DTW) in a QbE task. In [6], Gaussian posteriorgrams and DTW algorithm are used in spoken term detection. DTW and its derivatives like segmental DTW are used in different tasks such as pattern discovery in speech [9].

In this study, two different KWS systems are proposed in order to introduce additional detections thus achieving a better KWS performance from a combined system consisting of the proposed systems and an existing KWS system which is based on LVCSR output. In the first method, an index is generated by quantizing the posteriorgram representation and then weighted finite state transducer (WFST) based search is performed. In the second method, a frame-based search is performed using subsequence DTW (sDTW) algorithm which is frequently used in QbE applications in which the queries are also in spoken form [10].

The difference between QbE and our search systems is that the queries are in text form in our case. Therefore, it is required to obtain a posteriorgram representation for text-based queries in order to apply posteriorgram based methods. For this purpose, two different query generation approaches are followed. The first method uses a binary posteriorgram and the other one uses an average posteriorgram for the query where the averages are obtained from the training set.

The rest of the paper is organized as follows: In Section 2, the existing KWS system and the proposed systems will be introduced. In Section 3, experimental setup will be described. After presenting the results of the combined system and discussing the results in Section 4, the paper will be concluded in Section 5.

2. Methods

In this section, after briefly introducing the existing KWS system, the posteriorgram based approaches and the methods used to obtain the posteriorgram of the written query will be described. Posteriorgram is a time versus class matrix representing the posterior probability of each class for each specific time frame [7]. Each vector corresponding to a time interval denotes the probability distributions of the classes. In our KWS system, we used phonetic posteriorgrams where each class corresponds to a distinct phone. In our problem, these vectors are obtained from phone based recognizers. The basic structure of the LVCSR and the search system is described in [11]. The difference from [11] is that the basic KWS system of this study uses deep neural network (DNN) acoustic models for LVCSR [12].

2.1. Symbolic index based KWS system

In this system, the aim is to generate an index by quantizing the posteriorgram representation of the audio. The quantization is achieved by k-means clustering.

Once the posteriorgram is obtained for the dataset, k-means clustering is applied to the feature vectors of each frame. Thus, each frame is labeled with a cluster label based on the Euclidean distances between them. A WFST index is generated as described in [13] by using the symbol sequences generated from the labelings. In this WFST, input labels and output labels correspond to symbols which are the cluster labels and utterances, respectively. The weights of the arcs in the WFST consist of begin time, end time and a score. The difference between the lattice-based index and the index described here is that in the

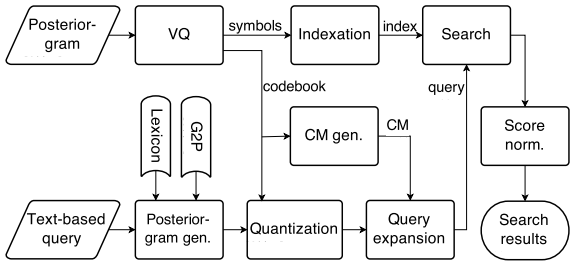

Figure 1: *Flowchart of the symbolic index based KWS system*

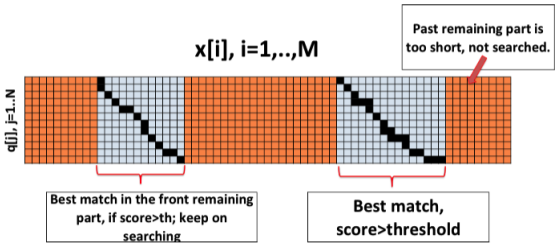

Figure 2: *KWS using sDTW*

latter each utterance is represented by a single sequence of symbols and the scores in the weights correspond to the distance between the vectors and the clusters instead of log posterior probabilities. Using WFST based index is an efficient way for both indexing and searching [14] since the index can accept any type of WFST as a query which allows flexibility in the query modeling. In this study, this property is exploited when searching for the queries expanded with a CM. The aim of a CM is to search for alternative queries by expanding the original query [14, 4]. A symbol-based CM is used in this study and the confusion weights are calculated using the distances between the cluster centers. This CM models only the substitutions so insertions and deletions are not allowed.

As an alternative to clustering the data to be searched, if phonetic alignments are available for the training data, the cluster centers can be obtained using this information. In this case, the posteriorgram frames corresponding to each phone are clustered using hierarchical k-means, thus a set of cluster centers are obtained for each phone, separately. In this case, the data to be searched is encoded by the phone label which has the closest cluster center to the data posteriorgram vector. After encoding all frames, the index is generated as described in [13]. The scores in the weights of the index is determined from the Euclidean distance between the frame and the closest center which is different than the weight structure of [13]. The simplest choice is to use a single cluster for each phone which is equivalent to using an average posteriorgram for each phone.

After finding the posteriorgram representation of the query, each frame is encoded by finding the closest cluster center. This encoding gives the label sequence from which the query WFST is obtained. Although a linear structure can be chosen for the query WFST, repeating labels can be modeled by including self-loops in the WFST.

The search step of this KWS system consists of the composition of the query WFST with the index WFST. As a result, a hit list consisting of keywords, utterance names, scores, begin and end times is obtained. If the query, CM and the index WFST are denoted by Q, CM and I, respectively, the symbolic KWS system can be summarized as in (1).

$$\text{Result} = \text{N-best}(Q \circ CM \circ I) \tag{1}$$

In (1), N-best operation is applied after composition operation to keep only the best possible paths. Figure 1 shows the block diagram of the described system.

2.2. sDTW based KWS system

In this system, KWS is done using the sDTW algorithm. sDTW is used for finding the best alignment between a short sequence and all subsequences of a considerably larger sequence. The best alignment yields the most similar subsequence given the two sequences. The similarity search is done using the averages of the distances between each frame of two sequences. Instead of calculating the average distance values for all possible frame alignments, the algorithm dynamically accumulates the best distance so far by keeping the best alignment.

In order to calculate the distances between the frames of the two sequences, cosine distance is used as the distance measure since the phone posteriorgrams are used as feature vectors where each frame is in fact a probability distribution. If we call the short sequence (query) as $\boldsymbol{q}(i), i = 1, \cdots, N$, and the long sequence (document) as $\boldsymbol{x}(j), j = 1, \cdots, M$; the cosine distance between $\boldsymbol{q}(i)$ and $\boldsymbol{x}(j)$ is:

$$d_{\cos}(\boldsymbol{q}(i), \boldsymbol{x}(j)) = -\log\left(\frac{\boldsymbol{q}(i).\boldsymbol{x}(j)}{|\boldsymbol{q}(i)||\boldsymbol{x}(j)|}\right) \tag{2}$$

An accumulated distance matrix $\boldsymbol{A}$, with size NxM, is used in dynamic programming to avoid the calculation of all possible distance averages. The (i, j)th element of $\boldsymbol{A}$ can be expressed as the total distance of the best alignment between $\boldsymbol{q}$ and $\boldsymbol{x}$ up to ($\boldsymbol{q}(i)$,$\boldsymbol{x}(j)$). Similarly, the $\boldsymbol{L}$ matrix stores the length of the best alignment up to ($\boldsymbol{q}(i)$,$\boldsymbol{x}(j)$).

The (i, j)th element of the $\boldsymbol{B}$ matrix, called the beginning matrix, stores the beginning frame of the subsequence of $\boldsymbol{x}$, whose alignment with $\boldsymbol{q}$ ends at (i, j). As the time progresses along the sequences, the optimal beginning frame information is transferred dynamically. The sDTW algorithm is shown in Algorithm 1.

Algorithm 1 sDTW

for $i = 1$ to N, $j = 1$ to M **do**
  if i=1 **then**
    $\boldsymbol{A}(1, j) = d_{\cos}(\boldsymbol{q}(1), \boldsymbol{x}(j))$
    $\boldsymbol{L}(1, j) = 1, \boldsymbol{B}(1, j) = j$
  else if j=1 **then**
    $\boldsymbol{A}(i, 1) = \sum_{k=1}^{i} d_{\cos}(\boldsymbol{q}(k), \boldsymbol{x}(1))$
    $\boldsymbol{L}(i, 1) = i, \boldsymbol{B}(i, 1) = 1$
  else
    $\Omega = \{(i, j-1), (i-1, j), (i-1, j-1)\}$
    $(r, s) = \text{argmin}_{\forall (p,q) \epsilon \Omega} \frac{\boldsymbol{A}(p,q) + d_{\cos}(\boldsymbol{q}(i), \boldsymbol{x}(j))}{\boldsymbol{L}(p,q)+1}$
    $\boldsymbol{A}(i, j) = \boldsymbol{A}(r, s) + d_{\cos}(\boldsymbol{q}(i), \boldsymbol{x}(j))$
    $\boldsymbol{L}(i, j) = \boldsymbol{L}(r, s) + 1, \boldsymbol{B}(i, j) = \boldsymbol{B}(r, s)$
  end if
end for

The ending frame of the best subsequence of $\boldsymbol{x}$ can be found as $\text{argmin}(\boldsymbol{A}(N, :))$. A search score that favors longer alignments can be calculated as follows:

$$\text{score}(j) = 1 - \frac{\boldsymbol{A}(N, j)}{\boldsymbol{L}(N, j)} \qquad j = 1, \cdots, M. \tag{3}$$

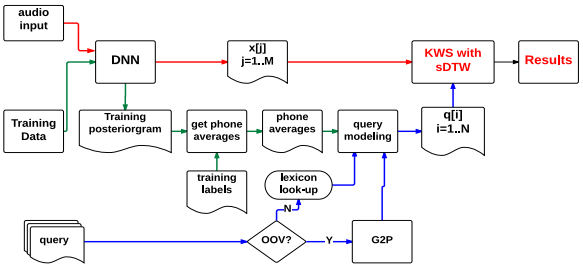

Figure 3: *Flowchart of the sDTW based KWS system*

The subsequence of $\boldsymbol{x}$, that has the best similarity to the query is the one yielding the maximum score. The subsequences whose score is above a certain threshold is marked as hit. The beginning and ending frames of the hit can be calculated as in (4).

$$\text{hit} = \boldsymbol{x}(\boldsymbol{B}(N, \text{argmax}(\text{score})) : \text{argmax}(\text{score})) \quad (4)$$

As stated above, $\boldsymbol{B}(N, j)$ points at the best starting point of the subsequence that ends at j.

There may be multiple occurrences of the query in the document file. In order to search for all matches that are above the threshold, a recursive search algorithm is implemented as follows: After the best match is found as in (4) and if it is above the threshold, the hit is removed from the document and the remaining parts are searched for their best matches. This procedure is continued until the best matches in all document parts are below the threshold or the document parts are too short to be searched. This recursive search is visualized in Figure 2. The block diagram of the sDTW KWS system can be seen in Figure 3.

2.3. Posteriorgram representation of the written query

In the proposed KWS systems, the query and the data to be searched should be in the same form. Therefore, the written queries should be converted to posteriorgrams which is an important step for the success of the KWS system. Initially, each query is represented by a sequence of phones. For IV queries this is achieved simply by a lexicon lookup whereas for OOV queries, grapheme-to-phoneme (G2P) conversion is applied [15]. Then the query posteriorgram is constructed by associating one or more posteriorgram vectors with each phone in these sequences. Two different methods are used to generate the query posteriorgrams. In the first method, a vector corresponding to a phone is constructed such that it contains only a 1 at the position which corresponds to the phone and 0 elsewhere. In this representation, these vectors are repeated for a fixed number of times. In addition, an optional silence is used for multi-word queries.

In the second method, an average posterior vector for each phone class is calculated using the alignments of a subset of the training data for which transcriptions are available. This is achieved simply by averaging the posterior vectors of the frames labeled with that phone in the training posteriorgrams. Average duration of each phone is also extracted using the counts of the consecutive frames labeled with each phone over the training corpus. Then the queries are modeled by using the average posteriorgram of each phone in the sequence.

Figure 4 shows the two different posteriorgrams of a sample query generated by the two methods. In this figure, the sparse vectors are represented by the pairs of indices and values which are greater than 0, the number next to a vector indicates the

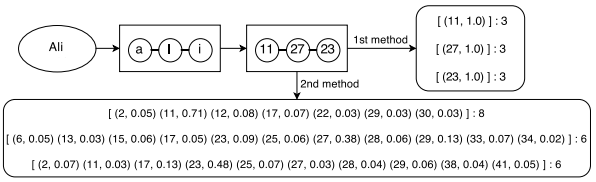

Figure 4: *Two methods for query posteriorgram generation*

number of times the vector is repeated in the query posteriorgram. The conversion to the phone sequences common to both methods, that is obtaining the sequence '11-27-23' for the keyword 'Ali'. After this point, the first method generates a binary posteriorgram by replicating the same vector for a fixed number of times whereas in the second method each phone is represented with an average posteriorgram and its length is determined by the average duration information calculated from the training data.

3. Experiments

Experiments are performed on the IARPA Babel Turkish limited language pack (LimitedLP) dataset which contains conversational telephone speech. KWS evaluation is performed using 307 queries 88 (28.67%) of which are OOV.

For the LVSCR based KWS system, the Babel KWS setup in Kaldi toolkit is used[1]. This pipeline is based on a WFST index generated from LVCSR lattices and the LVSCR system uses a DNN with p-norm activations [12]. For posteriorgram generation, the same DNN is used. A forward pass of the data through this network results in a higher dimensional state level posteriorgram. Using this output we obtain the phone posteriorgram by accumulating the posterior values of states corresponding to each phone which leads to a 43 dimensional posteriorgram. Kaldi toolkit [16] is also used in indexation and WFST based search of the symbolic index based system.

The KWS systems are evaluated according to term-weighted value (TWV) which is a frequently used measure in determining the performance of a KWS system [17] and it is calculated as follows:

$$\text{TWV}(\theta) = 1 - \frac{1}{|\mathcal{Q}|} \sum_{q \in \mathcal{Q}} P_{\text{miss}}(q, \theta) + \beta P_{\text{FA}}(q, \theta) \quad (5)$$

In this expression, $\mathcal{Q}$ denotes the set of queries, $|\mathcal{Q}|$ denotes the number of queries in $\mathcal{Q}$, $P_{\text{miss}}(q, \theta)$ and $P_{\text{FA}}(q, \theta)$ show the miss and false alarm probabilities for a query q when a threshold θ is applied to the search scores.

The maximum value that TWV can get is 1 and it corresponds to correctly detecting all of the queries, with no false alarms. For systems returning an empty hit list, TWV is 0 and TWV can get negative values. The maximum TWV obtained from all possible threshold values is called maximum TWV (MTWV). In this study, TWV of the search results is calculated using the NIST F4DE tool. Other performance measures used in this study are optimal TWV (OTWV) and supremum TWV (STWV). OTWV is calculated using a separate threshold for each query that gives the MTWV for the same query. STWV is found by taking search scores for the correct detections and false alarms as 1 and 0, respectively [18]. We also calculate the TWV over all queries where two different thresholds are ap-

[1] http://svn.code.sf.net/p/kaldi/code/trunk/egs/babel/s5c

Table 2: *MTWV, MTWV*, OTWV and STWV results from L: LVSCR lattice based system, L+PP: Combination of three systems (both IV and OOV), L+$\mathrm{PP_{OOV}}$: Combination of L (IV and OOV) and the OOV results from the proposed systems*

System	MTWV			MTWV*	OTWV			STWV		
	All	IV	OOV	All	All	IV	OOV	All	IV	OOV
L	0.3745	0.4500	0.1887	0.3764	0.4157	0.4896	0.2240	0.6624	0.7567	0.4220
L + PP	0.3625	0.4233	0.2250	0.3674	**0.4845**	**0.5390**	**0.3455**	**0.7115**	**0.7725**	**0.5560**
L + $\mathrm{PP_{OOV}}$	**0.3820**	**0.4500**	**0.2250**	**0.3866**	0.4490	0.4896	0.3455	0.7002	0.7567	0.5560

plied to IV and OOV queries that give the MTWVs for them separately, which we call MTWV*.

4. Results and Discussion

In this section, after presenting the performances of the proposed systems individually, we describe the system combination method and then give the results for the merged systems. The baseline LVCSR based system that our proposed systems are merged with achieves a MTWV of 0.3745 over all queries. The MTWV of the proposed systems are shown in Table 1. In the first column of this table, 'Sym' and 'Sym-Sup' are results from the symbolic index based KWS system. The former directly uses the test data and the latter uses training posteriorgrams together with phone labels for finding the cluster centers. Results from sDTW based KWS system are denoted by 'sDTW' in the first column. The second column of the table indicates two types of query posteriorgram generation methods described in Section 2.3, namely, binary posteriorgram ('Bin') and average posteriorgram ('Avg'). The third column shows the normalization methods for which we use either sum-to-one normalization ('STO') or a linear mapping of scores to 0-1 range ('Lin'). The last three columns respectively show MTWVs calculated over all queries, using IV queries and OOV queries only.

Table 1: *MTWV for individual systems*

System	Query	Norm.	All	IV	OOV
Sym	Bin	STO	0.0213	0.0260	0.0098
Sym	Avg	STO	0.0213	0.0266	0.0098
Sym-Sup	Bin	STO	**0.0227**	**0.0237**	**0.0228**
Sym-Sup	Avg	STO	0.0259	0.0297	0.0216
sDTW	Bin	STO	**0.1036**	**0.1076**	**0.0940**
sDTW	Avg	STO	0.0780	0.0903	0.0478
sDTW	Bin	Lin	0.1174	0.1127	0.1388
sDTW	Avg	Lin	0.1128	0.1125	0.1217

In addition to the performances of individual systems, the performance of the combination of system outputs is also evaluated. For this purpose, IV and OOV search results from the baseline LVCSR lattice based system ('L') and the proposed systems are merged. In order to obtain the merged scores for each hit, the scores of the corresponding hits from different systems are summed and then normalized by the number of systems that contributed to the sum, i.e. the number of systems having a non-zero score for this hit. The start time and duration are taken from the highest-scoring hit. This method is similar to using the CombMNZ method described in [19].

The MTWV, MTWV*, OTWV and STWV results for the LVCSR lattice based system alone ('L') and the merged systems are shown in Table 2. In this table, L+PP is the combination of the baseline system 'L' with the proposed systems 'Sym-Sup + Bin + STO' and 'sDTW + Bin + STO' shown in bold in Table 1. Although 'Lin' normalization improves the results for the sDTW based system, it is not useful in combination. Therefore, we choose 'STO' normalization. From L+PP row of the table, it can be seen that there is 19% relative improvement of the MTWV calculated using OOV queries only as compared to the baseline results although the MTWV over all queries decreased due to the mismatch between scores from different systems. The score mismatch problem can also be observed from the increase in the OTWV and STWV results in the combined systems because when the thresholds differ for each query, we can get significantly higher number of true detections for both IV and OOV queries. Upon these observations, L+$\mathrm{PP_{OOV}}$ scheme is used where we combine the baseline results with the OOV results obtained from proposed systems. When we calculate the performance measures over OOV queries separately, the gain for the OOV queries can be clearly seen. In addition, we observe 1% absolute improvement in MTWV* which is calculated using two different thresholds for IV and OOV queries.

5. Conclusions and future work

This research suggests that posteriorgram based KWS systems yield promising results for combination with LVCSR based systems for KWS tasks. Although simple methods are used to model query posteriorgrams given their graphemic forms, even with the *pseudo* query posteriorgrams with 1s and 0s or template repetitions, the proposed systems were able to conduct successful searches. Furthermore, when combined with the existing LVCSR based system, 19% relative improvement in the performance is observed. This improvement corroborates the observation that developing diverse and complementary systems is a way of improving the performance of a KWS system.

We think that further improvements can be achieved if real posteriorgram parts are used as queries when possible. If examples from the training data can be used to construct the words of the given query, this would yield a more realistic search pattern for the query.

6. Acknowledgements

This study uses the IARPA Babel Program base period language collection release babel105b-v0.4, supported by the Intelligence Advanced Research Projects Activity (IARPA) via Department of Defense U.S. Army Research Laboratory (DoD/ARL) contract number W911NF-12-C-0012. The U.S. Government is authorized to reproduce and distribute reprints for Governmental purposes notwithstanding any copyright annotation thereon. Disclaimer: The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies or endorsements, either expressed or implied, of IARPA, DoD/ARL, or the U.S. Government.

7. References

[1] 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.

[2] M. Saraclar and R. Sproat, "Lattice-based search for spoken utterance retrieval," in *HLT-NAACL*, 2004, pp. 129–136.

[3] K. Ng and V. W. Zue, "Subword-based approaches for spoken document retrieval," *Speech Communication*, vol. 32, no. 3, pp. 157–186, 2000.

[4] P. Karanasou, L. Burget, D. Vergyri, M. Akbacak, and A. Mandal, "Discriminatively trained phoneme confusion model for keyword spotting," in *Interspeech*, 2012, pp. 2434–2437.

[5] M. Saraclar, A. Sethy, B. Ramabhadran, L. Mangu, J. Cui, X. Cui, B. Kingsbury, and J. Mamou, "An empirical study of confusion modeling in keyword search for low resource languages," in *IEEE Workshop on Automatic Speech Recognition and Understanding (ASRU)*, 2013, pp. 464–469.

[6] Y. Zhang, "Unsupervised speech processing with applications to query-by-example spoken term detection," Ph.D. dissertation, Massachusetts Institute of Technology, 2013.

[7] T. J. Hazen, W. Shen, and C. White, "Query-by-example spoken term detection using phonetic posteriorgram templates," in *IEEE Workshop on Automatic Speech Recognition and Understanding (ASRU)*, 2009, pp. 421–426.

[8] Y. Zhang and J. R. Glass, "Unsupervised spoken keyword spotting via segmental DTW on gaussian posteriorgrams," in *IEEE Workshop on Automatic Speech Recognition and Understanding (ASRU)*, 2009, pp. 398–403.

[9] A. S. Park and J. R. Glass, "Unsupervised pattern discovery in speech," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 16, no. 1, pp. 186–197, 2008.

[10] F. Metze, X. Anguera, E. Barnard, M. Davel, and G. Gravier, "Language independent search in MediaEval's spoken web search task," *Computer Speech & Language*, vol. 28, no. 5, pp. 1066–1082, 2014.

[11] G. Chen, S. Khudanpur, D. Povey, J. Trmal, D. Yarowsky, and O. Yilmaz, "Quantifying the value of pronunciation lexicons for keyword search in low resource languages," in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2013, pp. 8560–8564.

[12] J. Trmal, G. Chen, D. Povey, S. Khudanpur, P. Ghahremani, X. Zhang, V. Manohar, C. Liu, A. Jansen, D. Klakow *et al.*, "A keyword search system using open source software," in *Proceedings of the IEEE Workshop on Spoken Language Technologies (SLT)*, 2014.

[13] D. Can and M. Saraclar, "Lattice indexing for spoken term detection," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 19, no. 8, pp. 2338–2347, 2011.

[14] C. Parada, A. Sethy, and B. Ramabhadran, "Query-by-example spoken term detection for oov terms," in *IEEE Workshop on Automatic Speech Recognition and Understanding (ASRU)*, 2009, pp. 404–409.

[15] M. Bisani and H. Ney, "Joint-sequence models for grapheme-to-phoneme conversion," *Speech Communication*, vol. 50, no. 5, pp. 434 – 451, 2008.

[16] D. Povey, A. Ghoshal, G. Boulianne, L. Burget, O. Glembek, N. Goel, M. Hannemann, P. Motlíček, Y. Qian, P. Schwarz *et al.*, "The Kaldi speech recognition toolkit," in *IEEE Workshop on Automatic Speech Recognition and Understanding (ASRU)*, Dec. 2011.

[17] J. G. Fiscus, J. Ajot, J. S. Garofolo, and G. Doddingtion, "Results of the 2006 spoken term detection evaluation," in *Proceedings of ACM SIGIR Workshop on Searching Spontaneous Conversational*, 2007, pp. 51–55.

[18] "KWS14 keyword search evaluation plan," http://www.nist.gov/itl/iad/mig/upload/KWS14-evalplan-v11.pdf.

[19] J. Mamou, J. Cui, X. Cui, M. J. Gales, B. Kingsbury, K. Knill, L. Mangu, D. Nolden, M. Picheny, B. Ramabhadran *et al.*, "System combination and score normalization for spoken term detection," in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2013, pp. 8272–8276.