Low Resource Keyword Search With Synthesized Crosslingual Exemplars

Bolaji Yusuf, Batuhan Gundogdu, and Murat Saraclar, *Member, IEEE*

***Abstract*—The transfer of acoustic data across languages has been shown to improve keyword search (KWS) performance in data-scarce settings. In this paper, we propose a way of performing this transfer that reduces the impact of the prevalence of out-of-vocabulary (OOV) terms on KWS in such a setting. We investigate a novel usage of multilingual features for KWS with very little training data in the target languages. The crux of our approach is the use of synthetic phone exemplars to convert the search into a query-by-example task, which we solve with the dynamic time warping algorithm. Using bottleneck features obtained from a network trained multilingually on a set of (source) languages, we train an extended distance metric learner (EDML) for *four* target languages from the IARPA Babel program (which are distinct from the source languages). Compared with a baseline system that is based on automatic speech recognition (ASR) with a multilingual acoustic model, we observe an average term weighted value improvement of 0.0603 absolute (74% relative) in a setting with only 1 h of training data in the target language. When the data scarcity is relaxed to 10 h, we find that phone posteriors obtained by fine-tuning the multilingual network give better EDML systems. In this relaxed setting, the EDML systems still perform better than the baseline on OOV terms. Given their complementary natures, combining the EDML and the ASR-based baseline results in even further performance improvements in all settings.**

***Index Terms*—Spoken term detection, distance metric learning, exemplar matching, dynamic time warping, multilingual features, query by example, low resource keyword search, transfer learning.**

I. Introduction

THE proliferation of devices capable of recording speech, coupled with the decreased cost of storage, has resulted in a large trove of spoken data in the form of audiovisual lectures, broadcast news, podcasts, public forum recordings, interviews, interrogations, call-center recordings etc. The ability to search such archives accurately and efficiently would assist significantly in research, media consumption and law enforcement. One of the areas of research that has emerged to achieve this goal is keyword search, also known as spoken term detection (STD), which tries to solve the problem of finding the location of a short query within a large spoken document. The problem tackled in this paper is a form of STD in which the query is provided as a string of words in textual form and the system is expected to return a list of locations within the document hypothesized to contain the query term, along with a relevance score and binary (YES/NO) decision for each location.

The common approach to keyword search involves transcribing the audio archive into words with a large vocabulary continuous speech recognition (LVCSR) system and then conducting textual retrieval on the transcription. To avoid the drop in recall rate associated with searching on one-best transcriptions in the presence of recognition errors, multi-hypothesis lattices [1] or confusion networks [2] are used to generate the search index. The LVCSR backends used for keyword search require hundreds of hours of transcribed speech for training; therefore, outside of the small fraction of the world's languages that are resource rich, the performance of ASR systems deteriorates, and with it, that of ASR-based KWS. An extreme form of this deterioration occurs when the query contains a word that is not in the vocabulary used to train the LVCSR. Since the output space of the LVCSR is limited to its lexicon, the search index from word-based LVCSR simply cannot contain such words and so, regardless of actual occurrence, the system returns no results. Common sources of OOV words include proper nouns, loan words and neologisms, as well as inflections in morphologically-rich languages.

Given that search is an open vocabulary problem by nature, several solutions have been proposed to the OOV problem. One such solution is to construct lattices of subword units such as phones, syllables or morphs [3]–[5] which leads to a relaxation of the lexical constraint that enables such systems to handle OOV words. This relaxation comes with the cost of increased false alarms; therefore systems that utilize word lattices for in-vocabulary (IV) terms and subword lattices for OOV term retrieval have also been proposed [6], [7]. Another method of alleviating the effect of OOV terms involves lexicon expansion prior to decoding [8]. By collecting words from external sources and generating automatic pronunciations for them, this approach increases the size of the lexicon and thus, reduces the incidence of OOV words by making some of them IV in effect. Query expansion is another OOV handling method which removes the need for the multiple decoding passes in subword/hybrid lattices [9], [10]. Using a confusion model, OOV search terms are replaced with similar-sounding IV "proxy" ones, which are then searched. In [11], it was shown that the proxy method resulted in similar OOV performance when compared to a subword

Manuscript received November 2, 2018; revised March 3, 2019 and April 8, 2019; accepted April 8, 2019. Date of publication April 15, 2019; date of current version May 7, 2019. This work was supported in part by the Scientific and Technological Research Council of Turkey (TUBITAK) under Project No. 116E076. The associate editor coordinating the review of this manuscript and approving it for publication was Dr. Kong Aik Lee. *(Corresponding author: Bolaji Yusuf.)*

B. Yusuf and M. Saraclar are with the Department of Electrical and Electronics Engineering, Bogazici University, Istanbul 34342, Turkey (e-mail: bolaji.yusuf@boun.edu.tr; murat.saraclar@boun.edu.tr).

B. Gundogdu is with the Department of Electrical and Electronics Engineering, National Defense University, Naval Academy, Istanbul 34342, Turkey (e-mail: mbgundogdu@dho.edu.tr).

Digital Object Identifier 10.1109/TASLP.2019.2911164

system that does phonetic search on a converted word lattice [5], [12]. It should be noted though that in [13], it was shown that using a syllabic lattice for the phonetic search gave better OOV performance than using a word lattice.

We recently proposed an approach to KWS that skips the ASR step by converting the problem into a QBE search [14]. Using a DNN acoustic model, a posteriorgram representation is obtained and the posterior features belonging to each phoneme class are used to compute a centroid vector; for any query, the centroids of its phonemes are concatenated to replace the spoken query in QBE-STD and dynamic time warping (DTW)–based search is conducted. With this system, there is no drop off in performance from IV to OOV terms, since the search involves frame-level matching of acoustic sequences, without any language model.

Exploiting multilingual information in acoustic modeling has recently produced promising results in low resource KWS [15]. It has been shown that training a neural network with a shared bottleneck layer helps utilize the multilingual acoustic information for better KWS performance.

In this work, we investigate the use of multilingual bottleneck features in very low resource conditions. With one hour of transcribed training data for the target language, the improvement brought by crosslingual acoustic modeling in LVCSR-based KWS systems is limited due to an increase in the OOV rate. Therefore, in this paper we follow the DTW-based KWS methodology similar to [14] since we expect a high prevalence of OOV terms in our very low resource target. The main difference is the use of crosslingually[1] ported representations to make up for the scarcity of acoustic data. To this end, we modify the metric learner, which was proposed for posteriorgrams, in order to make it work for the richer BNF by introducing a subnetwork that transforms the document into a more easily separable subspace - a modification that also improves the performance on posteriorgrams. For our target of very low resource keyword search (one hour of target language training data), we show that the proposed system with BNF outperforms the LVCSR-based crosslingual baseline by a considerable margin (0.0603 average ATWV differential). With more (10 hours) target language training data, although the LVCSR-based baseline system is unsurprisingly better on IV terms, the proposed system retains a substantial superiority over the baseline on OOV terms. Furthermore, we conduct experiments to quantify the impact of the multilingual pretraining on the quality of posteriorgrams and find that compared to using monolingual posteriorgrams, building our system with multilingually-pretrained posteriorgrams results in an MTWV improvement of 0.0971 on the development set.

The main contributions of this work can be listed as follows:

- Crosslingual bottleneck features are used to produce (context independent) phone state centroid vectors which are concatenated to form synthetic query exemplars for a template matching–based search which outperforms the LVCSR-based KWS methodology in a very low resource setting.
- The asymmetric Siamese network–based DML topology proposed in [14] is restructured to facilitate operating with bottleneck features, rather than just posteriorgrams, with the introduction of a subnetwork to optimize the document representation.
- The use of posteriorgrams obtained from a finetuned multilingual network for DTW-based search. Although such finetuning has been used in LVCSR-based approaches, the same has not been done for DTW-based KWS.
- The effect of training corpus size is quantified by experiments conducted under three conditions with 1 hour, 3 hours, and 10 hours respectively on four different languages from the IARPA Babel Program.
- A robust methodology is proposed to handle KWS with very limited training. With one hour of training data, the proposed methodology improves the performance of a state-of-the-art LVCSR-based KWS systems by 74%.

[1]Although the term crosslingual is sometimes used in the literature to mean that the transferred representation is trained on a single other language, we use it to mean that the representation is trained without examples from the target language.

II. Related Work

A. Posteriorgram-Based Keyword Search

The use of posteriorgrams for keyword search was proposed recently as an alternative and complementary method to LVCSR-based keyword search. Following the success of DTW in query-by-example tasks [16]–[19] even in the extreme conditions of the MediaEval campaign [20]–[24], similar systems were proposed for KWS [25], [26]. Given the obvious difference between the two tasks: the queries in QBE are provided in audio form while those in KWS are provided as text, the use of query "pseudo-posteriorgrams" is central to the application of DTW to KWS. From the force-aligned training data, averages and duration statistics are computed so that each query is represented as a concatenation of the average vectors of its constituent phonemes, each repeated by its average duration in the training alignment. This query pseudo-posteriorgram is then searched in the document posteriorgram with the subsequence dynamic time warping (sDTW) algorithm using variants of the cosine distance.

Further improvements were made in [27] where the cosine distance is replaced by a Siamese neural network that finds an optimal distance metric between a pair of frames. Once again, averages are used to represent each phoneme. In [28], a joint distance metric learning (JDML) network is proposed to both optimize the distance function and replace the ad-hoc average query model with a centroid that is learned in tandem with the distance.

B. Multilingual Networks for Low Resource KWS

Training accurate ASR and KWS systems is typically a data-intensive process. Due to the scarcity of transcribed data endemic in low resource languages, the ability to transfer knowledge from other, resource-rich, languages is an attractive area of research. In particular, the use of multilingually trained neural networks to improve ASR and KWS in target languages has been studied extensively.

The general framework involves using a network with layers shared across languages. Since there are multiple languages involved in the training, each with its own set of phonemes, there is a question of how to determine what the training labels (and output layer) should be. One approach is to use a single softmax layer at the output of the network with labels covering the phoneme sets of all the languages [29]–[32]. Another approach is to train with language specific softmax layers either by showing the network languages in a sequential manner [33] or by interleaving batches of samples from different languages [34]–[36] in a so-called "block-softmax". Unlike the sequential training, the block-softmax has the advantage of being unbiased towards any training language (given similar amounts of training data). The multilingual networks were used extensively in the IARPA Babel program [37] to improve low-resource ASR and KWS performance [15], [38]–[40].

The difference between our work and these previous applications is that while they use the multilingual data as a way to improve ASR (and KWS by proxy), we use an ASR-free, DTW-based KWS methodology. Although there has been work on data selection to minimize the amount of multilingual training data with minimal loss in performance [41], [42], we do not investigate this as we consider the multilingual network to be a "gray-box" where we have access to the network's weights but not its training data.

III. System Description

The overall approach proposed in this work involves using dynamic time warping to conduct keyword search in low resource languages. Since this approach entirely involves frame-level matching of acoustic sequences, it is more robust to the scarcity of linguistic resources. By using a transferred representation, we are able to leverage acoustic knowledge from other languages in building the models for the language of interest. In this way, we are able to alleviate the deleterious effects that acoustic and linguistic data scarcity have on keyword search performance.

A. Multilingual Features

The collection to be searched is represented as a sequence of feature vectors on which the dynamic time warping search is conducted. These features are initially derived from networks with multiligual backends and then further optimized for the search task. We consider two initial feature kinds: the bottleneck features which are obtained from a multilinugal neural network, and phone posterior features, which are derived by finetuning the multilingual network with the limited training data available for the language of interest.

1) Bottleneck Features: The bottleneck features represent each frame as the activations of a hidden layer of a multilingual deep neural network with a bottleneck layer that contains significantly fewer units than the other layers. The hidden layers of this generator network are shared across the source languages up till the bottleneck layer; on top of these layers, each language has an extra hidden layer and softmax output layer specific to it.

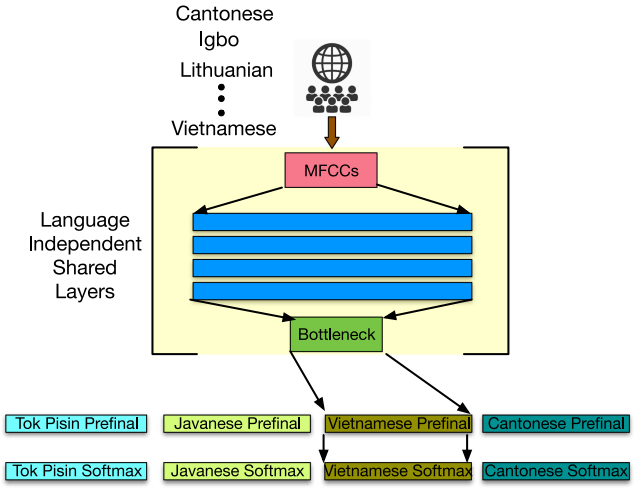

Fig. 1. The multilingual bottleneck network. The time delay connections are omitted for simplicity.

This generator network is trained with MFCC features to predict language-specific tied-triphone labels. The training involves pooling the features from the source languages along with their force-aligned labels and interleaving them. Each training batch contains samples from a single language whose output layer is active while the other languages' output layers are frozen with gradients set to zero. This training scheme enables us to obtain a discriminative, language-independent representation.

The BNF extractor used is a time-delay neural network (TDNN) with rectified linear units (ReLU) as activation functions. The TDNN layers facilitate the incorporation of longer temporal contexts than DNNs and, with clever subsampling [43], some of the latency associated with recurrent architectures is avoided while maintaining coverage of the entire input temporal context. A 42-dimensional bottleneck layer is used and the training is done with Kaldi's [44] multilingual[2] recipe.

2) Phone Posteriorgram: While the multilingual bottleneck representation provides a compact way to transfer knowledge from the source languages, it is intuitive that further improvements can be achieved from further target language-specific training since the search task (by virtue of having written queries) is language dependent. To this end, for each target language, the multilingual TDNN, truncated at the bottleneck layer (leaving only the shared layers), is extended with a new output layer which is then finetuned to predict that language's senones. From the output of this network, we get the phone posteriorgram representation. The posteriorgram represents each frame as a vector of phone probabilities which are computed by summing up the senone probabilities associated with each phoneme.

B. DTW and Query Model for Keyword Search

For a given query vector sequence, $\mathcal{X} := (\mathbf{x}^1, \mathbf{x}^2, \ldots, \mathbf{x}^N)$, the sDTW algorithm [45] finds a matching subsequence of the document vector sequence, $\mathcal{Y} := (\mathbf{y}^1, \mathbf{y}^2, \ldots, \mathbf{y}^M)$. The sDTW

[2]babel_multilang/s5

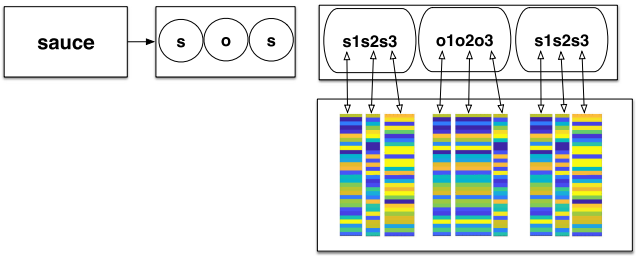

Fig. 2. A simple query model. The learned representations of each phoneme's states are concatenated to form the query sequence.

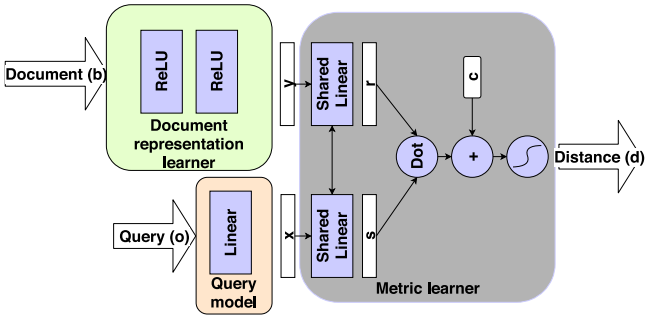

Fig. 3. The extended distance metric learning network.

algorithm returns an optimal alignment path, $\mathbf{\Pi}$ between $\mathcal{X}$ and subsequences of $\mathcal{Y}$ as well as a confidence score computed from the average distortion incurred along the optimal path:

$$\text{score} = 1 - \frac{1}{\text{length}(\mathbf{\Pi})} \sum_{n_i, m_i \epsilon \mathbf{\Pi}} d(\mathbf{x}_{n_i}, \mathbf{y}_{m_i}). \tag{1}$$

Since the query is provided as text, which is a different modality from the document, and conducting DTW-based search requires that both sequences be in the same modality, the query is artificially modeled to resemble the document representation. The procedure for this is as follows:

1) The query is converted into a sequence of phonemes using the Sequitur grapheme-to-phoneme (G2P) [46] conversion toolkit. Each phoneme is further decomposed into context-independent HMM states (five non-phonemic states and three states for each phoneme). Although the sequence of phones can be used directly, the sub-phonemic states allow for finer discrimination in the optimized dynamic time warping.
2) Using the forced alignment of the available transcribed training data, a set of clusters along with duration statistics is collected for each state. For each state, a centroid vector representative of its cluster of features is learned from the data. For a given query, the state centroid vectors of its constituent phonemes are concatenated. We experimentally verified that repeating each vector by its average phone duration (obtained from training alignments) improves the term weighted value by an average of 0.0428 across languages. Further details on the centroid computation are deferred to Section III-C.

This procedure can be be seen in Figure 2.

C. Optimizing the Search

The dynamic time warping algorithm uses a sequence of locally optimal steps to compute the path and distortion score. It can be noted from Equation (1) that the score is dependent on:

- The document frame representation, $\mathbf{y}_{m_i}$,
- the query frame representation, $\mathbf{x}_{n_i}$, and
- the distance function, $d(.,.)$, used.

To optimize each of these, we propose using a unified framework implemented as an extended distance metric learning neural network. The EDML network is based on the simultaneous optimization of three parts:

1) The distance metric learner learns a function called the sigma distance [27] based on a weighted inner product between its inputs. For a query frame vector, $\mathbf{x}$, and a document frame vector, $\mathbf{y}$, the distance function, parameterized by the weight matrix, $\mathbf{W}$, and bias value, c, is given by:

$$d_\sigma(\mathbf{x}, \mathbf{y}) = \sigma(\mathbf{x}^T \mathbf{W}^T \mathbf{W} \mathbf{y} + c) \tag{2}$$

where $\sigma(\cdot)$ is the logistic sigmoid function. The parameters of this sub-network are optimized to discriminate between the phone states so that, ideally, the network outputs:

$$\hat{d}_\sigma(\mathbf{x}, \mathbf{y}) = \begin{cases} 0, & \text{if class}(\mathbf{x}) = \text{class}(\mathbf{y}) \\ 1, & \text{if class}(\mathbf{x}) \neq \text{class}(\mathbf{y}). \end{cases} \tag{3}$$

Note that the terms distance and metric are used in a strictly semantic sense, i.e., to describe a function that outputs low scores for similar vectors and high scores for dissimilar vectors, this function does not satisfy the mathematical axioms of metrics.

2) The document representation is learned with a series of nonlinear neural network layers. The sub-network responsible for this takes as input a feature vector (bottleneck or posteriorgram), $\mathbf{b}$ and outputs $\mathbf{y}$. This introduction of this subnetwork is the difference between EDML and the JDML network in [14]. When the input features are posterior features and this sub-network is omitted, the retrieval performance loss is modest; however, when bottleneck features are used, its inclusion is essential to getting good search performance [47]. This discrepancy can be attributed to the rich, unstructured nature of the bottleneck features with non-linearly-separable classes. Overall, this part of the network serves to simplify the document representation and obtain more easily separable class representations.

$$\mathbf{y} = \mathcal{F}(\mathbf{b}). \tag{4}$$

The function ($\mathcal{F}$) is a stack of ReLU layers.

3) The third sub-network generates state models. Its input is a one-hot vector, $\mathbf{o}$, representing the state being modeled. This vector is transformed into the state centroid through a simple matrix multiplication:

$$\mathbf{x} = \mathbf{V}\mathbf{o} \tag{5}$$

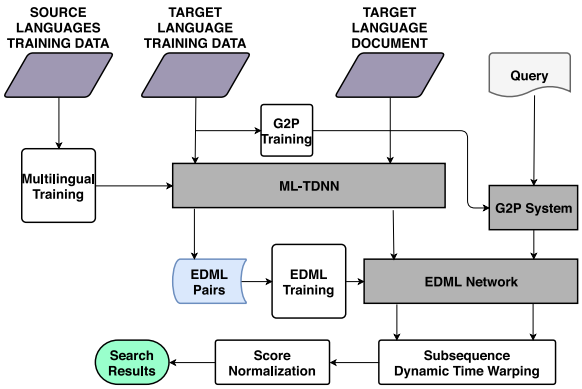

Fig. 4. Flowchart of the proposed system.

The query and document representation learners are prepended to the distance learner as shown in Figure 3 to form a unitary network to be optimized.

1) Training the Extended Distance Metric Learner: The whole network is trained with the backpropagation algorithm. On the forward pass mini-batches of input pairs, $(\mathbf{b}, \mathbf{o})$, are fed into the network and the operations in Equations (5), (4) and (2) are performed (in that order) to obtain the dissimilarity between the representations $(\mathbf{y}, \mathbf{x})$ of the inputs. As the objective function, we use the cross-entropy between the output dissimilarity, $d_\sigma(\mathbf{x}, \mathbf{y})$, and the actual dissimilarity, $\hat{d}_\sigma(\mathbf{x}, \mathbf{y})$, defined in Equation 3. The objective function per sample is defined as:

$$\begin{aligned} J_{CE} = & -\hat{d}_\sigma(\mathbf{x}, \mathbf{y}) \log d_\sigma(\mathbf{x}, \mathbf{y}) \\ & - (1 - \hat{d}_\sigma(\mathbf{x}, \mathbf{y})) \log(1 - d_\sigma(\mathbf{x}, \mathbf{y})). \end{aligned} \tag{6}$$

The gradients of the parameters with respect to this objective function are computed and the parameters are updated with the Adam optimization method [48].

Due to the unequal representation of classes in the training set, a naive random sampling strategy of training pairs heavily biases the network to model highly populated classes well at the expense of less populated ones. To deal with this imbalance, we use a class sampling strategy to select the training frames. The procedure can be summarized thus:

1) Sample two distinct classes from the set of HMM states with one-hot representations, $\mathbf{o}_1$ and $\mathbf{o}_2$.
2) Sample two feature vectors, $\mathbf{b}_1$ and $\mathbf{b}_2$ belonging to the classes of $\mathbf{o}_1$ and $\mathbf{o}_2$ respectively.
3) Train the network with sample pairs: $(\mathbf{b}_1, \mathbf{o}_1)$, $(\mathbf{b}_1, \mathbf{o}_2)$, $(\mathbf{b}_2, \mathbf{o}_1)$ and $(\mathbf{b}_2, \mathbf{o}_2)$ and their corresponding labels.

By repeating this procedure, the network is trained with samples of approximately equal class representation. The EDML network[3] is implemented in Pytorch [49].

2) EDML-based keyword search: The EDML network provides a way to optimize the query and document representations as well as the distortion function for the dynamic time warping. Prior to the actual search, the activations of the neural network from the layers directly preceding the final dot-product are stored along with the final bias, c. For the document feature vector sequence, $\mathcal{B} := (\mathbf{b}^1, \mathbf{b}^2, \ldots, \mathbf{b}^M)$, we compute and store a sequence, $\mathcal{R} := (\mathbf{r}^1, \mathbf{r}^2, \ldots, \mathbf{r}^N)$ where:

$$\mathbf{r}^i = \mathbf{W}\mathcal{F}(\mathbf{b}^i). \tag{7}$$

This procedure is somewhat analogous to index generation in LVCSR-based search methodologies. Furthermore, for the set of phone states, with one-hot representations, $\mathcal{O} = \{\mathbf{o}_j\}$, we compute and store another set of representations, $\mathcal{S} = \{\mathbf{s}_j\}$ such that:

$$\mathbf{s}_j = \mathbf{W}(\mathbf{V}\mathbf{o}_j). \tag{8}$$

Queries are thus represented as a concatenation of the pertinent state vectors from $\mathcal{S}$ and searched within the document, $\mathcal{R}$, with sDTW using the distance function:

$$d_\Sigma(\mathbf{r}^i, \mathbf{s}^j) = \sigma(\langle \mathbf{r}^i, \mathbf{s}^j \rangle + c). \tag{9}$$

IV. Experimental Results and Discussion

In this section, we describe the experiments performed to demonstrate the effectiveness of the method described in the previous section. First, we describe the dataset on which the experiments are run and the metrics used to measure system performance. We experiment on four languages. For each language, we run a pair of experiments to observe the performance of the models at different levels of data scarcity and compare the results to a competitive baseline trained with the same data.

A. Dataset

The experiments are conducted on the limited language pack (LLP) data from the IARPA Babel Program [37] which contains ten hours of transcribed conversational telephone speech for each language. We divide the languages available to us into two groups: our target languages, on which we evaluate our models, are Pashto,[4], Turkish,[5] Zulu[6] and Kazakh.[7] The other 19 languages are used as the source languages whose LLP training data (10 hours each, 190 hours total) are pooled together to train the multilingual bottleneck feature extractor. This separation of languages is done to simulate a scenario in which the target language's training data is not available at the time of the multilingual training, and the multilingual network is just a black box that has no knowledge of the target language.

For each language, we train an HMM-GMM with speaker adaptive training (SAT) and use it to to get frame level forced alignments. In the source languages, these alignments are used as labels for the multilingual TDNN training. In the target languages, the alignments are used for TDNN finetuning and EDML training. We run two sets of experiments for each of the target languages:

1) LR-10: In this setting, we use the entire 10-hour LLP training data for each target language.
2) LR-1: In this setting, we use only a one-hour subset of the target language training data to train our models.

[3] *https://github.com/ysfb/crosslingual_exemplars*

[4] babel104-v0.4bY (dev:kwlist3, evalpart1:kwlist4).
[5] babel105b-v0.4 (dev:kwlist, evalpart1:kwlist2).
[6] babel206b-v0.1e (dev:kwlist3, evalpart1:kwlist4).
[7] babel302b-v1.0a (dev:kwlist, evalpart1:kwlist4).

TABLE I
QUERY DISTRIBUTION PER LANGUAGE

Language	Pashto		Turkish		Zulu		Kazakh	
Document	Dev	Eval	Dev	Eval	Dev	Eval	Dev	Eval
#Queries	2065	4203	307	3171	2000	3310	4171	4533
LR-10 OOV (%)	29	23	29	38	40	34	26	35
LR-1 OOV (%)	65	62	68	81	84	76	62	80

Note that the multilingual network is the same as in the LR-10 setting (trained with 10 hours from each of the source languages).

In each setting, we use the corresponding lexicon subset to train the G2P system, i.e. full 10-hour lexicon in the LR-10 setting, and one-hour lexicon subset in the LR-1 setting.

In addition to the training data, each language has a pair of spoken documents on which the search is conducted. The development (dev) archive has ten hours of speech with the necessary labels to evaluate system performance. It is on this document that we tune the hyperparameters of our models such as the pruning threshold, normalization percentile and the score threshold. The evaluation (eval) archive is the five-hour document on which we purely test the models we have trained and tuned on the development datasets. Although the length of the evaluation document provided in the Babel program is about fifteen hours for each language, the reference information required for performance evaluation is openly available for only a five-hour subset (referred to as `evalpart1`) of each. Table I shows the keyword distribution for each language in terms of total number of keyword and OOV rates for both settings. Note that in the LR-1 setting, the OOV rate is much higher than in the LR-10 setting, since the pronunciation lexicon, like the rest of the training data, is shrunken in the LR-1 setting.

B. Evaluation Metrics

A good evaluation metric simulates how good an end user would consider the results of a model to be. In this work, we use the term weighted value to measure system performance. The TWV gives a measure of recall and precision at a pre-specified global threshold. Given a set of terms, $\mathcal{Q}$ and a threshold, θ, the TWV is defined thus:

$$TWV(\theta, \mathcal{Q}) = 1 - \frac{1}{|\mathcal{Q}|} \sum_{q \epsilon \mathcal{Q}} (P_{miss}(q, \theta) + \beta P_{fa}(q, \theta)) \quad (10)$$

where $P_{miss}(q, \theta)$ and $P_{fa}(q, \theta))$ are the probabilities of misses and false alarms respectively, and β is a parameter that controls the relative costs of false alarms and misses. Following the NIST STD evaluations [50], we set $\beta = 999.9$. On the dev-set, we report the maximum term weighted value (MTWV) which is the TWV at the threshold that maximizes it. This threshold is then used to compute the actual term weighted value (ATWV) for the eval set.

C. System Setup

1) Baseline Systems: Our baseline model uses the contemporary LVCSR-based KWS approach. To have a fair comparison, we also use the multilingual information to bootstrap the baseline acoustic model, which results in considerably better performance than using the monolingual LVCSR system (see Table II). We truncate the multilingual TDNN at the bottleneck layer (the last layer that is shared across the source languages), and for each of the target languages, we append a ReLU layer followed by a softmax output layer and finetune the entire network on that language's training data while lowering the learning rate (by a factor of 10) for the transferred layers.

TABLE II
MTWV COMPARISON OF THE BASELINE SYSTEM WITH OR WITHOUT MULTILINGUAL PRE-TRAINING ON THE DEV-SET. IN LR-1*, THE SYSTEM IS TRAINED WITH 1-HOUR ACOUSTIC DATA AND A 10-HOUR LEXICON

Language	Corpus	Monolingual	Multilingual
Pashto	LR-1	0.0236	0.0709
	LR-1*	0.0416	0.1229
	LR-10	0.2019	0.2754
Turkish	LR-1	0.0692	0.1514
	LR-1*	0.1061	0.2241
	LR-10	0.4059	0.4892
Zulu	LR-1	0.0403	0.1194
	LR-1*	0.0630	0.1846
	LR-10	0.2948	0.3716
Kazakh	LR-1	0.0376	0.1107
	LR-1*	0.0465	0.1453
	LR-10	0.2956	0.3543
Average	LR-1	0.0427	0.1131
	LR-1*	0.0643	0.1692
	LR-10	0.2996	0.3726

An index constructed from the LVCSR lattices is used to search for the queries. The IV terms are searched in a word index. For the OOV words, we use a subword index, which we found to be better than using the proxy keyword method [9], [10]. In the Kaldi recipe we used, syllable lattices are obtained and then converted to phonetic ones as in the best model from [13]. The OOV phone sequence is then searched on the resulting phone index. To facilitate setting a global score threshold across keywords, the KST method of [51] is used for score normalization.

In the LR-1 setting, we consider a second, upper-bound, baseline that uses the full ten-hour lexicon instead of just the one-hour subset. This simulates a method like lexicon expansion [8] which works on the premise that it is easier to obtain the pronunciation of words in a language than it is to get more transcribed speech data. It should be noted that this is an ideal expansion since the lexicon expansion methods generally use automatically generated pronunciations and not the manual lexicon that we use.

Table II shows the impact of multilingual pretraining on the baseline system. Across all languages and settings, the use of multilingual data significantly improves the baseline system, and thus provides a stronger baseline to compare with. This result has already been shown in other works and is provided here only for completeness.

Table III provides a comparison of subword decoding with proxy keywords for handling OOV terms. We found the subword decoding approach to be consistently better for LR-1 and LR-10 and so we use it as our baseline. Interestingly, the performances of the two approaches are comparable when we use the augmented lexicon (*LR-1**).

2) EDML-based KWS Systems: We train the proposed EDML models with two feature sets and use them for search:

TABLE III
MTWV COMPARISON OF BASELINE OOV HANDLING METHODS ON THE DEV-SET

Language	Corpus	Proxy	Subword
Pashto	LR-1	0.0083	0.0305
	LR-1*	0.1006	0.0993
	LR-10	0.0931	0.1453
Turkish	LR-1	0.0171	0.0730
	LR-1*	0.1799	0.1617
	LR-10	0.1865	0.2553
Zulu	LR-1	0.0096	0.1058
	LR-1*	0.1360	0.1802
	LR-10	0.1335	0.3354
Kazakh	LR-1	0.0110	0.0649
	LR-1*	0.0937	0.1121
	LR-10	0.1229	0.1838
Average	LR-1	0.0115	0.0686
	LR-1*	0.1276	0.1383
	LR-10	0.1340	0.2300

1) EDML_BNF: In this system, the bottleneck features are used directly for the EDML system development.
2) EDML_PPG: In this system, the baseline system is used to generate phone posteriorgrams for the EDML.

In both cases, the EDML networks comprise:

- The document representation learner, $\mathcal{F}$, has 3 ReLU layers with 1024-1024-200 units and a dropout rate of 0.5 on each layer.
- The query model, $\mathbf{V}$, has an output dimension of 200.
- The distance metric learner has $\mathbf{W} \in \mathbb{R}^{200} \times 200$.

3) Score Normalization and Fusion: Since the TWV is measured at a single global threshold across keywords, and the keywords have scores with different distributions, it is necessary to first normalize the scores to bring the scores of different keywords into comparable ranges. We modify the b-norm of [52] to normalize with the 90-th percentile score instead of the median. This percentile, found by tuning on the Turkish dev set, is used for all the other experiments.

To fully utilize the complementary nature of the proposed system to the baseline, we also perform a fusion of scores between the proposed system and the baseline for each language under both training conditions. First, each system's scores are re-calibrated to minimize the normalized cross-entropy [53] and then overlapping hypotheses are merged by simply summing up their scores while singletons keep their original scores. The calibration is done with an affine transform whose parameters are learned on the dev set and then used on the eval set.

D. Impact of Document Representation Learner

The main difference between the EDML network used in this work and the JDML previously proposed in [14] is the introduction of a sequence of nonlinear layers that transform the document representation so that the query model and distance metric learner can be more easily trained. We introduced this subnetwork to cope with the highly variant nature of the *bottleneck features*, and conduct a series of experiments to quantify the importance of these nonlinearities with different kinds of features. The results are shown in Tables IV and V.

In both settings, we see that although EDML gives slight but consistent improvements over the JDML when the input features are posteriorgrams, this improvement is considerably higher when bottleneck features are used. This is particularly significant in the LR-1 setting (Table IV) where the bottleneck features, which perform worse than phone posteriorgrams with JDML (-0.0712 average ATWV difference on eval set), perform better with EDML ($+0.0177$), resulting in the best overall system in the LR-1 setting.

TABLE IV
TWV COMPARISON OF JDML AND EDML IN THE LR-1 SETTING. THE DEV SET SCORES ARE MTWV WHILE THE EVAL SET SCORES ARE ATWV

Language	Feature	Dev		Eval	
		JDML	EDML	JDML	EDML
Pashto	BNF	0.0634	**0.0990**	0.0626	**0.1036**
	PPG	0.0917	0.0892	0.1026	0.0968
Turkish	BNF	0.1547	**0.2820**	0.0810	0.1598
	PPG	0.2131	0.2532	0.1412	**0.1614**
Zulu	BNF	0.1167	**0.2119**	0.1239	**0.2229**
	PPG	0.1693	0.1820	0.1644	0.1911
Kazakh	BNF	0.0561	**0.1350**	0.0168	**0.0829**
	PPG	0.1055	0.0977	0.0559	0.0490
Average	BNF	0.0977	**0.1820**	0.0711	**0.1423**
	PPG	0.1449	0.1555	0.1160	0.1246

TABLE V
PERFORMANCE COMPARISON OF JDML AND EDML IN THE LR-10 SETTING. RESULTS OBTAINED USING MONOLINGUAL POSTERIORGRAMS [14] ARE ALSO INCLUDED

Language	Feature	Dev		Eval	
		JDML	EDML	JDML	EDML
Pashto	Mono-PPG	0.1101	**0.1208**	0.1148	**0.1214**
	PPG	0.1494	**0.1740**	0.1638	**0.1832**
	BNF	0.0818	**0.1507**	0.0873	**0.1607**
Turkish	Mono-PPG	0.2617	**0.2783**	0.1586	**0.1815**
	PPG	0.3424	**0.3991**	0.2433	**0.2795**
	BNF	0.1746	**0.3557**	0.0799	**0.2476**
Zulu	Mono-PPG	0.1816	**0.2057**	0.1648	**0.1974**
	PPG	0.2527	**0.3064**	0.2202	**0.2834**
	BNF	0.1464	**0.2882**	0.1319	**0.2630**
Kazakh	Mono-PPG	0.1146	**0.1259**	0.0566	**0.0658**
	PPG	0.2112	**0.2399**	0.1378	**0.1647**
	BNF	0.0987	**0.2024**	0.0313	**0.1164**
Average	Mono-PPG	0.1670	**0.1827**	0.1237	**0.1415**
	PPG	0.2389	**0.2799**	0.1913	**0.2277**
	BNF	0.1254	**0.2493**	0.0826	**0.1969**

In the LR-10 setting (Table V), we also perform a direct comparison with the previous work in [14]. Compared to that work (the "Mono-PPG" rows in the table), we note that the use of EDML improves the TWV. Furthermore, when the multilingual features are used, even better TWVs are obtained. It is worth noting that the BNF performs worse than even the monolingual posteriorgrams when JDML is used, but is only outperformed by the multilingual posteriorgrams with EDML.

In fact, the average MTWV obtained from the monolingual posteriorgrams with ten hours of training data is almost identical to that obtained using one-hour's worth of supervision with the multilingual bottleneck features (compare the corresponding rows of Tables IV and V).

E. Comparison and Fusion with LVCSR Based Search

We directly compare the results of our best EDML based systems with the (multilingual) LVCSR based systems in this section in each setting. From the previous section, we note that in the

TABLE VI
LR-1: TWV OF EDML_BNF WITH LVCSR BASED SEARCH. L IS THE LVCSR BASELINE AND L* IS AUGMENTED WITH A 10H LEXICON

Language	System	Dev (MTWV)			Eval (ATWV)		
		IV	OOV	All	IV	OOV	All
Pashto	L	0.1458	0.0305	0.0709	0.1786	-0.0002	0.0677
	L*	0.1594	0.0993	0.1229	0.1839	0.1004	0.1402
	EDML	0.1070	0.0973	0.0990	0.1074	0.1002	0.1036
	EDML+L*	**0.1889**	**0.1704**	**0.1756**	**0.2003**	**0.1526**	**0.1754**
Turkish	L	0.3182	0.0730	0.1515	0.1652	0.0682	0.0866
	L*	0.3548	0.1617	0.2241	0.1844	0.1607	0.1667
	EDML	0.3263	0.2662	0.2820	0.1508	0.1629	0.1598
	EDML+L*	**0.4171**	**0.3258**	**0.3499**	**0.2257**	**0.2297**	**0.2287**
Zulu	L	0.1905	0.1058	0.1194	0.2545	0.0797	0.1216
	L*	0.2072	0.1802	0.1846	0.2390	0.1294	0.1613
	EDML	0.1732	0.2196	0.2119	0.2547	0.2098	0.2229
	EDML+L*	**0.2463**	**0.2818**	**0.2751**	**0.3133**	**0.2398**	**0.2612**
Kazakh	L	0.1854	0.0649	0.1031	0.1083	0.0380	0.0521
	L*	0.1990	0.1121	0.1453	0.1244	0.0827	0.0944
	EDML	0.1277	0.1411	0.1350	0.0780	0.0848	0.0829
	EDML+L*	**0.2189**	**0.1883**	**0.1985**	**0.1481**	**0.1299**	**0.1350**
Average	L	0.2100	0.0685	0.1112	0.1767	0.0464	0.0820
	L*	0.2301	0.1383	0.1692	0.1829	0.1183	0.1406
	EDML	0.1835	0.1810	0.1820	0.1477	0.1394	0.1423
	EDML+L*	**0.2678**	**0.2416**	**0.2498**	**0.2219**	**0.1880**	**0.2001**

TABLE VII
LR-10: TWV OF EDML_PPG WITH LVCSR BASED SEARCH. L IS THE LVCSR BASELINE

Language	System	Dev (MTWV)			Eval (ATWV)		
		IV	OOV	All	IV	OOV	All
Pashto	L	0.3186	0.1453	0.2754	**0.3482**	0.1362	0.3190
	EDML	0.1612	0.2168	0.1740	0.1781	0.2156	0.1832
	EDML+L	**0.3232**	**0.2430**	**0.3031**	0.3473	**0.2406**	**0.3326**
Turkish	L	0.5814	0.2553	0.4892	0.4538	0.2787	0.4051
	EDML	0.4321	0.3240	0.3991	0.2648	0.3177	0.2795
	EDML+L	**0.5871**	**0.3519**	**0.5162**	**0.4726**	**0.3767**	**0.4459**
Zulu	L	0.3957	0.3354	0.3716	0.3705	0.2504	0.3381
	EDML	0.2680	0.3682	0.3064	0.2634	0.3376	0.2834
	EDML+L	**0.4175**	**0.4086**	**0.4118**	**0.3909**	**0.3531**	**0.3807**
Kazakh	L	**0.4150**	0.1838	0.3543	0.3370	0.1803	0.2993
	EDML	0.2245	0.2877	0.2399	0.1615	0.1749	0.1647
	EDML+L	0.4096	**0.3271**	**0.3867**	**0.3404**	**0.2320**	**0.3143**
Average	L	0.4277	0.2299	0.3726	0.3774	0.2114	0.3404
	EDML	0.2714	0.2992	0.2798	0.2169	0.2615	0.2277
	EDML+L	**0.4343**	**0.3327**	**0.4044**	**0.3878**	**0.3006**	**0.3684**

LR-1 setting EDML_BNF performs better than EDML_PPG, with the reverse holding true in the LR-10 setting (see Tables IV and V). The LVCSR systems use word lattices for IV and subword lattices for OOV keyword search. Overall, for OOV terms, the EDML systems improve the TWV by reducing misses while slightly increasing false alarms.

1) LR-1 setting: Table VI shows the results in the LR-1 setting, where we train our models using only one hour of transcribed speech from the target languages. Compared with the LVCSR baseline, although the EDML-based systems perform worse on IV terms, they are considerably better on OOV ones, even when the baseline is modified to use the ten-hour lexicon.

Overall, the EDML system with bottleneck features improves upon the augmented baseline (L*) by an average MTWV of 0.0128 (0.0708 compared to the actual baseline) on the dev-set. On the eval set, the corresponding ATWV improvements are 0.0603 and 0.0017 over the baseline and augmented baseline respectively. Note that for the EDML systems, the IV and OOV TWVs are almost identical.

When the augmented baseline is combined with the EDML_BNF system, it is improved by 0.0806 MTWV on the dev-set and 0.0595 ATWV on the eval-set. When just OOV terms are considered, the improvements after fusion are slightly more pronounced at 0.1083 on dev MTWV and 0.0697 on eval ATWV. This is despite the fact that a high number of the OOV terms are actually IV to the baseline with the augmented lexicon (the exact figures can be obtained from Table I). This result is significant because the system is expected to encounter a high volume of OOV terms in the long run, given the open vocabulary nature of the search problem.

2) LR-10 setting: Table VII shows the results in the LR-10 setting, where we use the entire 10-hour target language training data to train the systems. Similar to the behavior observed in the LR-1 setting, the baseline performs better than the EDML systems on IV terms (with a wider margin in this setting). Meanwhile, on OOV terms, using the EDML_PPG provides an average MTWV gain of 0.0693 on the dev set and an average eval set ATWV improvement of 0.0501 over the baseline.

On fusion of the two systems, the OOV gains on the dev and eval set increase to 0.1026 and 0.0892 respectively. Even the IV performance of the baseline system is improved upon fusion despite it already having a better performance than the EDML system.

F. Comparison of BNF with PPG

In our EDML experiments, we have used two sets of features, namely multilingual posteriorgrams and bottleneck features. Figures 5 and 6 show the comparative TWV performance of both features as the amount of training data is decreased. Each bar represents $100 \times \frac{TWV(BNF)-TWV(PPG)}{TWV(PPG)}\%$ for each training corpus size for each language. To obtain these results, we ran an additional set of experiments with 3 hours of training data per language.

At 10 hours, using BNFs results in worse performance than using posteriorgrams. As the training data is decreased to three hours, the two features give almost equal term weighted values on both the dev and eval sets; and in the setting with only one

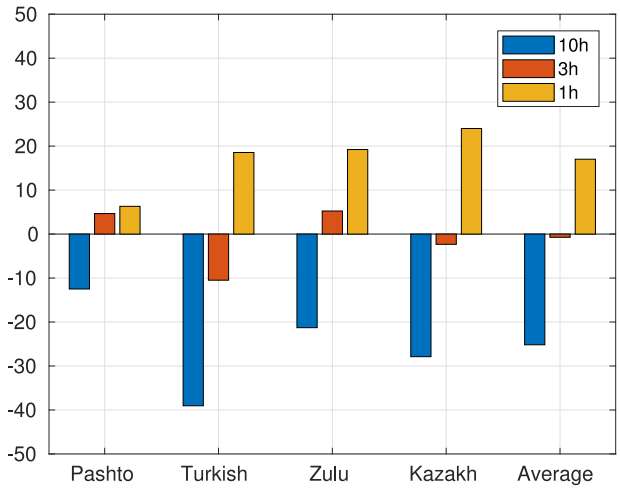

Fig. 5. Dev set MTWV progression. Each bar represents the percentage differential MTWV between EDML_BNF and EDML_PPG.

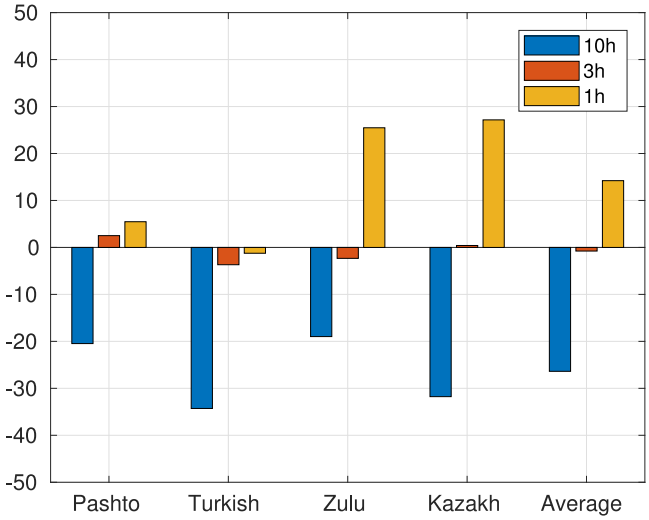

Fig. 6. Eval set ATWV progression. Each bar represents the percentage differential ATWV between EDML_BNF and EDML_PPG.

hour of training data, the BNF systems outperform their posteriorgram counterparts. Overall, compared with posteriorgrams, the BNFs perform better as the training data size is decreased.

This trend can be explained by the way the features are generated: posteriorgrams are *target language dependent* while bottleneck features are not. Since the search *is* language dependent, given enough data, the posteriorgrams are expected to perform better. However, with little training data, the quality of the posteriorgram representation is hampered by the lack of enough training samples to generalize well enough for the target language. Therefore, in very low resource settings (below 3h from our experiments), simply using the language independent bottleneck representation is the better alternative.

V. Conclusion

In this work, we tackle some of the issues associated with keyword search in data-scarce settings, namely the paucity of the acoustic and linguistic data required for building an LVCSR-based keyword search system. We mitigate the effects of acoustic data scarcity by bootstrapping with data from other languages. To reduce the the impact of linguistic data scarcity, we propose a model based on dynamic time warping template matching that is less dependent on linguistic data (language model and lexicon) and so is less sensitive to the scarcity of such. Where the LVCSR system requires extra processing to handle out-of-vocabulary keywords, the proposed model does not differentiate between IV and OOV terms.

In a scenario with only one hour of training data per target language, we find that the proposed EDML system works better with the language independent bottleneck features. Furthermore, the EDML system performs better than the LVCSR system. This performance difference is especially pronounced when only OOV terms are considered, even when the baseline is augmented with a large, unrealistically good lexicon.

In a milder setting with ten hours of target language training data, we find that the EDML system performs better when the data representation integrates target language knowledge in the form of finetuned posterior features. Although the EDML system still outperforms the LVCSR baseline on OOV terms, the impact of this difference on the overall system is less pronounced due to the reduced OOV rate. Being based on strictly frame-level acoustic matching makes the proposed system insensitive to whether or not a term contains an OOV word; this robustness comes at the expense of being able take advantage of some of the side-information that allows the LVCSR-based baseline to perform better on the retrieval of IV terms. Another baseline that we compare with is the EDML trained with strictly monolingual posterior features; in this, we see that the utilization of the multilingual features results in vastly improved keyword search performance.

There are two main weaknesses of our approach compared to the LVCSR based search in a real use case: the relatively worse performance on IV term retrieval and the computational cost of dynamic time warping. Future work will include improving the IV term performance e.g., by using actual examples from the training collection, an approach that was recently used for rescoring LVCSR keyword search results in [54]. Another line of work will be focused on using faster, more memory efficient approximations to speed up the dynamic time warping or building models that bypass it entirely.

Acknowledgment

This study makes use of IARPA Babel Program data.

References

[1] C. Chelba, T. J. Hazen, and M. Saraclar, "Retrieval and browsing of spoken content," *IEEE Signal Process. Mag.*, vol. 25, no. 3, pp. 39–49, May 2008.
[2] L. Mangu, E. Brill, and A. Stolcke, "Finding consensus in speech recognition: Word error minimization and other applications of confusion networks," *Comput. Speech Lang.*, vol. 14, no. 4, pp. 373–400, 2000.
[3] K. Ng and V. W. Zue, "Subword-based approaches for spoken document retrieval," *Speech Commun.*, vol. 32, no. 3, pp. 157–186, 2000.
[4] B. Logan, J.-M. Van Thong, and P. J. Moreno, "Approaches to reduce the effects of OOV queries on indexed spoken audio," *IEEE Trans. Multimedia*, vol. 7, no. 5, pp. 899–906, Oct. 2005.

[5] M. Saraçlar and R. Sproat, "Lattice-based search for spoken utterance retrieval," in *Proc. Human Lang. Technol. Conf. North Am. Chapter Assoc. Comput. Linguistics*, vol. 51, 2004, pp. 129–136.
[6] J. Mamou, B. Ramabhadran, and O. Siohan, "Vocabulary independent spoken term detection," in *Proc. 30th Annu. Int. ACM SIGIR Conf. Res. Develop. Inf. Retrieval*, 2007, pp. 615–622.
[7] I. Szöke, M. Fapšo, and L. Burget, "Hybrid word-subword decoding for spoken term detection," in *Proc. 31st Annu. Int. ACM SIGIR Conf. Res. Develop. Inf. Retrieval*, 2008, pp. 42–48.
[8] A. Gandhe, L. Qin, F. Metze, A. Rudnicky, I. Lane, and M. Eck, "Using web text to improve keyword spotting in speech," in *Proc. IEEE Workshop Autom. Speech Recog. Understanding*, 2013, pp. 428–433.
[9] G. Chen, O. Yilmaz, J. Trmal, D. Povey, and S. Khudanpur, "Using proxies for OOV keywords in the keyword search task," in *Proc. IEEE Workshop Autom. Speech Recog. Understanding*, Dec. 2013, pp. 416–421.
[10] M. Saraclar *et al.*, "An empirical study of confusion modeling in keyword search for low resource languages," in *Proc. IEEE Workshop Autom. Speech Recog. Understanding*, 2013, pp. 464–469.
[11] H. Su *et al.*, "Improvements on transducing syllable lattice to word lattice for keyword search," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process.*, 2015, pp. 4729–4733.
[12] D. Karakos, I. Bulyko, R. Schwartz, S. Tsakalidis, L. Nguyen, and J. Makhoul, "Normalization of phonetic keyword search scores," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process.*, 2014, pp. 7834–7838.
[13] D. Karakos and R. Schwartz, "Subword and phonetic search for detecting out-of-vocabulary keywords," in *Proc. Annu. Conf. Int. Speech Commun. Assoc.*, 2014, pp. 2469–2473.
[14] B. Gündoğdu, B. Yusuf, and M. Saraçlar, "Joint learning of distance metric and query model for posteriorgram-based keyword search," *IEEE J. Sel. Topics Signal Process.*, vol. 11, no. 8, pp. 1318–1328, Dec. 2017.
[15] T. Sercu *et al.*, "Network architectures for multilingual speech representation learning," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process.*, 2017, pp. 5295–5299.
[16] Y. Zhang and J. R. Glass, "Unsupervised spoken keyword spotting via segmental DTW on Gaussian posteriorgrams," in *Proc. IEEE Workshop Autom. Speech Recog. Understanding*, 2009, pp. 398–403.
[17] A. S. Park and J. R. Glass, "Unsupervised pattern discovery in speech," *IEEE Trans. Audio, Speech, Lang. Process.*, vol. 16, no. 1, pp. 186–197, Jan. 2008.
[18] T. J. Hazen, W. Shen, and C. White, "Query-by-example spoken term detection using phonetic posteriorgram templates," in *Proc. IEEE Workshop Autom. Speech Recog. Understanding*, 2009, pp. 421–426.
[19] C. Parada, A. Sethy, and B. Ramabhadran, "Query-by-example spoken term detection for OOV terms," in *Proc. IEEE Workshop Autom. Speech Recog. Understanding*, Nov. 2009, pp. 404–409.
[20] X. Anguera, "Speaker independent discriminant feature extraction for acoustic pattern-matching," in *Proc. IEEE Int. Conf. Acoust., Speech Signal Process.*, 2012, pp. 485–488.
[21] L. J. Rodriguez-Fuentes and M. Penagarikano, "MediaEval 2013 spoken web search task: System performance measures," Tech. Rep. TR-2013-1, Dept. Electr. Electron., Univ. Basque Country, Leioa, Spain, 2013.
[22] L. J. Rodriguez-Fuentes, A. Varona, M. Penagarikano, G. Bordel, and M. Diez, "High-performance query-by-example spoken term detection on the SWS 2013 evaluation," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process.*, May 2014, pp. 7819–7823.
[23] I. Szoke, L. Burget, F. Grezl, J. H. Cernocky, and L. Ondel, "Calibration and fusion of query-by-example systems–BUT SWS 2013," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process.*, 2014, pp. 7849–7853.
[24] J. Proenga, A. Veiga, and F. Perdigão, "Query by example search with segmented dynamic time warping for non-exact spoken queries," in *Proc. 23rd Eur. Signal Process. Conf.*, 2015, pp. 1661–1665.
[25] L. Sarı, B. Gündoğdu, and M. Saraçlar, "Fusion of LVCSR and posteriorgram based keyword search," in *Proc. Annu. Conf. Int. Speech Commun. Assoc.*, 2015, pp. 824–828.
[26] B. Gündoğdu, L. Sarı, G. Çetinkaya, and M. Saraçlar, "Template-based keyword search with pseudo posteriorgrams," in *Proc. 24th Signal Process. Commun. Appl. Conf.*, 2016, pp. 973–976.
[27] B. Gündoğdu and M. Saraclar, "Distance metric learning for posteriorgram based keyword search," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process.*, 2017, pp. 5660–5664.
[28] B. Gündoğdu and M. Saraçlar, "Similarity learning based query modeling for keyword search," in *Proc. Annu. Conf. Int. Speech Commun. Assoc.*, 2017, pp. 3617–3621.
[29] S. Thomas, S. Ganapathy, and H. Hermansky, "Cross-lingual and multi-stream posterior features for low resource LVCSR systems," in *Proc. Annu. Conf. Int. Speech Commun. Assoc.*, 2010, pp. 877–880.
[30] N. T. Vu, D. Imseng, D. Povey, P. Motlicek, T. Schultz, and H. Bourlard, "Multilingual deep neural network based acoustic modeling for rapid language adaptation," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process.*, 2014, pp. 7639–7643.
[31] N. T. Vu, W. Breiter, F. Metze, and T. Schultz, "An investigation on initialization schemes for multilayer perceptron training using multilingual data and their effect on ASR performance," in *Proc. Annu. Conf. Int. Speech Commun. Assoc.*, 2012, pp. 2586–2589.
[32] K. M. Knill, M. J. F. Gales, S. P. Rath, P. C. Woodland, C. Zhang, and S. Zhang, "Investigation of multilingual deep neural networks for spoken term detection," in *Proc. IEEE Workshop Autom. Speech Recog. Understanding*, Dec. 2013, pp. 138–143.
[33] A. Ghoshal, P. Swietojanski, and S. Renals, "Multilingual training of deep neural networks," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process.*, 2013, pp. 7319–7323.
[34] S. Scanzio, P. Laface, L. Fissore, R. Gemello, and F. Mana, "On the use of a multilingual neural network front-end," in *Proc. Annu. Conf. Int. Speech Commun. Assoc.*, 2008, pp. 2711–2714.
[35] K. Veselý, M. Karafiát, F. Grézl, M. Janda, and E. Egorova, "The language-independent bottleneck features," in *Proc. IEEE Workshop Spoken Lang. Technol.*, 2012, pp. 336–341.
[36] G. Heigold *et al.*, "Multilingual acoustic models using distributed deep neural networks," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process.*, 2013, pp. 8619–8623.
[37] M. Harper, *IARPA Babel Program*, 2014, accessed on: Jun. 2018. [Online]. Available: https://www.iarpa.gov/index.php/research-programs/babel
[38] Z. Tüske, P. Golik, D. Nolden, R. Schlüter, and H. Ney, "Data augmentation, feature combination, and multilingual neural networks to improve ASR and KWS performance for low-resource languages," in *Proc. Annu. Conf. Int. Speech Commun. Assoc.*, 2014, pp. 1420–1424.
[39] M. Karafíat *et al.*, "2016 BUT Babel system: Multilingual BLSTM acoustic model with i-vector based adaptation," in *Proc. Annu. Conf. Int. Speech Commun. Assoc.*, 2017, pp. 719–723.
[40] J. Cui *et al.*, "Multilingual representations for low resource speech recognition and keyword search," in *Proc. IEEE Workshop Autom. Speech Recog. Understanding*, 2015, pp. 259–266.
[41] E. Chuangsuwanich, Y. Zhang, and J. Glass, "Multilingual data selection for training stacked bottleneck features," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process.*, 2016, pp. 5410–5414.
[42] S. Thomas, K. Audhkhasi, J. Cui, B. Kingsbury, and B. Ramabhadran, "Multilingual data selection for low resource speech recognition," in *Proc. Annu. Conf. Int. Speech Commun. Assoc.*, 2016, pp. 3853–3857.
[43] V. Peddinti, D. Povey, and S. Khudanpur, "A time delay neural network architecture for efficient modeling of long temporal contexts," in *Proc. Annu. Conf. Int. Speech Commun. Assoc.*, 2015, pp. 3214–3218.
[44] D. Povey *et al.*, "The Kaldi speech recognition toolkit," in *Proc. IEEE Workshop Autom. Speech Recog. Understanding*, 2011, pp. 1–4.
[45] M. Müller, *Information Retrieval for Music and Motion*. Berlin, Germany: Springer-Verlag, 2007.
[46] M. Bisani and H. Ney, "Joint-sequence models for grapheme-to-phoneme conversion," *Speech Commun.*, vol. 50, no. 5, pp. 434–451, 2008.
[47] B. Yusuf, B. Gundogdu, and M. Saraclar, "Beyond posteriorgram: Bottleneck features for keyword search," in *Proc. 26th Signal Process. Commun. Appl. Conf.*, 2018, pp. 1–4.
[48] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," 2014, arXiv preprint arXiv:1412.6980.
[49] A. Paszke *et al.*, "Automatic differentiation in PyTorch," in *Proc. NIPS-Workshop*, 2017, pp. 1–4.
[50] J. G. Fiscus, J. Ajot, J. S. Garofolo, and G. Doddington, "Results of the 2006 spoken term detection evaluation," in *Proc. ACM SIGIR Workshop Searching Spontaneous Conversational Speech*, 2007, pp. 51–57.
[51] D. R. Miller *et al.*, "Rapid and accurate spoken term detection," in *Proc. Annu. Conf. Int. Speech Commun. Assoc.*, 2007, pp. 314–317.
[52] B. Gündogdu, "Keyword search for low resource languages," Ph.D. dissertation, Bogaziçi Univ., Istanbul, Turkey, 2017.
[53] N. Brümmer and J. Du Preez, "Application-independent evaluation of speaker detection," *Comput. Speech Lang.*, vol. 20, nos. 2/3, pp. 230–275, 2006.
[54] V. T. Pham, H. Xu, X. Xiao, N. F. Chen, E. S. Chng, and H. Li, "Re-ranking spoken term detection with acoustic exemplars of keywords," *Speech Commun.*, vol. 104, pp. 12–23, 2018.

Authors', photographs and biographies not available at the time of publication.