A Small-Footprint End-to-End KWS System in Low Resources

Gui-Xin Shi
Beijing National Research Center for Information Science and Technology, Department of Electronic Engineering, Tsinghua University
Beijing, China
shiguixin2019@tsinghua.edu.cn

Wei-Qiang Zhang
Beijing National Research Center for Information Science and Technology, Department of Electronic Engineering, Tsinghua University
Beijing, China
wqzhang@tsinghua.edu.cn

Hao Wu
Beijing National Research Center for Information Science and Technology, Department of Electronic Engineering, Tsinghua University
Beijing, China

Yao Liu
China General Technology Research Institute
Beijing, China

ABSTRACT

In this paper, we propose an efficient end-to-end architecture, based on Connectionist Temporal Classification (CTC), for low-resource small-footprint keyword spotting (KWS) system. For a low-resource KWS system, it is difficult for the network to thoroughly learn the features of keywords. The intuition behind our new model is that a priori information of the keyword is available. In contrast to the conventional KWS system, we modify the label set by adding the preset keyword(s) to the original label set to enhance the learning performance and optimize the final detection task of the system. Besides, CTC is applied to address the sequential alignment problem. We employ GRU as the encoding layer in our system because of the dataset small. Experiments using the WSJ0 dataset show that the proposed KWS system is significantly more accurate than the baseline system. Compared to the character-level-only KWS system, the proposed system can obviously improve the performance. Furthermore, the improved system works well in terms of low resource conditions, especially for long words.

CCS CONCEPTS

• **Computing methodologies → Speech recognition**

KEYWORDS

Connectionist Temporal Classification (CTC); G ated Recurrent Units (GRUs); low-resource small-footprint keyword spotting (KWS); label set

1 INTRODUCTION

Keyword spotting (KWS) is a speech task to detect one or more specific word(s) in an audio signal. The simplest approach to KWS is to use Automatic Speech Recognition (ASR) to generate a transcript of the speech corpus, then treat this as a text search problem [1, 2]. Keyword detection can be regarded as a special application of ASR, but it is different from ASR [3] because KWS only focuses on the recognition task of the user's predetermined keyword. ASR gives the transcription of all utterances/speech database. However, KWS asks much simpler questions:" Is this query term in this utterance? If yes, where is it?" [4].

KWS system based on Large Vocabulary Continuous Speech Recognition (LVCSR) has become the most popular technology [5]. However, the traditional LVCSR system also has many shortcomings. In a popular KWS approach, the LVCSR based ASR system processes the speech database and converts it into lattices containing multiple hypothesized word sequences along with their associated confidence scores and time stamps. A typical ASR system consists of several modules including acoustic, lexicon, and language models [6,7]. Although the last decade has witnessed dramatic improvements in acoustic and language models, current LVCSR systems still suffer heavily from the scaffolding of complicated legacy architectures. Problems that need to be eliminated are as follows.

1 Stepwise refinement: The speech data needs to be pre-processed. Currently, most of the current state-of-the-art ASR systems use a hybrid DNN-Hidden Markov model (HMM) system. Given the input acoustic feature vector, the DNN predicts the posterior probability of HMM states.

2 Incoherence in optimization: The above modules (acoustic, lexicon, and language models, etc.) are usually optimized separately with different objectives. Each module is not trained to match the other modules, which may result in incoherence in optimization. The independence of model training means that the objective of each model during training is also inconsistent with the final optimization goal of the system. The performance improvement of one model does not necessarily improve the performance of the whole KWS system. What is worse, it is difficult to collaboratively optimize these modules.

3 Unavailability in low resources: The training of different models is independent and requires a large amount of data with labels. The typical LVCSR-based KWS system needs a large amount of data to guarantee good performance.

SPML '19, November 27–29, 2019, Hangzhou, China

ACM ISBN 978-1-4503-7221-3/19/11…$15.00
DOI: https://doi.org/10.1145/3372806.3372822

Consequently, it is a very expensive and complicated task to establish a KWS system based on LVCSR. What is more, it is quite difficult for nonexperts to use or develop such LVCSR-based KWS systems for new applications. Recently, end-to-end (E2E) ASR based KWS systems have become more and more popular. Then scholars have proposed KWS systems based on E2E ASR [1,6,7,8,9,10]. The core of E2E ASR technology is to train an independent deep neural network to operate the mapping from speech sequence to text sequence, instead of training various modules of traditional ASR. The Connectionist Temporal Classification (CTC) is one of the best solutions for implementing an E2E KWS system. CTC is a direct approach for sequence labeling tasks because it allows the network to predict labels at any point in input sequences [8]. It simplifies the KWS system through a single Recurrent Neural Network (RNN) and does not need the label alignment and pre-segmentation.

Chris Lengerich and Awni Hannun [9] propose an end-to-end system that can achieve high-precision KWS and voice activity detection without retraining. Based on the CTC and LSTM-RNN, the reference [10] constructs an E2E KWS system for Mandarin. [11] improved the work in [10] and designs another E2E KWS system based on CTC and RNN. This system uses Mandarin syllables as the output labels, rather than the phonemes or characters as in [10]. Inspired by these works, our proposed E2E KWS system is also based on the combination of CTC and RNN, but with a modified output label set. The intuition behind our approach is that a priori information of the keyword is available.

Based on the CTC framework of [9], we propose an improved E2E KWS system, which integrates the prior knowledge of target keywords into the front-end training process, achieving the accuracy of target keyword-specific detection. Usually, the label set in the CTC module contains 26 English characters, spaces, English periods and English quotation marks, and these 29 labels are also used in the training and recognition process. If we know the target keyword before training the model, we can add an extra-label of this keyword to enhance the learning performance and optimize the final detection task of the system. Experiments using the WSJ0 dataset show that the improved KWS system is significantly more accurate than the baseline system [9].

The rest of this paper is as follows. Section 2 gives an overview of the KWS system used in this paper. The experimental setup, as well as the results, are described in Section 3 and Section 4, respectively. Finally, Section 5 concludes the paper and discusses future work.

2 PROPOSED END-TO-END KWS SYSTEM

Our proposed KWS system consists of two modules, the CTC-based neural network, and the searching module. The overall architecture of our proposed KWS system is shown in Figure 1. After performing a fast Fourier transformation on the input speech data, we extract the acoustic features from the spectrogram. Then, we can use CTC to train the model.

2.1 Connectionist Temporal Classifier (CTC)

The difficulty of training comes from there being many more observations than there are labels. For example, in speech audio there can be multiple time slices which correspond to a single phoneme. Since we don't know the alignment of the observed sequence with the target labels, we predict a probability distribution at each time step. The concept of CTC was introduced in 2006 [12]. CTC trains recurrent neural networks (RNNs) such as LSTM networks to tackle sequence problems where the timing is variable. CTC refers to the outputs and scoring and is independent of the underlying neural network structure. The main idea of CTC is to add a blank symbol to the label set and RNN to deal with different lengths of sequences.

Given an input sequence $x = (x_1, x_2, \ldots, x_T)$ of length T, define an RNN as a continuous map $\mathcal{N}_\omega : (\mathbb{R}^m)^T \mapsto (\mathbb{R}^n)^T$, where m and n are inputs and outputs size, respectively. Denote $y = \mathcal{N}_\omega(x) = (y_1, y_2, \ldots, y_T)$ by the output sequence of the CTC network, in which y_k^t is the activation of output unit k at time t. Usually, the output units include all English characters (L), and blank symbol. In our work, we add the preset keyword(s) to the traditional label set, resulting in a novel label set:

$$L' = L \cup \{ \text{keyword} \} \cup \{\text{blank}\} \tag{1}$$

Then y_k^t is the probability that label k occurs at time t, which defines a distribution over the set L'^T of length T sequences over the alphabet L':

$$p(\pi|x) = \prod_{t=1}^{T} y_{\pi_t}^t \tag{2}$$

where $y_{\pi_t^t}$ denotes the π_t-th output unit at time t. Note that we need to remove the blank label and repeated labels, to obtain the right output sequence. Define a map function

$$B : L'^T \mapsto L^{<T} \tag{3}$$

where L'^T only includes non-blank labels. With the mapping relationship, the probability of $l \in L^{<T}$ can be expressed as

$$p(y|x) = \sum_{\pi \in \mathrm{B}^{-1}(y)} p(\pi|x) = \sum_{\pi \in \mathrm{B}^{-1}(y)} \prod_{t=1}^{T} p\left(\pi_t|x_t\right) \tag{4}$$

Training by Backpropagation Gradient Descent, the object function is

$$\theta = \arg\min_\theta \sum_{(x,y)\in S} -\log(p(y|x;\theta)) \tag{5}$$

where S is the training data and can be calculated by Forward-Backward Algorithm according to Graves et.al [12].

2.2 Network Architecture

CTC networks usually employ RNNs such as long short-term memory (LSTM) networks for training. Gated recurrent units (GRUs) are a gating mechanism in RNNs, introduced in 2014 by Kyunghyun Cho et. al [13]. The GRU is like a LSTM with a forget gate [14]. But it lacks an output gate, therefore, it has fewer parameters and less computation load than LSTM. GRUs have been shown to exhibit even better performance on certain smaller datasets [15]. In this paper, we use a small dataset to train the network, so we employ GRU as the encoding layer in our system. The chart of the network architecture is shown in Figure 2. The first layer is a 2-dimensional convolution with a stride of three. And gated recurrent RNN with 256 hidden units in each layer is used in the next three layers of the neural network. Note that the

chosen parameters of CTC networks are corresponding to the best performance among different parameters.
A CTC network has a continuous output (e.g. softmax), which is fitted through training to model the probability of a label. The CTC scores can then be used with the back-propagation algorithm to update the neural network weights.

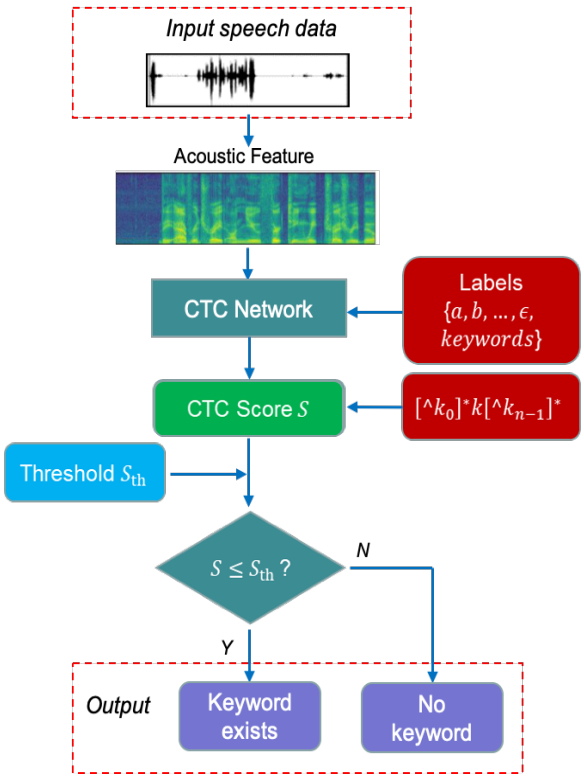

Figure 1. Overall architecture of the proposed E2E KWS system. Here the symbol ϵ represents the blank label.

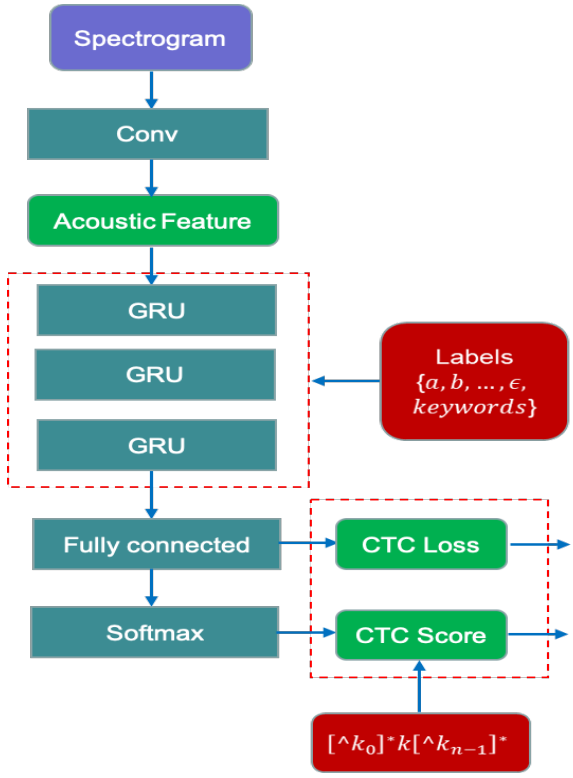

Figure 2. Network architecture.

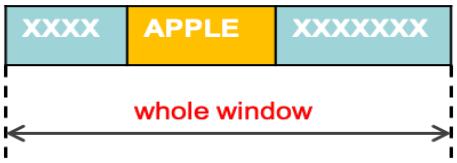

Figure 3. Schematic diagram of equation (6). The keyword is 'APPLE'.

2.3 Searching Module

For online KWS tasks, we want to detect the keyword(s) with low latency. Thus, we choose an easy way to handle the searching module. We use a moving window to segment a whole utterance into speech clips $x_{t:t+T_w}, 0 \le t \le T - T_w$, where T_w is the given length of moving windows. It is the truth that the lengths of different keywords vary. It is inefficient for users to change the window length every time. In this paper, we address this problem in a regularized way, instead of changing the window length. Intuitively, the probability $p\left(y|x_{t:t+T_w}\right)$ increases obviously after the keyword occurs. For a given keyword k, instead of scoring itself, we score the regular expression $\left[\ ^\wedge k_0\ \right]^* k \left[^\wedge k_{n-1}\right]^*$, where k is any keyword and k_0 and k_{n-1} are the first and last character of k, respectively. To be more specific, as shown in Figure 3, this regular expression means we score the possibility of a sequence containing k. For each speech clip , we calculate the corresponding CTC score, i.e.,

$$S_{CTC}\left(k|x_{t:t+T_w}\right) = -\ln\left(p\left(\left[^\wedge k_0\right]^* k \left[^\wedge k_{n-1}\right]^* |x_{t:t+T_w}\right)\right), 0 \le t \le T - T_w \quad (6)$$

Apparently, the minimum of $S_{CTC}\left(k|x_{t:t+T_w}\right)$ corresponds to the occurrence of k. The keyword may occur anywhere in an utterance, so we should score every clip and choose the minimum. So, the CTC score is calculated according to (7)

$$\text{Score} = \min\left(S_{\text{CTC}}\left(k|x_{t:t+T_w}\right), 0 \le t \le T - T_w\right) \quad (7)$$

Given a threshold S_{th}, we have

$$\text{Classfication} = \begin{cases} \text{Positive, if Score} \le S_{\text{th}} \\ \text{Negtive, if Score} > S_{\text{th}} \end{cases} \quad (8)$$

The threshold is determined during training. In our system, is decided by the minimizing the distance method. Minimizing distance method is a way to find the threshold corresponding to

a point in the ROC curve, subject to the minimum distance to the coordinate , i.e.,

$$S_{th} \triangleq \arg\min_{S_{th}} \sqrt{(1-TPR)^2 + FPR^2} \quad (9)$$

where and are true positive rate and false positive rate, respectively.

3 EXPERIMENT SET UP

3.1 Dataset and baseline

We take the system in [9] as our baseline, which is a character-level-only E2E system with CTC loss function. The data used to train the model is the WSJ0 dataset. The whole dataset contains about 15-hour training speech with labels. To explore the performance of the proposed model with low resources condition, we use part of the WSJ0 data set to train our model. It has 4,026 utterances (8.53 hours) without noise. And the test data (WSJ test data) has 0.67-hour transcribed data including 330 sentences.
We compare the performance of the baseline and proposed systems in a consistent configuration. The model parameters are optimized with stochastic gradient descent and a minibatch size of 128 for 58 epochs. The learning rate and momentum parameters are 2e-3 and 0.95, chosen to optimize the speed of convergence. The learning rate anneals by a factor of 0.9 every 800 iterations.
The architecture of the network is as described in Section 2.1 and Section 2.2. The filters for the convolution layer are 15 by 32 over the time and frequency dimensions respectively. We use 32 filters in all models. According to (6), we classify an utterance as positive if the CTC score is smaller than the threshold determined during training.

3.2 Feature Extraction

The input audio file is first framed, the window length is 8ms, and the frame is shifted by 1ms. We perform a fast Fourier transform on each frame of data before taking the logarithm. As for the moving windows, according to the keyword length and wave file duration, we choose 800 milliseconds as window size and 50 milliseconds as step size in the testing part.

3.3 Metrics

In the following experiments, we evaluate our system by calculating the F1 score, accuracy, precision, and recall. F1 value is calculated by (10)

$$F1 = \frac{2\times\text{Precision}\times\text{Recall}}{\text{Precision}+\text{Recall}} \quad (10)$$

where

$$\text{Precision} = \frac{\text{number of true positive}}{\text{number of true positive+false positive}} \times 100\% \quad (11)$$

$$\text{Recall} = \frac{\text{number of true positive}}{\text{number of true positive+false negtive}} \times 100\%. \quad (12)$$

$$\text{Accuracy} = \frac{\text{number of (true positive+true negtive)}}{\text{number of all samples}} \times 100\%. \quad (13)$$

Table 1. Results of keyword 'president'

	Baseline	Improved system
F1 score	0.8000	**0.9231**
Accuracy	0.9909	**0.9970**
Precision	0.6667	**0.8571**
Recall	1	1

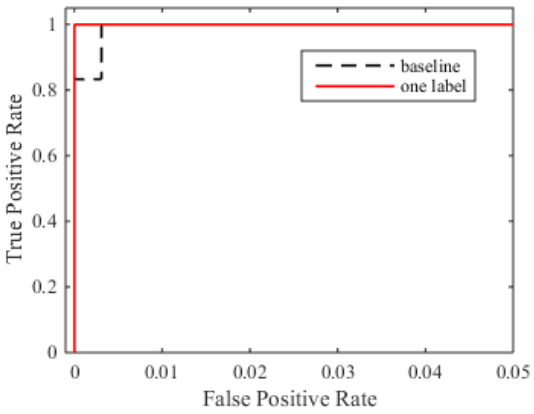

Figure 4. ROC curves for keyword 'president'. Here the figure label 'one-label' represents our proposed KWS system with one keyword.

4 RESULTS AND ANALYSIS

In real life, we may use one or more word(s) to wake up the device. In a KWS system, users may also be interested in more than one keyword. Therefore, to evaluate our system overall, the experiment consists of two scenarios: one keyword only and multiple keywords.

4.1 Results of experiment 1

In experiment 1, we explore the impact of our proposed method on the one-keyword system by ROC curves, F1 score, accuracy, precision, and recall. Select 'president' as the only keyword for the KWS system. KWS performance is measured by plotting a Receiver Operating Curve (ROC), which calculates the true positive rate (TPR) per false positive rate (FPR). The higher the TPR per FPR rate is, the better. The ROC curves are plotted in Figure 4. We observe that the TPR of our method is more satisfied than the baseline, on condition of low FPR. The results of other metrics are shown in Table 1. Then change the keyword to 'quarter' and repeat the experiment. The results are shown in Figure 5 and Table 2. It can be seen that the proposed system outperforms the baseline. A prior knowledge of the keyword contributes to this improvement.

4.2 Results of experiment 2

Then we compare the performance of models with one and more keywords in experiment 2. Because the CTC loss is the continuous multiplication of each label's probability in the keyword (as expressed in (5)), we choose eleven different words which have different lengths and syllables in pronunciation. Also, because of the

small training dataset, we ensure the occurrence times (i.e., positive utterances) of all the keywords in the training dataset are more than 50. The chosen keywords and the number of their occurrences in the training dataset are listed in Table 3.

We evaluate our system by calculating the F1 score, accuracy, precision, and recall. The mean values of those metrics for both the baseline and our systems are shown in Table 4. To be specific, Figure 6 and Figure 7 compare the F1 values and accuracies of different keywords for different systems. It should be noted that the results of the one-label system are obtained through retraining the model, with only one known keyword each time. In contrast to the one-label system, the mul-label system trains the model with all the preset keywords in one time.

Table 2. Results of keyword 'quarte'

	Baseline	Improved system
F1 score	0.7059	**0.7586**
Accuracy	0.9697	**0.97880**
Precision	0.5455	**0.6471**
Recall	**1**	0.9167

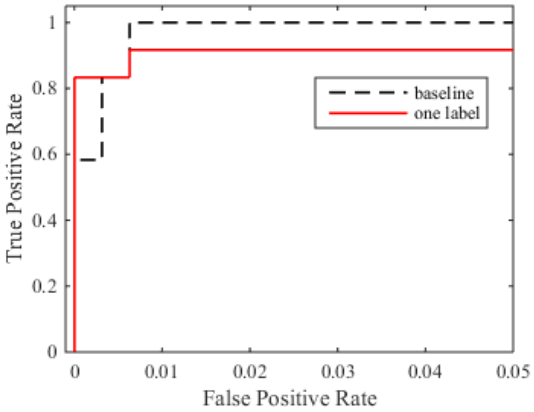

Figure 5. ROC curves for keyword 'quarter'. Here the figure label 'one-label' represents our proposed KWS system with one keyword.

Table 3. Chosen keywords occurrence times in training dataset

Keyword	Occurrence times in the straining setm
company	306
dollar	297
percent	195
market	160
million	158
stock	141
thousand	91
government	85
president	81
quarter	81
financial	59

Table 4. Mean values of metrics for the baseline and proposed systems

	Baseline	One-label	Mul-label
F1 score	0.8303	0.9017	**0.9183**
Accuracy	0.9832	0.9854	**0.9882**
Precision	0.7805	0.8536	**0.9014**
Recall	0.9522	**0.9631**	0.9457

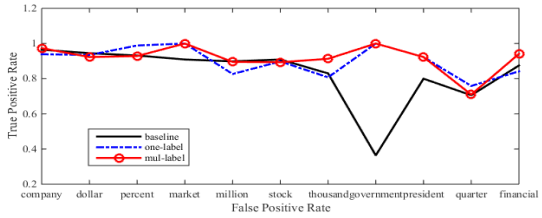

Figure 6. F1 scores versus different keywords. Here 'one-label' and 'mul-label' represent our proposed KWS system with one keyword and multiple keywords, respectively.

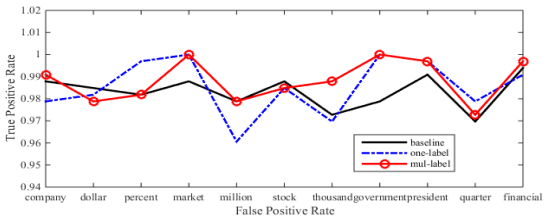

Figure 7. Accuracies versus different keywords. Here 'one-label' and mul-label' represent our proposed KWS system with one keyword and multiple keywords, respectively.

The proposed models (including both the one-label and mul-label system) work well in terms of low resource conditions, especially for long words ('government', 'president', 'thousand'). This is intuitive since more characters will increase the character recognition error. With the extra label(s), our system can recognize the keyword more easily, leading to better KWS performance. As for some keyword ('company' and 'dollar') that occurs in high frequencies, the performance gap between the baseline and improved systems shrink sharply. The reason is that the information about positive utterances is sufficient to detect the keyword. What is more, since the training data is limited and the system is small foot-print, the time cost in training is relatively low compared to those LVCSR methods.

5 CONCLUSIONS

In this paper, we propose an efficient end-to-end KWS system, in the condition of the low resource. CTC network is employed to avoid sequence alignment and segment. Since data resource is limited, GRU is applied as the encoding layer. Different from other end-to-end KWS systems, we modify the label set by adding keyword(s) into the original label set. Experiments demonstrate the improvement of the performance because the network can learn features of the keyword(s) better. We evaluate our system by the F1 score, accuracy, precision, recall and ROC curves. Compared to the character-level-only KWS system, the proposed system performs better. In the case of multiple keywords, we prefer to add these keywords in the label set before training the model. In the case of one keyword, although the system needs to be retrained when we change keyword(s), it is not a big problem under the condition of the small foot-print system and low resource. Furthermore, the system can be applied to the KWS system for other languages that lack training data. In the future, we intend to deal with noisy data.

6 ACKNOWLEDGMENTS

This work was supported by the National Natural Science Foundation of China under Grant No. U1836219 and the National Key R&D Program of China. The corresponding author is Wei-Qiang Zhang.

7 REFERENCES

[1] Sainath, T. N., and Parada, C. (2015). Convolutional neural networks for small-footprint keyword spotting. In Sixteenth Annual Conference of the International Speech Communication Association.

[2] Chen, G., Parada, C., and Heigold, G. 2014. Small-footprint keyword spotting using deep neural networks. In 2014 IEEE International Conference on Acoustics, Speech and Signal Processing (May, 2014) 4087-4091. DOI= 10.1109/ICASSP.2014.6854370.

[3] Saraclar, M., Sethy, A., Ramabhadran, B., Mangu, L., Cui, J., Cui, X., ... and Mamou, J. 2013. An empirical study of confusion modeling in keyword search for low resource languages. InÂă2013 IEEE Workshop on Automatic Speech Recognition and UnderstandingÂă(December, 2013) 464-469. DOI= 10.1109/ASRU.2013.6707774.

[4] Audhkhasi, K., Rosenberg, A., Sethy, A., Ramabhadran, B., and Kingsbury, B. 2017. End-to-end ASR-free keyword search from speech. IEEE Journal of Selected Topics in Signal Processing, 11, 8, 1351-1359. DOI= 10.1109/JSTSP.2017.2759726

[5] Miller, D. R., Kleber, M., Kao, C. L., Kimball, O., Colthurst, T., Lowe, S. A., and Gish, H. (2007). Rapid and accurate spoken term detection. In Eighth Annual Conference of the international speech communication association.

[6] Kim, S., Hori, T., and Watanabe, S. 2017. Joint CTC-attention based end-to-end speech recognition using multi-task learning. InÂă2017 IEEE international conference on acoustics, speech and signal processing. (March, 2017) 4835-4839. DOI= 10.1109/ICASSP.2017.7953075.

[7] Watanabe, S., Hori, T., Kim, S., Hershey, J. R., and Hayashi, T. 2017. Hybrid CTC/attention architecture for end-to-end speech recognition.ÂăIEEE Journal of Selected Topics in Signal Processing 11, 8 (December,2017) 1240-1253. DOI= 10.1109/JSTSP.2017.2763455

[8] Graves, A., and Jaitly, N.2014. Towards end-to-end speech recognition with recurrent neural networks. InÂăInternational conference on machine learningÂă(January,2014) 1764-1772.

[9] Lengerich, C., and Hannun, A. 2016. An end-to-end architecture for keyword spotting and voice activity detection.ÂăarXiv preprint arXiv:1611.09405.

[10] Bai, Y., Yi, J., Ni, H., Wen, Z., Liu, B., Li, Y., and Tao, J. 2016. End-to-end keywords spotting based on connectionist temporal classification for mandarin. InÂă2016 10th International Symposium on Chinese Spoken Language Processing (October,2016)Âă1-5. DOI= 10.1109/ISCSLP.2016.7918460

[11] Wang, Y., and Long, Y. 2018. Keyword spotting based on CTC and RNN for mandarin chinese speech. InÂă2018 11th International Symposium on Chinese Spoken Language Processing (November, 2018), 374-378. DOI= 10.1109/ISCSLP.2018.8706631

[12] Graves, A., FernÃąndez, S., Gomez, F., and Schmidhuber, J. 2006. Connectionist temporal classification: labelling unsegmented sequence data with recurrent neural networks. InÂăProceedings of the 23rd international conference on Machine learningÂă(June, 2006) 369-376. DOI= http://doi.acm.org/ 10.1145/1143844.1143891

[13] Cho, K., Van MerriÃńnboer, B., Gulcehre, C., Bahdanau, D., Bougares, F., Schwenk, H., and Bengio, Y. 2014. Learning phrase representations using RNN encoder-decoder for statistical machine

[14] Gers, F. A., Schmidhuber, J., and Cummins, F. 1999. Learning to forget: Continual prediction with LSTM.

[15] Chung, J., Gulcehre, C., Cho, K., and Bengio, Y. 2014. Empirical evaluation of gated recurrent neural networks on sequence modeling.ÂăarXiv preprint arXiv:1412.3555.