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

Unrestricted Vocabulary Keyword Spotting using LSTM-CTC

Yimeng Zhuang, Xuankai Chang, Yanmin Qian, Kai Yu

Key Laboratory of Shanghai Education Commission for Intelligent Interaction and Cognitive Engineering SpeechLab, Department of Computer Science and Engineering
Shanghai Jiao Tong University, Shanghai, China

{yimmon, xuank, yanminqian, kai.yu}@sjtu.edu.cn

Abstract

Keyword spotting (KWS) aims to detect predefined keywords in continuous speech. Recently, direct deep learning approaches have been used for KWS and achieved great success. However, these approaches mostly assume fixed keyword vocabulary and require significant retraining efforts if new keywords are to be detected. For unrestricted vocabulary, HMM based keyword-filler framework is still the mainstream technique. In this paper, a novel deep learning approach is proposed for unrestricted vocabulary KWS based on Connectionist Temporal Classification (CTC) with Long Short-Term Memory (LSTM). Here, an LSTM is trained to discriminant phones with the CTC criterion. During KWS, an arbitrary keyword can be specified and it is represented by one or more phone sequences. Due to the property of peaky phone posteriors of CTC, the LSTM can produce a phone lattice. Then, a fast substring matching algorithm based on minimum edit distance is used to search the keyword phone sequence on the phone lattice. The approach is highly efficient and vocabulary independent. Experiments showed that the proposed approach can achieve significantly better results compared to a DNN-HMM based keyword-filler decoding system. In addition, the proposed approach is also more efficient than the DNN-HMM KWS baseline.

Index Terms: Keyword Spotting, Long Short-Term Memory, Connectionist Temporal Classification

1. Introduction

Keyword spotting (KWS) is one of the most widely used speech-related techniques, which is a highly accurate and efficient recognizer specializing in the detection of some words or phrases of interest in continuous speech. KWS has many applications, such as speech data mining, audio indexing, phone call routing, phone call monitoring, voice command. A famous application is Googles voice search [1] which continuously monitors keyword "OK Google" to initiate voice input.

Although a large number of keyword spotting techniques have been proposed in the past several decades, most of them are categorized into four groups. The first is large vocabulary continuous speech recognition (LVCSR) based method [2, 3, 4]. This is a very traditional and straight-forward approach to keyword spotting. It involves the processes of transcribing speech into text and indexing for search, which causes large resources consumption and is vulnerable to detect out-of-vocabulary terms. Acoustic KWS [5] is another type of common KWS methods. Unlike LVSCR-based KWS, rather than using a large vocabulary covering all potential spoken words, a smaller set of designated keywords and non-keywords are modelled. A popular approach is the Keyword-Filler Hidden Markov Model [6, 7, 8]. The third is the Query-by-Example (QbyE) method [9, 10, 11], which utilizes keyword audio samples to generate a keyword template or a set of keyword templates and makes comparison between the test sample and the keyword templates to spot keyword. The last one is lattice-based approach [12, 13], which spots keywords by first performing a N-best Viterbi recognition algorithm to construct a database containing phone lattice representations for all audio and then searching the keywords on that database. Since searching is on text, the operations are very fast.

Many attempts to apply deep learning on keyword spotting task have achieved significant improvements, such as [14, 15, 16]. But most of them focus on spotting a set of specific keywords when designing the models, which leads to less extensibility. In this paper, we follow with interest developing an effective and efficient keyword spotting system for arbitrary keywords while it does not require excessive amounts of specific training data and does not need to retrain the models.

The proposed method is based on the low-level signal processing ability of Connectionist Temporal Classification (CTC) with Long Short-Term Memory (LSTM). LSTM [17, 18] is a type of recurrent neural network which has cycles feeding the activations from previous time steps. LSTM is designed for modelling of long-range temporal context that improves sequence labelling. CTC [19, 20, 21] is a criterion for training recurrent neural networks to label unsegmented sequences directly, and it is first introduced in [19]. Unlike other objective functions, CTC lets the network automatically locate and align phone labels during training. The outputs of CTC are interpreted as a probability distribution over all possible phones and an extra *blank* unit is utilized to model confusion information of speech signal. An important property of CTC that makes the proposed method possible is that the output probability distribution of CTC is observed extremely sparse and peaky.

In [22], a keyword spotter for specific keywords using BLSTM-CTC is first proposed but only the most intuitive post-processing is applied. Besides, [23] comes up with a complicated CTC-DBN decoder which is shown to outperform a Keyword-Filler Hidden Markov Model system. In this paper, the LSTM is unidirectional so the number of parameters is reasonable for many applications. An efficient and concise post-processing algorithm is also proposed, which generates a searchable lattice on the basis of the output probability distribution of LSTM-CTC and then performs search. Finally, the decision is made through threshold comparison. Besides, some measures are explored to further improve the performance of the method.

The rest of the paper is organized as follows. Section 2 describes the details of applying LSTM-CTC for a KWS task, followed by experiments in section 3. Finally, section 4 concludes

This work was supported by the Program for Professor of Special Appointment (Eastern Scholar) at Shanghai Institutions of Higher Learning, the China NSFC project No. 61573241 and the Interdisciplinary Program (14JCZ03) of Shanghai Jiao Tong University in China.

http://dx.doi.org/10.21437/Interspeech.2016-753

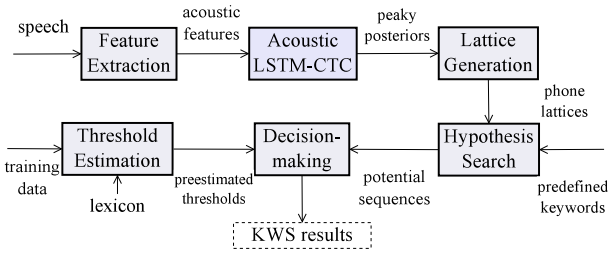

Figure 1: *Framework of the proposed LSTM-CTC method.*

the whole paper and discusses future works.

2. LSTM-CTC for KWS

The proposed LSTM-CTC KWS framework is illustrated in Figure 1. First, acoustic feature extraction is performed to reduce the redundant information of the input speech signal. Then, these feature vectors are fed frame by frame as input to a well-trained acoustic LSTM-CTC network. A phone lattice generation module receives the peaky phone posteriors of LSTM-CTC output and produces a searchable phone lattice for doing hypothesized phone sequence search, which will find out the most similar phone sequence to the target keyword phone sequence in the lattice. A decision then will be made by comparing the score of the resulting phone sequences from previous search with a pre-estimated threshold. A threshold estimation module is used to estimate different thresholds for different keywords based on a set of training data and a lexicon, rather than using a single fixed threshold for all keywords.

2.1. Phone Lattice Generation

As is shown in Figure 2, a typical LSTM-CTC network prediction consists of a series of spikes or peaks separated by *blanks*. Each spike corresponds to an output activation, which represents the posteriors of observing phones at a particular frame. *Blank* means null prediction.

The lower portion of Figure 2 shows the generated phone lattice corresponding to the above LSTM-CTC output. The first step to obtain the lattice is to locate spikes. In this paper, the most intuitive approach is adopted, i.e. by scanning the whole time range, if at a frame the total posteriors of all non-blank phones exceed a predefined threshold, h_{spike}, then that frame is a spike. Since actually the LSTM-CTC output is not perfect, continuous frames may be considered as spikes in this way and it will impair the efficiency and performance in later processes. To address this problem, the phone of the highest posterior at each frame is considered as the primary phone of that frame. If continuous frames are considered as spikes and they share the same primary phone, all these spikes are discarded except the one with highest total posterior. Experiments show that setting h_{spike} between 0.1 and 0.3 is reasonable.

In the phone lattice, each spike corresponds to a column of nodes, and each node represents a phone which has relatively large posterior in the frame where the spike locates. For a spike, phones of posteriors larger than h_{node} are selected as potential phones and appended into the corresponding nodes column.

The nodes in every two adjacent columns are fully connected. Thereby, any path from one node to another represents a potential phone sequence which consists of phones represented by nodes through the path. There is no weight on the connections between two nodes, however, nodes have posteriors denoting the probabilities of observing the phones at the spikes.

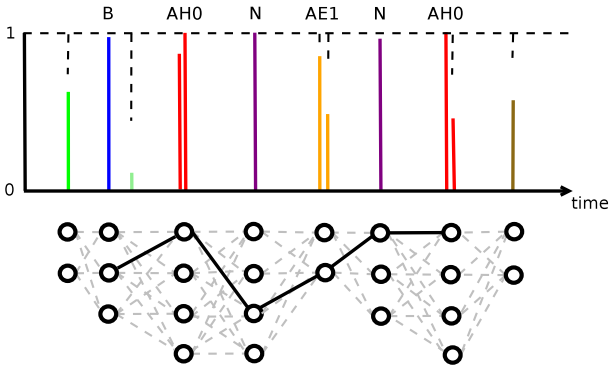

Figure 2: *LSTM-CTC output of a speech segment and the corresponding lattice. Different colours represent the posteriors of different phones. The black line in the lattice indicates a potential path and the grey dashed lines are all valid connections.*

2.2. Hypothesis Search

The lattice explained in Section 2.1 is searchable. In this section, it is described how to find out the most similar phone sequence. The term "the most similar phone sequence" means (i) the phone sequence has relatively high observation probability, and (ii) the minimum edit distance (MED) [24] between the phone sequence and the phone sequence of keyword is small.

Suppose $T = \{t_1, t_2, \cdots, t_n\}$ is a target keyword phone sequence of length n, where t_i denotes the i-th phone in T. Similarly, let $H = \{n_{ij_1}, n_{(i+1)j_2}, \cdots, n_{(i+m-1)j_m}\}$ be a hypothesized phone sequence, where n_{ij} is the j-th phone node in the i-th column of the lattice. For convenience, H is also represented by $H = \{h_1, h_2, \cdots, h_m\}$, that is, $h_k = n_{(i+k-1)j_k}$. For a test utterance, let L_H represent the corresponding CTC phone lattice containing all possible phone sequences. Then, the probability of target keyword T existing in L_H is

$$P(T|L_H) \propto P(L_H|T)P(T) \approx P(H_{max}|T)P(T) \quad (1)$$

where H_{max} is the most similar phone sequence that we want to find out and it should have the highest probability given T in L_H. Due to T is given, whether the keyword T exists or not in the test utterance is decided on the value of $P(H_{max}|T)$. Next, we find H_{max}.

$$\begin{aligned} H_{max} &= \underset{H}{\operatorname{argmax}}\, P(H|T) = \underset{H}{\operatorname{argmax}}\, \frac{P(T|H)P(H)}{P(T)} \\ &= \underset{H}{\operatorname{argmax}}\, P(T|H)P(H) \end{aligned} \quad (2)$$

where $P(H)$ is the observation probability of H, and it can be calculated by multiplying the posterior probabilities of every phone nodes of H under the unigram assumption, that is,

$$P(H) = \prod_{i=1}^{m} P(h_i|H_{1:i-1}) \approx \prod_{i=1}^{m} P(h_i) \quad (3)$$

$P(T|H)$ is not strictly estimated through multiplying the probabilities of each edit operation when computing the minimum edit distance between H and T, that is,

$$P(T|H) \triangleq \prod_{i=1}^{MED(T,H)} P(op_i|R = T, E = H) \quad (4)$$

$$P(op_i|R = T, E = H) = \begin{cases} P(insert(e_i)) & \text{if } op_i \in I \\ P(delete(r_i)) & \text{if } op_i \in D \\ P(r_i|e_i) & \text{if } op_i \in S \end{cases} \quad (5)$$

where $MED(T, H)$ indicates the minimum number of edit operations between T and H by insertions, deletions or substi-

tutions. $P(op_i|R = T, E = H)$ denotes given the reference sequence R is T and the hypothesis sequence E is H, the probability of the i-th edit operation op_i between E and R. Since MED algorithm is certain and both E and R are given, each edit operation is known. According to the type of op_i, there are three kinds of probabilities: the probability of insertion operation $P(insert(e_i))$, the probability of deletion operation $P(delete(r_i))$ and the probability of substitution operation $P(r_i|e_i)$. I, D and S denote the sets of insertion operations, deletion operations and substitution operations respectively. e_i and r_i are the phones involved in op_i (insertion and deletion involve one phone and substitution involves two) and they belong to hypothesis E and reference R respectively. These three kinds of probabilities can be estimated from prior knowledge. In this paper, they are estimated directly through a phone-level confusion matrix [25] calculated by comparing the phone alignment between an ASR hypothesis and the corresponding reference.

Instead of enumerating all possible H in the lattice, formula 2 can be efficiently computed in $O(Nnd)$ time using a dynamic programming technique, where N is the total number of nodes in the lattice, and d is the depth of the lattice. For normal keywords and h_{node}, n and d are small ($n < 15$ and $d < 5$), and N depends on the number of phones of a utterance. Therefore, the hypothesis search is efficient. In addition, to avoid underflows on digital computer when implementing the algorithm, the natural logs are applied to the calculations of probabilities.

2.3. Threshold Estimation

Since different keywords consist of different phones and for different phones the LSTM-CTC may have different modelling effects, the proposed approach tries to estimate flexible thresholds for arbitrary keywords. Define the score of a phone sequence H as $score(H) = P(T|H)P(H)$, which can be calculated by formula 3 and formula 4. Therefore, expanding $P(T|H)P(H)$ and reordering the factors, we get

$$\begin{aligned} score(H_{max}) &= \prod_{i=1}^{MED(T,H_{max})} P(op_i|R{=}T,E{=}H_{max}) \prod_{h\in H_{max}} P(h) \\ &= \{ \prod_{op_i \in D\cup S\cup M} Q^T_{L_H}(r_i)\} \{ \prod_{op_i \in I} Q^T_{L_H}(r_i)\} \end{aligned} \tag{6}$$

where

$$Q^T_{L_H}(r_i) = P(op_i|R = T, E = H_{max})P(e_i) \tag{7}$$

Here, for convenience in math, M is introduced to represent the set of phone matching (fake operations), which does not contribute to the edit distance, and let the probabilities of matching "operations" equal one so that the formula makes sense. The first part in formula 7 represents the probability of edit operation related to phone r_i and the second part is the observation probability of the phone e_i that is the phone corresponding to r_i in E. To make the formula look compact, if op_i is a deletion operation, $P(e_i)$ is nonexistent, so its value is set to 1. We estimate the threshold by averaging the Q values for each phone r_i, i.e. $Q^T_{L_H}(r_i)$. Analysis of data reveals that insertion operations are always minority, so the second part of formula 6 is ignored when estimating thresholds.

The policy of threshold estimation is as follows: for a phone x, a pre-calculated peaky posteriors output of the LSTM-CTC corresponding to a training utterance and an arbitrary keyword T that contains x are selected randomly from a set of training data and a lexicon respectively. Then the hypothesis search algorithm is performed to find out the most similar phone sequence H_{max} for T in the lattice L_H corresponding to the training utterance. After H_{max} is found, if $op_i \in S \cup D \cup M$ and r_i equals x, $Q^T_{L_H}(r_i)$ is computed through formula 7. This process is repeated N times. The average Q value for phone x is calculated as

$$\bar{Q}(x) = \frac{1}{N} \sum_{i=1}^{N} Q^T_{L_H}{}^{(i)}(x) \tag{8}$$

where $Q^T_{L_H}{}^{(i)}(x)$ is the Q value of x computed in the i-th time. For each phone, its Q value is computed in similar way. Therefore, the threshold of a given keyword $T = \{t_1, t_2, \cdots, t_n\}$ can be estimated by

$$\theta(T) = \prod_{i=1}^{n} \bar{Q}(t_i) \cdot C \tag{9}$$

where C is a constant scaling factor.

2.4. Word Boundary

In addition to normal phones, a special label "wb" is introduced to model the word boundaries. It is similar to the short pause "sp" in ASR, but "wb" is forced to be inserted between every two adjacent words in the reference when training the LSTM-CTC and also inserted into keywords when doing KWS. For example, keyword "NINETY NINE" becomes "wb NINETY wb NINE wb". When doing hypothesis search, an additional rule is adopted, i.e. for all i which satisfies $op_i \in I$ and $e_i = wb$, the probabilities of op_i and $P(e_i)$ are set to one. This extra rule makes the insertion operations of "wb" allowable and has no influence on probability calculations. The probabilities of substitution operation and deletion operation of "wb" are still computed through confusion matrix. The purposes of adding "wb" are that (a) increasing the length of short keywords avoids its phone sequence being a substring of a longer word, (b) the insertion operations of "wb" having no influence on probability calculations help the search algorithm to stride across those misidentified spikes.

3. Experiments

3.1. Experiment Setup

3.1.1. Data

A speaker-independent 5k vocabulary dataset of the Wall Street Journal (WSJ0) corpus [26] is used to train and evaluate the proposed LSTM-CTC KWS method. Words or phrases which satisfy the following conditions are selected randomly as the keywords. (i) The keywords must appear at least 5 times in the reference, that guarantees there are enough positive examples to evaluate performance and reduce occasionality. (ii) The length of the phone sequences of the keywords is between 3 and 12. In addition, some similar words or phrases are added so that the challenge is greater. In total, 50 keywords are used.

3.1.2. Evaluation Metrics

The evaluation metrics that we report are Equal Error Rate (EER) and Figure of Merit (FOM). EER reflects the average equal error rate at which the false alarm rate is equal to the false rejection rate. The FOM is defined as the average Detection Rate (DR) in the range of False Alarms (FA) from 0 to 10 . DR and FA can be computed as,

$$DR = \frac{N_{TP}}{N_{TP} + N_{FN}}, \quad FA = \frac{N_{FP}}{Dur \times N_{kw}} \tag{10}$$

where N_{TP}, N_{FN} and N_{FP} are the number of true positives, false negatives and false positives respectively. Dur is the total duration of all test speech, and N_{kw} is the number of keywords. For EER values, smaller is better. For FOM values,

larger is better. Both EER and FOM are obtained by sweeping the fixed threshold or the constant scaling factor C for LSTM-CTC KWS.

3.1.3. Configuration

In this paper, the speech is analyzed using a 25ms Hamming window with a 10ms fixed frame rate. Fourier-transform-based log filter-bank features with 24 coefficients as well as first and second derivatives are extracted using mean normalisation. Here, an alternative architecture to standard LSTM is adopted, that is the so-called Long Short-Term Memory Projected (LSTMP) architecture [27]. In LSTMP, an extra linear recurrent projection layer connects from the output of the LSTM and connects to the input of the multiplicative gates. By increasing the number of memory cells and adjusting the size of projection layers, LSTMP is an effective approach to control the number of parameters of LSTM without performance deterioration. The LSTM input layer has a size of 72 corresponding to an acoustic feature vector and the output layer has 69 softmax units corresponding to a set of phones of a CMU pronouncing dictionary, and plus one softmax unit for *blank* label, and one optional softmax unit for "wb" if word boundaries are modelled. The LSTM is composed of 2 hidden layers, and the number of memory blocks in each hidden layer is 384 with one memory cell per block, and the size of the linear projection layer in each hidden layer is 128. The total number of parameters in the LSTM is about 813K. The outputs are delayed by 5 frames since the information from the future frames may help making better decisions for the current frame.

The LSTM network is initiated by cross-entropy criteria and sequentially trained using CTC. The training algorithm is stochastic gradient descent with a learning rate of 0.00006 and a momentum of 0.9. Besides, $h_{spike} = 0.2$ and $h_{node} = 0.005$.

3.1.4. Baseline systems

For performance comparison, conventional Keyword-Filler Hidden Markov Models are trained and evaluated on the same data set. The keyword models estimate the likelihood of the observed feature vector sequences, and the filler model is used to represent all non-keyword speech. Through Viterbi decoding, the best path is found and if the path passes a keyword model then the corresponding keyword is determined as detected. The Keyword-Filler HMM topology is identical to the one used in [15]. The EER and FOM are obtained by sweeping the transition probabilities between keyword and filler models. In this paper, cross-word triphones are used and HMM states are clustered by decision trees that leads to 1689 HMM states remained. A GMM with 40 Gaussian mixtures and two DNNs of different structures are used to compute the HMM state densities in different experiments and act as baselines. For better performance, the acoustic features for the GMM is cepstral mean normalized MFCC coefficients 1 to 12, log energy, as well as first and second order delta coefficients. The acoustic features mentioned in section 3.1.3 are still used for DNNs and LSTM. Besides, one DNN has 4 hidden layers with 512 nodes per layer and a larger DNN has 7 hidden layers with 512 nodes per layer. Both DNNs take an 11-frame context window with 5 left frames and 5 right frames as input and compute activations using a sigmoid function and output the posteriors of 1689 HMM states.

3.2. Performance Comparison

The performances of different KWS systems are shown in Table 1. Here, both modelling word boundary and threshold estimation approaches are used in the LSTM-CTC KWS system. The results indicate that LSTM-CTC KWS achieves significant gains over the Keyword-Filler Hidden Markov Models (relative 29% EER reduction, and 12% FOM increase), demonstrating the effectiveness of the proposed method. It can be observed that the larger DNN-HMM does not achieve more improvements than the smaller DNN-HMM, we conjecture that the bottleneck of performance may not be DNN but the method itself, since the Keyword-Filler HMM approach may be vulnerable to distinguish similar keywords. Besides, the LSTM-CTC model also has less parameters than the baselines.

Table 1: *Keyword-Filler HMMs vs. LSTM-CTC KWS system.*

	EER	FOM	Parameters
GMM-HMM (40mix)	6.4	71.8	5.4M
DNN-HMM (4x512)	5.1	75.5	2.0M
DNN-HMM (7x512)	5.1	76.0	2.8M
LSTM-CTC KWS	3.6	85.2	813K

3.3. Effect of Modelling Word Boundary

Table 2: *Performances of LSTM-CTC KWS with or without modelling word boundary.*

Keyword Length	wb	EER	FOM
short	$\times$	9.0	76.9
short	$\surd$	4.5	87.3
long	$\times$	3.1	92.0
long	$\surd$	1.8	97.2

Those keywords which contain less than 6 phones are considered as short keywords, otherwise the keywords are long, thereby, two keyword sets are tested. Table 2 illustrates the effect of modelling word boundary in LSTM-CTC KWS systems. Both short keywords and long keywords yield performance improvements, 50% EER reduction for short keywords and 42% EER reduction for long keywords. Besides, the performances of long keywords consistently beat short keywords. This is because the phone sequences of short keywords are more likely being substrings of other words or phrases and it will cause a higher false alarm rate.

3.4. Effect of Threshold Estimation

Table 3: *Performances of LSTM-CTC KWS with different threshold policies.*

	EER	FOM
Fixed threshold	4.0	81.4
Estimated on dev set	3.6	85.2

Table 3 indicates the performances of LSTM-CTC KWS using a fixed preset threshold for all keywords or using the threshold estimation approach mentioned in section 2.3 to calculate various thresholds for different keywords. The utterances of the development set of WSJ0 are used to estimated thresholds. As expected, both EER and FOM become better with the estimated thresholds.

4. Conclusions

In this paper, a novel KWS approach is proposed with LSTM-CTC, which is utilized to identify phones by exploiting long-range context information and generate a lattice, then the minimum edit distance based post-processing algorithm is implemented on the phone lattice. Experiments on a WSJ0 dataset showed that the proposed system prevailed over the traditional HMM based approaches. Future work will look into combination of the LSTM-CTC approach and Keyword-Filler models.

5. References

[1] J. Schalkwyk, D. Beeferman, F. Beaufays, B. Byrne, C. Chelba, M. Cohen, M. Kamvar, and B. Strope, "Your Word is my Command: Google Search by Voice: A Case Study," in *Advances in Speech Recognition*. Springer, 2010, pp. 61–90.

[2] D. Vergyri, I. Shafran, A. Stolcke, V. R. R. Gadde, M. Akbacak, B. Roark, and W. Wang, "The SRI/OGI 2006 spoken term detection system." in *INTERSPEECH*. Citeseer, 2007, pp. 2393–2396.

[3] D. R. 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 *Eighth Annual Conference of the International Speech Communication Association*, 2007.

[4] M. Weintraub, "LVCSR log-likelihood ratio scoring for keyword spotting," in *Acoustics, Speech, and Signal Processing, 1995. ICASSP-95., 1995 International Conference on*, vol. 1. IEEE, 1995, pp. 297–300.

[5] A. Mandal, K. P. Kumar, and P. Mitra, "Recent developments in spoken term detection: a survey," *International Journal of Speech Technology*, vol. 17, no. 2, pp. 183–198, 2014.

[6] R. C. Rose and D. B. Paul, "A hidden Markov model based keyword recognition system," in *Acoustics, Speech, and Signal Processing, 1990. ICASSP-90., 1990 International Conference on*. IEEE, 1990, pp. 129–132.

[7] J. G. Wilpon, L. R. Rabiner, C.-H. Lee, and E. Goldman, "Automatic recognition of keywords in unconstrained speech using hidden Markov models," *Acoustics, Speech and Signal Processing, IEEE Transactions on*, vol. 38, no. 11, pp. 1870–1878, 1990.

[8] J. Wilpon, L. Miller, and P. Modi, "Improvements and applications for key word recognition using hidden Markov modeling techniques," in *Acoustics, Speech, and Signal Processing, 1991. ICASSP-91., 1991 International Conference on*. IEEE, 1991, pp. 309–312.

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

[10] M. Barakat, C. H. Ritz, and D. A. Stirling, "An improved template-based approach to keyword spotting applied to the spoken content of user generated video blogs," in *Multimedia and Expo (ICME), 2012 IEEE International Conference on*. IEEE, 2012, pp. 723–728.

[11] G. Chen, C. Parada, and T. N. Sainath, "Query-by-example keyword spotting using long short-term memory networks," in *Acoustics, Speech and Signal Processing (ICASSP), 2015 IEEE International Conference on*. IEEE, 2015, pp. 5236–5240.

[12] K. Thambiratnam and S. Sridharan, "Dynamic Match Phone-Lattice Searches For Very Fast And Accurate Unrestricted Vocabulary Keyword Spotting." in *ICASSP (1)*, 2005, pp. 465–468.

[13] P. S. Cardillo, M. Clements, and M. S. Miller, "Phonetic searching vs. LVCSR: How to find what you really want in audio archives," *International Journal of Speech Technology*, vol. 5, no. 1, pp. 9–22, 2002.

[14] T. Sainath and C. Parada, "Convolutional neural networks for small-footprint keyword spotting," in *Proc. Interspeech*, 2015.

[15] G. Chen, C. Parada, and G. Heigold, "Small-footprint keyword spotting using deep neural networks," in *Acoustics, Speech and Signal Processing (ICASSP), 2014 IEEE International Conference on*. IEEE, 2014, pp. 4087–4091.

[16] R. Prabhavalkar, R. Alvarez, C. Parada, P. Nakkiran, and T. N. Sainath, "Automatic gain control and multi-style training for robust small-footprint keyword spotting with deep neural networks," in *Acoustics, Speech and Signal Processing (ICASSP), 2015 IEEE International Conference on*. IEEE, 2015, pp. 4704–4708.

[17] A. Graves, A.-r. Mohamed, and G. Hinton, "Speech recognition with deep recurrent neural networks," in *Acoustics, Speech and Signal Processing (ICASSP), 2013 IEEE International Conference on*. IEEE, 2013, pp. 6645–6649.

[18] A. Graves, N. Jaitly, and A.-r. Mohamed, "Hybrid speech recognition with deep bidirectional LSTM," in *Automatic Speech Recognition and Understanding (ASRU), 2013 IEEE Workshop on*. IEEE, 2013, pp. 273–278.

[19] A. Graves, S. Fernández, F. Gomez, and J. Schmidhuber, "Connectionist temporal classification: labelling unsegmented sequence data with recurrent neural networks," in *Proceedings of the 23rd international conference on Machine learning*. ACM, 2006, pp. 369–376.

[20] A. Graves and N. Jaitly, "Towards end-to-end speech recognition with recurrent neural networks," in *Proceedings of the 31st International Conference on Machine Learning (ICML-14)*, 2014, pp. 1764–1772.

[21] Z. Chen, W. Deng, T. Xu, and K. Yu, "Phone Synchronous Decoding with CTC Lattice," in *INTERSPEECH*, 2016.

[22] S. Fernández, A. Graves, and J. Schmidhuber, "An application of recurrent neural networks to discriminative keyword spotting," in *Artificial Neural Networks–ICANN 2007*. Springer, 2007, pp. 220–229.

[23] M. Wöllmer, F. Eyben, B. Schuller, and G. Rigoll, "Spoken term detection with connectionist temporal classification: a novel hybrid ctc-dbn decoder," in *Acoustics Speech and Signal Processing (ICASSP), 2010 IEEE International Conference on*. IEEE, 2010, pp. 5274–5277.

[24] E. S. Ristad and P. N. Yianilos, "Learning string-edit distance," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 20, no. 5, pp. 522–532, 1998.

[25] U. V. Chaudhari and M. Picheny, "Improvements in phone based audio search via constrained match with high order confusion estimates," in *Automatic Speech Recognition & Understanding, 2007. ASRU. IEEE Workshop on*. IEEE, 2007, pp. 665–670.

[26] J. Garofalo, D. Graff, D. Paul, and D. Pallett, "Continous Speech Recognition (CSR-I) Wall Street Journal (WSJ0) news, complete," *Linguistic data consortium, philadelphia*, 1993.

[27] H. Sak, A. W. Senior, and F. Beaufays, "Long short-term memory recurrent neural network architectures for large scale acoustic modeling." in *INTERSPEECH*, 2014, pp. 338–342.