INTERSPEECH 2016
September 8–12, 2016, San Francisco, USA

A novel discriminative score calibration method for keyword search

Zhiqiang Lv, Meng Cai, Wei-Qiang Zhang, Jia Liu

Tsinghua National Laboratory for Information Science and Technology
Department of Electronic Engineering, Tsinghua University, Beijing 100084, China
{lv-zq12,cai-m10}@mails.tsinghua.edu.cn, {wqzhang,liuj}@tsinghua.edu.cn

Abstract

The performance of keyword search systems depends heavily on the quality of confidence scores. In this work, a novel discriminative score calibration method has been proposed. By training an MLP classifier employing the word posterior probability and several novel normalized scores, we can obtain a relative improvement of 4.67% for the actual term-weighted value (ATWV) metric on the OpenKWS15 development test dataset. In addition, a LSTM-CTC based keyword verification method has been proposed to supply extra acoustic information. After the information is added, a further improvement of 7.05% over the baseline can be observed.

Index Terms: keyword search, score calibration, CTC, neural network

1. Introduction

A typical keyword search system is based on the transcriptions generated by the automatic speech recognition (ASR) system. Most ASR systems provide a confidence estimate of word posterior probability [1] for every keyword detection. However, word posterior probability is often overestimated due to the pruning of the recognizer when decoding. In the decoding process, hypotheses which get relatively low likelihoods are removed from the final transcriptions. When computing the posterior probability of a word hypothesis, we usually treat the remaining hypothesis space as the total hypothesis space, which leads to the overestimated probabilities.

To obtain a more accurate confidence estimate for keyword search, classifiers were trained using features extracted from lattices and confusion networks ([2], [3], [4]). Discriminative scores output by classifiers can be used as new confidence scores and promising results have been achieved. Features used often include:

- Posterior probabilities and normalized scores after normalizing methods such as keyword specific threshold (KST) [5], sum-to-one (STO) [6] and probability of false alarm (pFA) [7].
- Lexical features such as the total number for graphemes, vowel graphemes, constant graphemes, phones, vowel phones and consonant phones [4].
- Structure features such as the edge rank, the confusion network bin size, the ranking score, the relative-to-max score and the entropy of the bin in confusion networks ([8], [9]).
- Utterance level features such as the start and end time of the detection relative to the start of the utterance where the keyword is found [4].
- Prosodic features such as the pitch for each detection ([4], [8]).
- Language model level features such as the frequency of the keyword, long contextual language information and n-gram frequencies ([8], [10]).

Except conventional classifiers, conditional random field (CRF) [11] has also been employed in score calibration ([10], [12]). CRF can solve the problem of sequence labeling and often shows better performance than conventional classifiers. Due to the nature of sequence processing, CRF often leads to higher complexity. Soto et al. [8] explored a rank model for rescoring confusion networks, and significant improvement of WER has been observed instead of ATWV.

The performance of discriminative classifiers is mainly up to feature engineering. To do discriminative calibration for keyword search, we often have to extract a lot of information carried by lattices and confusion networks. More complex features such as prosodic and lexical features even need some expert knowledge about the target language. This is rather tedious and sometimes impossible, especially when we have no access to the ASR system or know little about the target language.

In this article, we only extract the word posterior probability and the corresponding KST normalized score [13] from the final detection list. Other information contained in lattices or confusion networks is discarded. After normalizing the two scores using different number of sub-word units in keywords proposed in this work, we can also obtain satisfying improvement over the baseline system through discriminative calibration.

Acoustic similarity has also been taken into consideration for keyword search task and can lead to excellent results ([14], [15], [16]). They have to compute the similarity between two detections and do some re-ranking afterwards. To employ the acoustic information, we propose a LSTM-CTC based keyword verification method and try to build a universe verification system after keyword search. The method is proved to be quite effective to distinguish false alarms from true hits, and can therefore improve the keyword search system further.

In this paper, we briefly introduce the keyword search task and metrics in Section 2. Section 3 is about the LSTM-CTC based keyword verification method. Some basic environment setup is introduced in Section 4. Discriminative calibration results are presented in Section 5. Section 6 is the conclusion.

2. Task and metrics

The task of keyword search defined by NIST for the OpenKWS15 evaluations is to find all the exact matches of given queries in a corpus of unsegmented speech data. A query, which can also be called "keyword", can be a sequence of one or more words. The result of this task is a list of all the detections

This work is supported by National Natural Science Foundation of China under Grant No. 61273268, No. 61370034 and No. 61403224.

http://dx.doi.org/10.21437/Interspeech.2016-606

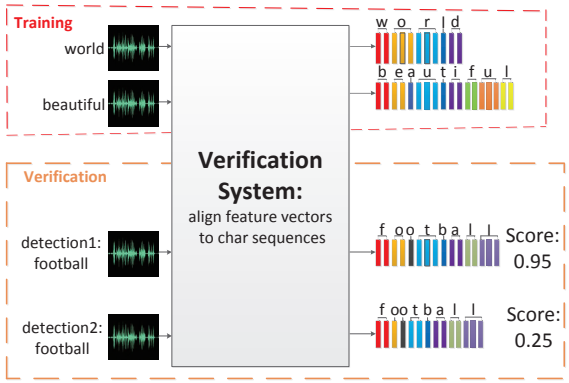

Figure 1: *The keyword verification system.*

of keywords found by keyword search systems. Each detection in the list consists of the keyword id, the utterance in which the detection is found, the start and end time of the detection, and the confidence score.

To evaluate the performance, term-weighted value (TWV) [17] is adopted:

$$TWV(\theta) = 1 - \frac{1}{K}\sum_{w=1}^{K}(\frac{\#miss(w,\theta)}{\#ref(w)} + \beta\frac{\#fa(w,\theta)}{T - \#ref(w)}) \quad (1)$$

where θ is the decision threshold, K is the number of keywords. $\#miss(w,\theta)$ is the number of true tokens of keyword w that are missed at threshold θ, $\#fa(w,\theta)$ is the number of false detections of keyword w at threshold θ, $\#ref(w)$ is the number of reference tokens of w, T is the total amount of the evaluated speech, β is a constant set at 999.9.

As we can see, TWV is a function of the decision threshold θ. A global threshold θ is used to make the hard decision whether a detected keyword is correct. The TWV at the specified global threshold is the actual TWV (ATWV). The optimal threshold results in the maximum term-weighted value (MTWV).

3. LSTM-CTC based keyword verification

After having the detection list, all we want to do is to make sure whether the detection is indeed a correct hit of the keyword or not. Therefore, we try to build a keyword verification system by doing force alignment between the char sequence of keywords and the original speech feature, as is shown in Figure 1.

Connectionist temporal classification (CTC) [18] has been proved to be very suitable for labeling such unsegmented sequence data. Given a input feature sequence X of length T and its char sequence W, CTC is a loss function defined below:

$$L_{CTC}(X,W) = \sum_{C_W} p(C|X) = \sum_{C_W}\prod_{t=1}^{T} p(c_t|X), \quad (2)$$

where C_W is any label sequence of length T corresponds to the correct char sequence W. By summing up over all sets of label locations that yield the same label sequence W, CTC determines a probability distribution over possible labelings, conditioned on the input sequence X.

The long short-term memory (LSTM) neural network [19] has been demonstrated to be quite effective to deal with sequence labeling problems. For our verification system, we put a CTC layer on top of a LSTM neural network trained directly using the original speech feature. For training, we use single word samples form the transcriptions. For evaluation, each input feature sequence is the span of the keyword detection in the utterance, and the label sequence is the corresponding keyword char sequence. The CTC loss of the verification system is regarded as a new score for the detection.

In fact, the CTC loss of the verification system is similar to the acoustic likelihood used in ASR systems. The difference between the two scores lies in:

- The CTC loss is trained on char level and no pronunciation lexicon is needed. The acoustic likelihood in ASR systems is trained on phoneme level or more precisely on state level and usually pronunciation lexicon is necessary. They capture acoustic information of different sub-word unit levels. The training frameworks are also quite different.
- The acoustic likelihood score is only the likelihood of a possible label sequence, while the CTC loss is the likelihood summation of all possible label sequences, which makes the CTC loss more approximate to the acoustic posterior probability. The acoustic likelihood is often a number of wide numeric range and is hard to be used directly as a confidence score.

The verification system proposed in this work is not another ASR system. We can view the verification system as part of an ASR system for it only focuses on verifying whether a segment of speech feature is corresponding to the specified label sequence. The ultimate difference is that the verification system tries to solve the problem of "whether or not" while the ASR system has to deal with the problem "what it is". Besides, we have to process only the segment of the detection instead of the whole utterance for verification and no language modeling is needed. The verification system is easier to build and more flexible.

4. Experiments setup

4.1. The ASR system setup

The baseline keyword search system is the best single system we built for the OpenKWS15 Evaluations with the very limited language pack (VLLP) [20]. The VLLP contains only 3 hours of transcribed training data plus 40 hours of untranscribed training data. Considering the very limited transcribed data resource, we do some data augmentation through adding noise and vocal tract length perturbation (VTLP) [21]. A subspace GMM (SGMM) based acoustic model [22] is trained using the augmented dataset. The language model is trained using the transcriptions of the VLLP training data, together with some selected web data.

The untranscribed data is then decoded using the trained SGMM acoustic model and semi-supervised training is performed. The final ASR system is based on the convolutional maxout neural network (CMNN) acoustic model [23] with 40-dimensional Mel filterbank features and their first- and second-order derivatives. Two convolutional layers with 256 maxout neurons and five fully-connected layers with 1000 maxout neurons are used. Each maxout neurons contains 2 pieces. A max-pooling layer with a pooling size of 3 is used between the convolutional layers. The first convolutional layer has a band width

of 8. The second convolutional layer has a band width of 4. Dropout training is applied to the fully-connected part of the CMNN.

4.2. Discriminative score calibration setup

All the keyword search experiments are conducted on the OpenKWS15 development test dataset, which consists of 10 hours of transcribed data. Detections are divided into 4 classes [24] : "YES, CORR" representing a correct hit, "YES, FA" representing a false alarm, "NO, MISS" representing a true hit but judged to be a false alarm, "NO, CORR!DET" representing a correctly judged false alarm.

The keyword list for training the MLP classifier consists of 1657 keywords and 97215 detections. There is a number of detections with the label "NO, CORR!DET". To avoid the imbalance of training data, only 18884 training samples are selected, of which there are 4025 "YES, CORR", 703 "NO, MISS", 4631 "YES, FA" and 9525 "NO, CORR!DET". The MLP classifier contains two hidden layers, of which each has 256 hidden neurons. The keyword list for evaluation consists of 791 keywords, having 137762 detections.

For each detection list of both the training and evaluation keyword list, a KST normalization has been done. Therefore, there is a posterior probability together with its corresponding KST normalized score for every detection in the detection list. The two scores can be used to train the classifier. The final discriminative score output by the MLP classifier is then:

$$s_{final} = score(YES, CORR) + score(NO, MISS). \quad (3)$$

We also take the number of sub-word units in the keyword into consideration. As have been stated, each keyword consists of words. Each word consists of characters. We denote the number of words in a keyword as $\#words$, and the number of characters in a keyword as $\#characters$. Scores are normalized according to the number of sub-word units:

$$s_{word} = s^{(1/\#words)}, \quad (4)$$

$$s_{char} = s^{(1/\#characters)}. \quad (5)$$

Assume that a keyword W consists of n sub-word units and we denote each unit as w_i. The probability of the keyword can be written as:

$$s(W) = \prod_{i=1}^{n} s(w_i). \quad (6)$$

Therefore, normalized scores in Eq. 4 and Eq. 5 are geometric means of the original score and they can describe the stableness of every sub-word unit in the keyword. The two scores also try to eliminate the effect of different lengths of keywords to some extent. Similarly, a frame-level normalized score is defined as:

$$s_{frame} = s^{(1/\#frames)}, \quad (7)$$

where $\#frames$ is the number of frames where the detection lasts.

Both the posterior probability and KST normalized score are normalized according to Eq. 4, Eq. 5 and Eq. 7. Then we have 8 different scores for training the classifier.

4.3. LSTM-CTC based verification setup

We use the 3 hours of transcribed data of the VLLP for training the LSTM-CTC based verification system. Firstly, we split the utterance into single words according to the transcriptions and 24600 samples are obtained. 20000 of the 24600 samples are taken as the training data and the rest as the valid data. Due to the very limited training data, a quite naive LSTM neural network is trained using the CTC loss function. The LSTM network only has one hidden layer and the hidden size is set to be 100. 54 output char labels are set including a special "blank" label needed for the CTC loss function.

The training data for the verification system is rather limited. To obtain a good enough verification system, we try to "borrow data" from other languages and therefor multilingual bottleneck (MBN) features ([25], [26]) are employed. The MBN features are trained using 6 other Babel language packs supplied for the OpenKWS15 Evaluations. A DNN with 7 hidden layers are trained to extract the MBN feature. The sixth hidden layer is a linear layer with 128 neurons, which produces the MBN features. Other hidden layers are sigmoid layers with 1500 neurons. The inputs are 40-dimensional Mel filterbank features plus 3 dimensional pitch features, and their first- and second-order derivatives. After the 128-dimensional MBN features have been extracted, the 9 consecutive feature frames are concatenated and the feature dimension is reduced to 40 using LDA and semi-tied covariance transform. More details about the training of the MBN features can be found in [20].

After the LSTM-CTC network has been trained, the CTC loss is computed for both the training detection list and the evaluation detection list. And the CTC loss is taken as another confidence score for each detection. The same normalization in Eq. 4, Eq. 5 and Eq. 7 is done to the CTC score. That means the LSTM-CTC verification system introduces another 4 scores for every detection.

To verify whether the verification system does help improve the performance of the keyword search system. Some analysis has been done and the results are presented in Table 1. In Table 1, we denote the posterior probability as "Posterior" and the KST normalized score as "KST". The CTC score is denotes as "CTC".

Table 1: *The arithmetic mean values of different scores for each class in the training detection list.*

	YES, CORR	NO, MISS	YES, FA	NO, CORR!DET
$Posterior$	0.7762	0.0978	0.3271	0.0123
$Posterior_{word}$	0.7975	0.1283	0.3563	0.0214
$Posterior_{char}$	0.9316	0.5303	0.7462	0.3160
$Posterior_{frame}$	0.9894	0.9009	0.9585	0.8229
KST	0.9257	0.1328	0.7697	0.0120
KST_{word}	0.9346	0.1744	0.7888	0.0516
KST_{char}	0.9856	0.5867	0.9562	0.3681
KST_{frame}	0.9981	0.8874	0.9945	0.7178
CTC	0.0147	0.0065	0.0039	0.0025
CTC_{word}	0.0222	0.0123	0.0062	0.0041
CTC_{char}	0.3171	0.1944	0.1826	0.1158
CTC_{frame}	0.8470	0.7655	0.7857	0.6960

From the results in Table 1, we can see that the CTC score can help differ every class from each other, especially between the class "NO, MISS" and "YES, FA". A big margin between the two classes is needed to build a high-performance keyword search system. An ideal condition is that all the detections in class "YES,FA" can have a lower score than the ones in class "NO,MISS". However, for the posterior probability and the

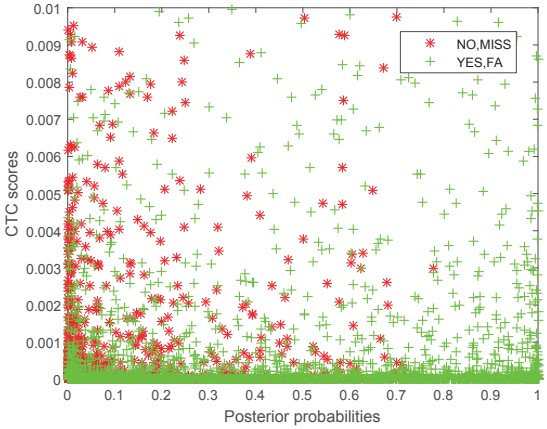

Figure 2: *Comparison of posterior probabilities and CTC scores for class "YES, FA" and "NO, MISS".*

KST normalized score, the average mean values of class "YES, FA" is higher than that of class "NO, MISS", which makes it difficult to exclude false alarms from the correct hits. For further analysis, original posterior probabilities and CTC scores for classes "YES, FA" and "NO, MISS" have been compared in Figure 2. Figure 2 shows that for detections with relatively low posterior probabilities, CTC scores can help find out true hits.

We directly take the CTC score as the final score and evaluate the performance against the baseline system. The results are in Table 2. For the baseline keyword search system, the word

Table 2: *The performance of taking the CTC score directly as the final score.*

	ATWV	MTWV
Baseline	0.4863	0.4917
CTC Score	0.3140	0.3170

posterior probability is taken as the final score. KST normalization is done before evaluation for both the baseline experiment and the CTC score experiment. From the results, we can see that though the CTC score can help differ different classes, the score itself is not a perfect measure for keyword search. Therefore, we only take CTC scores as features input to the discriminative classifier in the following experiments.

5. Results and discussions

5.1. Discriminative score calibration results

As has been stated, we try to do discriminative score calibration without having access to lattices, confusion networks or some expert knowledge about the target language. Our baseline discriminative score calibration experiment only employs the word posterior probability and the KST normalized score. This experiment is denoted as "Discriminative_2features".

Then scores normalized by the number of sub-word units proposed in this work is added and the experiment adding 6 more normalized scores is denoted as "Discriminative_8features".

Finally, the CTC scores output by the verification system is added, and the experiment is denoted as "Discriminative_12features".

For all the discriminative score calibration experiments, the discriminative score output by the MLP is computed according to Eq. 3. Each experiment also includes a KST normalization before evaluation. The results are shown in Table 3. We can see

Table 3: *The discriminative score calibration results.*

	ATWV	MTWV
$Baseline$	0.4863	0.4917
$Discriminative\_2features$	0.5018	0.5047
$Discriminative\_8features$	0.5090	0.5131
$Discriminative\_12features$	**0.5206**	**0.5234**

that discriminative score calibration improves the performance consistently, even trained only using the posterior probability and the KST normalization score. By adding 6 novel normalized scores proposed here, the relative improvement of ATWV over the baseline system can reach to 4.67%. After the CTC scores are added, the improvement can be furthered to 7.05%.

5.2. Discussions

The results above demonstrate the effectiveness of the three normalized scores proposed in this work. The experiments provide a possible method to improve the quality of confidence scores for keyword search, even if only the detection lists are supplied. That is to say, normalizing scores using the number of sub-word units can provide some extra information. It indicates that for a detection, the correctness of each component in it is quite important. This is very similar to the way a human judges whether a segment of speech is indeed some specified word sequence. We human beings always try to distinguish the minor difference. Some further exploration will be done looking into the details of a detection in the future.

In Section 4.3, it has been shown that the CTC loss has the potential to differ false alarms from correct hits. However, when taking the CTC loss as the confidence score directly, the performance is rather bad. The reason may be the serve lack of training data. The VLLP only consists of 3 hours' carefully transcribed speech and that may limit the potential of our verification system. In fact, LSTM neural networks with more hidden units and bi-directional LSTM neural networks have also been tried, while didn't bring much improvement. Assuming more training data is supplied, the performance of taking the CTC loss as the final score may be much better. Besides, the LSTM-CTC based verification system is built only for verifying char sequence. Some different sub-word units such as phonemes and morphemes can also be tried.

6. Conclusions

In this paper, we have proposed three novel normalized scores for discriminative score calibration. A LSTM-CTC based verification system using only 3 hours of transcribed data is also built to supply extra information. Experiments indicate that the three normalized scores and the verification loss can help improve the performance of discriminative score calibration. Training an MLP classifier with score features extracted from the detection list and the CTC loss, we can obtain a relative improvement of up to 7.05% over the baseline.

7. References

[1] G. Evermann and P. Woodland, "Large vocabulary decoding and confidence estimation using word posterior probabilities," in *Proc. ICASSP*, 2000, pp. 1655–1658.

[2] L. Mangu, E. Brill, and A. Stolcke, "Finding consensus in speech recognition: word error minimization and other applications of confusion networks," *Computer Speech & Language*, vol. 14, no. 4, pp. 373–400, 2000.

[3] D. Hillard and M. Ostendorf, "Compensating for word posterior estimation bias in confusion networks," in *Proc. ICASSP*, 2006, pp. 1153–1156.

[4] J. Tejedor, D. T. Toledano, M. Bautista, S. King, D. Wang, and J. Cols, "Augmented set of features for confidence estimation in spoken term detection," in *Proc. Interspeech*, 2010.

[5] D. R. H. Miller, M. Kleber, C. L. Kao, O. Kimball, T. Colthurst, S. A. Lowe, R. M. Schwartz, and H. Gish, "Rapid and accurate spoken term detection," in *Proc. Interspeech*, 2007.

[6] J. Mamou, J. Cui, X. Cui, M. J. F. Gales, B. Kingsbury, K. Knill, L. Mangu, D. Nolden, M. Picheny, B. Ramabhadran, R. Schluter, A. Sethy, and P. C. Woodland, "System combination and score normalization for spoken term detection," in *Proc. ICASSP*, 2013, pp. 8272–8276.

[7] B. Zhang, R. Schwartz, S. Tsakalidis, L. Nguyen, and S. Matsoukas, "White listing and score normalization for keyword spotting of noisy speech," in *Proc. Interspeech*, 2012.

[8] V. Soto, E. Cooper, L. Mangu, A. Rosenberg, and J. Hirschberg, "Rescoring confusion networks for keyword search," in *Proc. ICASSP*, 2014, pp. 7138–7142.

[9] V. T. Pham, H. Xu, N. F. Chen, S. Sivadas, B. P. Lim, E. S. Chng, and H. Li, "Discriminative score normalization for keyword search decision," in *Proc. ICASSP*, 2014, pp. 7078–7082.

[10] R. Nakatani, T. Takiguchi, and Y. Ariki, "Two-step correction of speech recongnition errors based on N-gram and long contextual information," in *Proc. Interspeech*, 2013, pp. 3747–3750.

[11] J. Lafferty, A. McCallum, and F. Pereira, "Conditional random fields: Probabilistic models for segmenting and labeling sequence data," in *Proc. ICML*, 2001.

[12] Z. Lv, M. Cai, W. Q. Zhang, and J. Liu, "Calibration of word posterior estimation in confusion networks for keyword search," in *Proc. APSIPA*, 2015, pp. 148–151.

[13] D. Karakos, R. Schwartz, S. Tsakalidis, L. Zhang, S. Ranjan, T. Ng, R. Hsiao, G. Saikumar, I. Bulyko, L. Nguyen, J. Makhoul, F. Grezl, M. Hannemann, M. Karafiat, I. Szoke, K. Vesely, L. Lamel, and V. B. Le, "Score normalization and system combination for improved keyword spotting," in *Proc. ASRU*, 2013, pp. 210–215.

[14] Y. N. Chen, C. P. Chen, H. Y. Lee, C. A. Chan, and L. S. Lee, "Improved spoken term detection with graph-based re-ranking in feature space," in *Proc. ICASSP*, 2011, pp. 5644–5647.

[15] H. Lee and L. Lee, "Enhanced spoken term detection using support vector machines and weighted pseudo examples," *IEEE TRANSACTIONS ON AUDIO, SPEECH, AND LANGUAGE PROCESSING*, vol. 21, no. 6, pp. 1272–1284, 2013.

[16] H. Lee, Z. Y., C. E., and G. J., "Graph-based re-ranking using acoustic feature similarity between search results for spoken term detection on low-resource languages."

[17] "KWS15 keyword search evaluation plan," http://www.nist.gov/itl/iad/mig/upload/KWS15-evalplan-v05.pdf, 2015.

[18] A. Graves, S. Fernandez, F. Gomez, and S. J., "Connectionist temporal classification: labelling unsegmented sequence data with recurrent neural networks," in *Proc. ACM*, 2006.

[19] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[20] M. Cai, Z. Lv, C. Lu, J. Kang, L. Hui, Z. Zhang, and J. Liu, "High-performance swahili keyword search with very limited language pack: The thuee system for the openkws15 evaluation," in *Proc. ASRU*, 2015, pp. 215–222.

[21] N. Jaitly and G. E. Hinton, "Vocal tract length pertubation (VTLP) improves speech recognition," in *Proc. ICML*, 2013.

[22] D. Povey, L. Burget, M. Agarwal, P. Akyazi, F. Kai, A. Ghoshal, O. Glembek, N. Goel, K. M., A. Rastrow, R. C. Rose, S. P., and T. S., "The subspace Gaussian mixture modela structured model for speech recognition," *Computer Speech & Language*, vol. 25, pp. 404–439, 2011.

[23] M. Cai, Y. Shi, J. Kang, J. Liu, and T. Su, "Convolutional maxout neural networks for low-resource speech recogition," in *Proc. ISCSLP*, 2014, pp. 133–137.

[24] J. Richards, M. Ma, and A. Rosenberg, "Using word burst analysis to rescore keyword search candidates on low-resourece languages," in *Proc. ICASSP*, 2014, pp. 7874–7878.

[25] K. M. Knill, M. J. F. Gales, S. P. Rath, P. C. Woodland, C. Zhang, and S.-X. Zhang, "Investigation of multilingual deep neural networks for spoken term detection," in *Proc. ASRU*, 2013, pp. 138–143.

[26] J. T. Huang, J. Li, D. Yu, L. Deng, and Y. Gong, "Crosslanguage knowledge transfer using multilingual deep neural network with shared hidden layers," in *Proc. ICASSP*, pp. 7304–7308.