Interspeech 2018
2-6 September 2018, Hyderabad, India

Compact Feedforward Sequential Memory Networks for Small-footprint Keyword Spotting

Mengzhe Chen, Shiliang Zhang, Ming Lei, Yong Liu, Haitao Yao, Jie Gao

Alibaba Group, P. R. China

{mengzhe.cmz, sly.zsl, lm86501, shoudu.ly, timmy.yht, haozhi.gj}@alibaba-inc.com

Abstract

Due to limited resource on devices and complicated scenarios, a compact model with high precision, low computational cost and latency is expected for small-footprint keyword spotting tasks. To fulfill these requirements, in this paper, compact Feedforward Sequential Memory Network (cFSMN) which combines low-rank matrix factorization with conventional FSMN is investigated for a far-field keyword spotting task. The effect of its architecture parameters is analyzed. Towards achieving lower computational cost, multiframe prediction (MFP) is applied to cFSMN. For enhancing the modeling capacity, an advanced MFP is attempted by inserting small DNN layers before output layers. The performance is measured by area under the curve (AUC) for detection error tradeoff (DET) curves. The experiments show that compared with a well-tuned long short-term memory (LSTM) which needs the same latency and twofold computational cost, the cFSMN achieves 18.11% and 29.21% AUC relative decreases on the test sets which are recorded in quiet and noisy environment respectively. After applying advanced MFP, the system gets 0.48% and 20.04% AUC relative decrease over conventional cFSMN on the quiet and noisy test sets respectively, while the computational cost relatively reduces 46.58%.

Index Terms: keyword spotting, compact feedforward sequential memory network, multiframe prediction, small-footprint

1. Introduction

Keyword spotting is a task of detecting pre-defined words in audio stream. Large vocabulary continuous speech recognition (LVCSR) based method is a traditional solution for this task. The audio is input to the system and keywords are searched in the resulting lattice[1, 2, 3]. With rapid development of voice assistant systems, small-footprint keyword spotting becomes a hot issue. These systems work on resource constrained devices and listen continuously for specific keywords to convert the status of devices. Thus, besides the metrics for general keyword spotting systems, low computational cost is also an important indicator. Obviously, with LVCSR decoding, the approach mentioned above cannot fulfill it.

A widely used approach for small-footprint keyword spotting is building hidden Markov models (HMMs) to model both the keyword and the background audio signals [4, 5, 6]. The decoding graph is built by the paths for each keyword and for non-keyword audio. Viterbi-based method is applied to search for the keywords in the decoding graph. Gaussian mixture model (GMM) is a traditional one for modeling observation probabilities of HMM. With deep neural network (DNN) gradually takes the place of GMM in speech recognition, DNN and other neural network based models are also attempted in this structure [7, 8]. In recent years, the systems that only use neural network without HMM involved are proposed [9, 10]. Instead of a searching process, the frame-wise posteriors are smoothed and directly used to predict the keywords by comparing the posteriors with a pre-defined threshold.

Acoustic model (AM) is crucial in small-footprint keyword spotting systems. It is expected to be powerful enough to fulfill the high demands. The first demand is obtaining a high recall at a low false alarm rate to guarantee basic user experience. The second is low computational cost, due to the limited resource on device and energy-saving requirements for the devices using batteries. Low latency is also a demand, since responses from the device without obvious delays are required. Meanwhile, the model should have a stable performance for complicated scenarios, like far-field, strong noise and reverberation etc. Due to the superior performances, deep learning based approaches are applied into the system. DNN [9] is a widely used model in this task. Convolutional neural network (CNN) are also explored [11, 12]. In DNN and CNN based systems, the context information is introduced into the network by stacking frames as input. Some systems are built with recurrent neural networks (RNNs) which are capable of modeling long temporal contexts [13, 14]. Long short-term memory (LSTM) is the most popular one among the various RNNs, since it solves vanishing gradient problem [15, 16]. Time delay neural network (TDNN) is a feedforward architecture which also can model long temporal contexts [17, 18]. [19] combines the strengths of DNN, CNN and RNN, and builds an architecture called convolutional recurrent neural network (CRNN).

Feedforward Sequential Memory Networks (FSMN) is designed to model long-term dependency without using recurrent feedback [20]. Equipped with learnable memory blocks, FSMN can record history and future information with limited computational cost. Thus, FSMN is suitable for small-footprint keyword spotting tasks. Due to the critical demand of our products, we are interested in exploring the models requiring lower computational cost without performance decline. Thus, compact FSMN (cFSMN) combined with advanced multiframe prediction (MFP) is adopted. cFSMN is an improved version which combines low-rank matrix factorization with conventional FSMN [21]. The model trained with MFP predicts multiple frames with one frame input [22]. An advanced version is adding small DNNs before output layers, which could improve the model capacity. The architecture of cFSMN+MFP has been successfully shipped on our various voice assistant products. In the experiments, DNN and LSTM are compared with cFSMN. DNN is a commonly-used model with low computational cost but limited modeling capacity. LSTM has better modeling capacity with an increase on computational cost. Our proposed structure has an adavantage on computational cost like DNN and competitive performance like LSTM at the same time.

The paper is organized as follows: Section 2 introduces our keyword spotting system, the architecture of cFSMN and the principle of multiframe prediction. Section 3 presents the ex-

10.21437/Interspeech.2018-1204

perimental setup and results. Conclusion is given in Section 4.

2. Keyword Spotting System

The systems without HMM involved always use word as modeling unit, so that the model is specified for the pre-defined keywords. However, in our products, besides the main keyword, the system should have the flexibility of adding new keywords. Thus, our system follows the conventional keyword/background HMM structure and uses senones (tied cdphone states) as modeling unit. The decoding graph consists of keyword and background paths. Each keyword path consists of a sequence of HMMs for one keyword. Adding a keyword needs adding a keyword path into the graph. Background paths are built for non-keyword speech, noise and silence. Viterbi searching in the decoding graph runs separately on the competing keyword and background paths through token passing. Once an active token reaches the end of a keyword path, the acoustic information of the hypothesized segment will be extracted. The system triggers when the regularized ratio of the keyword and background path scores exceeds a pre-set threshold.

During the searching, the score for each frame is predicted by an AM. The inputs of the AM are acoustic features, and the outputs are posterior distribution over the HMM states of keyword and background models. No language models are used.

2.1. Compact Feedforward Sequential Memory Networks

FSMN is a fully-connected feedforward neural network equipped with learnable memory blocks in hidden layers [23]. Inspired by the design of finite impulse response (FIR) filters, the memory block adopts a tapped-delay line structure to encode long context information into a fixed-size representation.

Considering the additional parameters introduced by the memory blocks, a variant FSMN architecture namely cFSMN is proposed [21]. cFSMN adopts low-rank matrix factorization to standard FSMN, thereby reducing computational cost without performance decline. Fig.1 gives an illustration of cFSMN. For each cFSMN layer, a linear projection layer is firstly applied. This layer makes cFSMN different from FSMN. It is smaller than hidden layer. Then, its output is fed to the memory block to form an element-wise weighted sum of current frame and its context. Finally, the sum is fed to next cFSMN layer through an affine transform and a nonlinearity.

Each cFSMN layer is calculated according to Eq.1,2,3. $\mathbf{h}_t^\ell$ and $\mathbf{h}_t^{\ell+1}$ denote lth and $l+1$th cFSMN hidden layer respectively at time t. $\mathbf{p}_t^\ell$ is the projection of the lth cFSMN hidden layer as shown in Eq.1. The memory block $\tilde{\mathbf{p}}_t^\ell$ is encoded by the context information of $\mathbf{p}_t^\ell$ as shown in Eq.2. Here, $\odot$ denotes element-wise multiplication of two vectors with same length. $N1$ and $N2$ denotes lookback and lookahead order respectively. The latency of cFSMN is related to the lookahead order in each memory block. $\mathbf{a}_i^\ell$ and $\mathbf{c}_i^\ell$ are the encoding coefficients of the memory block. The output of cFSMN hidden layer is calculated as Eq.3.

$$\mathbf{p}_t^\ell = \mathbf{W}^\ell \mathbf{h}_t^\ell + \mathbf{b}^\ell \tag{1}$$

$$\tilde{\mathbf{p}}_t^\ell = \mathbf{p}_t^\ell + \sum_{i=0}^{N_1} \mathbf{a}_i^\ell \odot \mathbf{p}_{t-i}^\ell + \sum_{j=1}^{N_2} \mathbf{c}_j^\ell \odot \mathbf{p}_{t+j}^\ell \tag{2}$$

$$\mathbf{h}_t^{\ell+1} = f(\mathbf{U}^\ell \tilde{\mathbf{p}}_t^\ell + \tilde{\mathbf{b}}^\ell). \tag{3}$$

With the memory blocks, cFSMN is able to capture long-term information of sequences. Since there is no recurrent cycles in the network, its computational cost is much less than

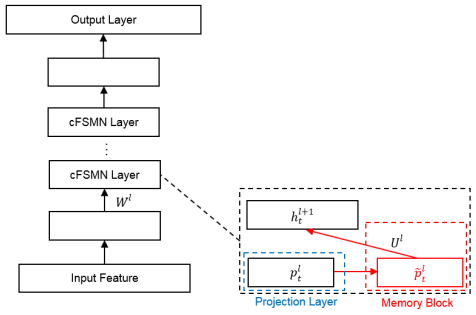

Figure 1: *Illustration of cFSMN*

LSTM. By increasing the order, cFSMN can record longer context with limited computational cost increase. Thus, compared with TDNNs whose computational cost is linear growth to the input context size, FSMN could make use of longer context with lower computational cost [24]. In addition, like other feedforward models, cFSMN can be trained more efficiently and reliably with backpropagation (BP) than recurrent network which is trained with back-propagation through time (BPTT).

2.2. Multiframe Prediction

Conventionally, each input of neural network acoustic models is used to predict the posterior distribution of current frame, as shown in Fig.2(a). To take advantage of time correlations between feature frames, [22] proposes multiframe prediction (MFP) approach. As shown in Fig.2(b), the network shares all hidden layers, and uses multiple output layers for multiple frames. In this way, each input frame fed into the network can get the predictions of K consecutive frames. As a result, it significantly speed up the training and decoding procedures. The experimental results on DNNs in speech recognition show that this method achieves comparable performance to the standard model, while achieving up to a 4X reduction in the computational cost of the neural network activations [22].

For complicated tasks, independent output layers may not be capable of modeling the shared representation of consecutive frames. For achieving more stable performances, we propose an advanced MFP architecture. As shown in Fig.2(c), some small hidden layers are added before each independent output layer for improving the modeling capacity.

3. Experiments and Results

3.1. Database

Our keyword spotting system are evaluated on a far-field television assistant task. In the experiments, the keyword is a four-syllable Mandarin word. The training data consists of two sets. One is a 24K-hour simulation set, and the details of simulation method refer to [25]. The other one is a 40-hour keyword-specific set recorded in real environment.

Two real-recorded test sets are used as positive examples to evaluate the recall rate. They are 4K utterances recorded in quiet environment and 8K utterances recorded in noisy environment. In the noisy environment, two computers make sounds of music, news or talk shows beside speakers. Meanwhile, we prepare 600-hour negative examples for evaluating the false alarm rate. These sentences are from various sources, like videos, music and news broadcasting etc. Finally, a half size non-overlapping set of positive and negative is used as development set for tuning

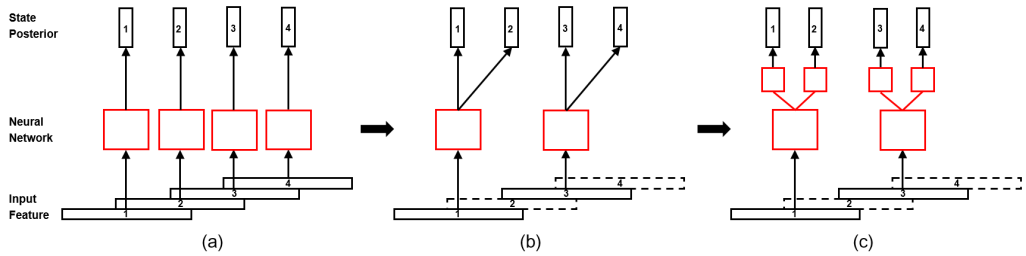

Figure 2: (a) *The conventional prediction approach,* (b) *The MFP approach, and* (c) *The advanced MFP approach.*

the decoding parameters and model architectures.

3.2. Model Training

In our system, one-state HMMs are used to model 917 senones. The input feature are 80-dimensional log-mel filterbank energies using 25ms Hamming window with 10ms shift. Since Lower Frame Rate (LFR) [26] is adopted, the time step is 30ms instead of 10ms.

All models are trained in a distributed manner using BMUF [27] optimization on 16 GPUs and frame-level cross entropy criterion. The cFSMN and DNN models are trained using standard BP with stochastic gradient descent (SGD), and LSTM is trained with BPTT. All the networks are randomly initialized by the Glorot-Bengio strategy described in [28], trained with simulation data and further finetuned with keyword-specific data.

3.3. Evaluation

The system is evaluated by several metrics: i. false reject, denotes the system is not triggered when the keyword is spoken. It equals to one minus recall rate; ii. false alarm, denotes the system is triggered when no keywords are spoken. Detection error tradeoff (DET) curves whose x-axis labels false alarm and y-axis labels false reject are used to exhibit the overall performance. Better models has lower curves. Due to confidentiality, instead of absolute numbers of false alarm, the numbers on x-axis are multiplied by a constant. For comparing different models, we compute area under the curve (AUC) of DET curves, and smaller AUC indicates better performance. iii. computational cost, which is evaluated by floating-point operations per second (FLOPS); iv. latency, which can be calculated according to the model architecture.

3.4. Impact of the model architecture

Limited by the computational resource, we firstly build a temporary baseline cFSMN which has four cFSMN hidden layers and two low-rank layers inserted after the input layer and before the output layer. Consecutive frames within a context window of 17(8+1+8) are stacked to produce the 1360-dimension input. The cFSMN layer has 250 nodes with a 128-node projection layer. $N1$ and $N2$ are 5 and 1, which means the network could lookback 5($N1$)*4(number of hidden layers)*30ms(LFR) and lookahead 1($N2$)*4(number of hidden layers)*30ms(LFR). Note that the latency consists of the lookahead part and the right context of input, so the total latency is 200ms.

We observe the impact of different architectures by changing the context of input feature, lookback order ($N1$), lookahead order ($N2$) and number of cFSMN hidden layers. Table 1 gives a summary of performances of all the architectures. No.1 is the result of baseline model. Note that the order follows the format of $a * b$. a denotes $N1/N2$ and b denotes the number of layers that have this order. AUC relative change over the baseline, FLOPS and latency are given in the table.

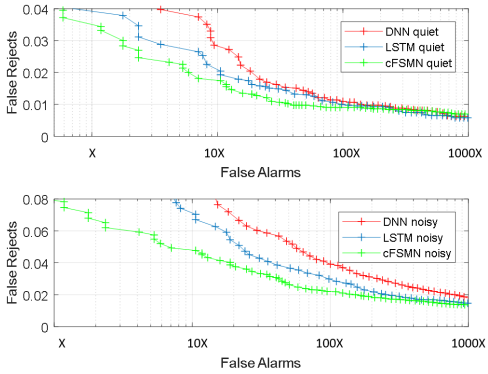

Figure 3: *DET curves of different models*

No.1 to No.4 is a group of experiments to test the impact of changing the context of input feature. The table shows that the AUCs of them are similar. Since longer context brings more computational cost and latency, shorter context would be selected. The change of lookback order is tested from No.5 to No.7 and No.4. We find that increasing the lookback order within limits could bring an improvement, while making it too long will do a harm. The reason is that the average duration of four-syllable keyword is less than 900ms, and looking back too much is not necessary and even has negative effect, like the No.7 which lookbacks 1200ms. The lookahead order will bring latency to the system, so $N2$ would set to be smaller than $N1$. The results from No.8 to No.10 and No.4 give the trend that increasing lookahead order brings improvement, while the latency also increases rapidly. The last group shows the results for different number of cFSMN layers. With more cFSMN layers, the AUCs get smaller, especially when the number increases from 3 to 4. After 4, the improvements are limited if taking the FLOPS and latency into consideration.

Conclusions can be made that using short context for input feature can reduce the computational cost effectively without obvious performance decrease, and increasing the lookback, lookahead order and the number of cFSMN layers within limits have obvious positive impacts on AUC performance with acceptable FLOPS and latency.

3.5. System performance

Allowing for metrics listed in Table 1 and the implementation of our decoder, the No.4 model is selected for further explo-

Table 1: *Summary of cFSMNs with different architectures*

ID	input	$N1$	$N2$	layer	AUC relative change		FLOPS (M)	latency (ms)
					quiet	noisy		
1	8+1+8	5*4	1*4	4	-	-	65.00	200
2	8+1+5	5*4	1*4	4	+0.32%	-1.70%	61.64	170
3	8+1+2	5*4	1*4	4	-0.54%	-3.40%	59.40	140
4	2+1+2	5*4	1*4	4	-0.11%	-2.76%	51.56	140
5	2+1+2	3*4	1*4	4	+1.46%	+8.93%	51.46	140
6	2+1+2	7*4	1*4	4	-4.48%	-11.44%	51.72	140
7	2+1+2	10*4	1*4	4	+23.36%	+3.58%	51.82	140
8	2+1+2	5*4	1*2+0*2	4	+3.31%	+6.70%	51.36	80
9	2+1+2	5*4	2*4	4	-9.77%	-19.33%	51.61	260
10	2+1+2	5*4	3*4	4	-9.82%	-19.55%	51.66	380
11	2+1+2	5*3	1*3	3	+25.52%	+12.28%	45.06	110
12	2+1+2	5*5	1*5	5	-3.51%	-6.62%	58.06	170
13	2+1+2	5*6	1*6	6	-9.09%	-8.58%	64.57	200

Table 2: *Performance of various AMs*

model	AUC relative change		FLOPS (M)	latency (ms)
	quiet	noisy		
LSTM	-	-	105.87	140
DNN	+42.85%	+42.76%	53.10	20
cFSMN	-18.11%	-29.21%	51.56	140

Table 3: *Performance of MFP training*

model	AUC relative change		FLOPS (M)	latency (ms)
	quiet	noisy		
No.9	-	-	51.61	260
MFP2	+19.09%	-11.04%	32.24	260
MFP3	+68.77%	+3.54%	25.55	380
MFP4	+86.06%	+39.27%	22.59	500
advanced MFP2	-0.48%	-20.04%	27.57	260
advanced MFP3	+40.27%	-10.73%	20.98	380
advanced MFP4	+68.06%	+32.55%	18.00	500

ration. According to the FLOPS of cFSMN, we explore various topologies of DNN and LSTM, and select the best ones for the comparison. The DNN is built by four fully-connected layers with 256 nodes and a low-rank layer for maintaining a similar FLOPS as cFSMN. With the same number of layers and hidden nodes, the computational cost of LSTM will be several times more than DNN, so we have to reduce the number of nodes to 190 and layers to 3 for getting a comparable FLOPS (twofold FLOPS of cFSMN). In addition, cFSMN has a latency from its lookahead strategy. For feeding the same information to LSTM, the model is trained with a four-frame target delay which follows the method in LVCSR [29]. All the models are well-tuned on the development set.

The performances of these models are shown in Fig.3. The two figures are results of the two test sets recorded in quiet and noisy environments respectively. According to the demand of this task, we focus on the false alarm ranging from 10X to 100X. During this interval, on both test sets, the performance of cFSMN (green) is definitely better than that of LSTM (blue), and LSTM is better than DNN (red). The details of comparison are exhibited in Table 2. Taking the LFR into consideration, the latencies of LSTM and cFSMN are both 140ms which is acceptable in our task. The FLOPSs of DNN and cFSMN are similar, while LSTM needs a double computational cost. According to AUC, the DNN is not competitive, and the cFSMN gets 18.11% relative AUC decrease in quiet environment and 29.21% in noisy environment compared with the LSTM.

3.6. Training with Multiframe Prediction

For training the model with MFP, the No.4 model in Table 1 is used as the basic model. Following the topology of standard MFP, the output layer of the basic model is replaced with multiple output layers which are fully connected to the last hidden layer. In the advanced version, small Rectified Linear Units (ReLU) DNN hidden layers are inserted before each independent output layer. In our experiments, the DNN consists of one 125-nodes layers and one 80-nodes layers. It seems that the advanced version adds more parts into the model than the standard version. In fact, since the last layer before the output is designed to be small, FLOPSs of the advanced ones are smaller than standard ones. These new-created models are finetuned with keyword data.

The standard and advanced versions of MFP models are trained for comparison. Meanwhile, different number of frames that predicted for each input frame are attempted, and the number is refer to as K. Table 3 exhibits the performances. Note that MFP will bring K times of lookahead latency due to our design of cFSMN with MFP. Since we use No.4 model as the basic model, the No.9 model in Table 1 whose $N2$ is twice as that of No.4 model is used for comparing.

Compared with standard MFP, the models trained with advanced MFP achieve better performances. It verifies that the small DNN layers help to enhance the modeling capacity. Compared with the No.9 model, when $K = 2$, the model trained with advanced MFP gets 0.48% and 20.04% AUC relative decreases on the quiet and noisy test sets respectively, meanwhile the FLOPS reduces 46.58% relatively. However, if K increases further, the system suffers from a serious performance decline. We believe lower FLOPS without performace decline could be obtained through tuning the architecture and training method of the small DNNs further.

4. Conclusions

In the paper, we present our work on building a cFSMN-HMM small-footprint keyword spotting system. We analyze the parameters of cFSMN architecture and compare it with other competitive models. To reduce the computational cost further, the advanced MFP is successfully applied into our system. The experiments show that cFSMN have advantages on AUC and FLOPS compared with LSTM and DNN. Combined with MFP, cFSMN could obtain lower FLOPS without performance decline.

5. References

[1] 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.

[2] J. Mamou, B. Ramabhadran, and O. Siohan, "Vocabulary independent spoken term detection," in *Proceedings of the 30th annual international ACM SIGIR conference on Research and development in information retrieval*. ACM, 2007, pp. 615–622.

[3] D. Vergyri, I. Shafran, A. Stolcke, R. R. Gadde, M. Akbacak, B. Roark, and W. Wang, "The sri/ogi 2006 spoken term detection system," in *Eighth Annual Conference of the International Speech Communication Association*, 2007.

[4] J. R. Rohlicek, W. Russell, S. Roukos, and H. Gish, "Continuous hidden markov modeling for speaker-independent word spotting," in *Acoustics, Speech, and Signal Processing, 1989. ICASSP-89., 1989 International Conference on*. IEEE, 1989, pp. 627–630.

[5] 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.

[6] 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.

[7] I.-F. Chen and C.-H. Lee, "A hybrid hmm/dnn approach to keyword spotting of short words." in *INTERSPEECH*, 2013, pp. 1574–1578.

[8] M. Sun, V. Nagaraja, B. Hoffmeister, and S. Vitaladevuni, "Model shrinking for embedded keyword spotting," in *Machine Learning and Applications (ICMLA), 2015 IEEE 14th International Conference on*. IEEE, 2015, pp. 369–374.

[9] 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.

[10] G. Tucker, M. Wu, M. Sun, S. Panchapagesan, G. Fu, and S. Vitaladevuni, "Model compression applied to small-footprint keyword spotting." in *INTERSPEECH*, 2016, pp. 1878–1882.

[11] T. N. Sainath and C. Parada, "Convolutional neural networks for small-footprint keyword spotting," in *Sixteenth Annual Conference of the International Speech Communication Association*, 2015.

[12] Y. Wang, P. Getreuer, T. Hughes, R. F. Lyon, and R. A. Saurous, "Trainable frontend for robust and far-field keyword spotting," in *Acoustics, Speech and Signal Processing (ICASSP), 2017 IEEE International Conference on*. IEEE, 2017, pp. 5670–5674.

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

[14] P. Baljekar, J. F. Lehman, and R. Singh, "Online word-spotting in continuous speech with recurrent neural networks," in *Spoken Language Technology Workshop (SLT), 2014 IEEE*. IEEE, 2014, pp. 536–541.

[15] M. Woellmer, B. Schuller, and G. Rigoll, "Keyword spotting exploiting long short-term memory," *Speech Communication*, vol. 55, no. 2, pp. 252–265, 2013.

[16] M. Sun, A. Raju, G. Tucker, S. Panchapagesan, G. Fu, A. Mandal, S. Matsoukas, N. Strom, and S. Vitaladevuni, "Max-pooling loss training of long short-term memory networks for small-footprint keyword spotting," in *Spoken Language Technology Workshop (SLT), 2016 IEEE*. IEEE, 2016, pp. 474–480.

[17] V. Peddinti, D. Povey, and S. Khudanpur, "A time delay neural network architecture for efficient modeling of long temporal contexts," in *Sixteenth Annual Conference of the International Speech Communication Association*, 2015.

[18] M. Sun, D. Snyder, Y. Gao, V. Nagaraja, M. Rodehorst, N. S. Panchapagesan, S. Matsoukas, and S. Vitaladevuni, "Compressed time delay neural network for small-footprint keyword spotting," *Proc. Interspeech 2017*, pp. 3607–3611, 2017.

[19] S. Ö. Arik, M. Kliegl, R. Child, J. Hestness, A. Gibiansky, C. Fougner, R. Prenger, and A. Coates, "Convolutional recurrent neural networks for small-footprint keyword spotting," *arXiv preprint arXiv:1703.05390*, 2017.

[20] S. Zhang, C. Liu, H. Jiang, S. Wei, L. Dai, and Y. Hu, "Feedforward sequential memory networks: A new structure to learn long-term dependency," *arXiv preprint arXiv:1512.08301*, 2015.

[21] S. Zhang, H. Jiang, S. Xiong, S. Wei, and L.-R. Dai, "Compact feedforward sequential memory networks for large vocabulary continuous speech recognition." in *INTERSPEECH*, 2016, pp. 3389–3393.

[22] V. Vanhoucke, M. Devin, and G. Heigold, "Multiframe deep neural networks for acoustic modeling," in *Acoustics, Speech and Signal Processing (ICASSP), 2013 IEEE International Conference on*. IEEE, 2013, pp. 7582–7585.

[23] S. Zhang, H. Jiang, S. Wei, and L. Dai, "Feedforward sequential memory neural networks without recurrent feedback," *arXiv preprint arXiv:1510.02693*, 2015.

[24] S. Zhang, C. Liu, H. Jiang, S. Wei, L. Dai, and Y. Hu, "Nonrecurrent neural structure for long-term dependence," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 25, no. 4, pp. 871–884, 2017.

[25] C. Kim, A. Misra, K. Chin, T. Hughes, A. Narayanan, T. Sainath, and M. Bacchiani, "Generation of large-scale simulated utterances in virtual rooms to train deep-neural networks for far-field speech recognition in google home," *Proc. INTERSPEECH. ISCA*, 2017.

[26] G. Pundak and T. N. Sainath, "Lower frame rate neural network acoustic models," in *Interspeech*, 2016, pp. 22–26.

[27] K. Chen and Q. Huo, "Scalable training of deep learning machines by incremental block training with intra-block parallel optimization and blockwise model-update filtering," in *Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2016, pp. 5880–5884.

[28] X. Glorot and Y. Bengio, "Understanding the difficulty of training deep feedforward neural networks," in *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics*, 2010, pp. 249–256.

[29] H. Sak, A. Senior, K. Rao, and F. Beaufays, "Fast and accurate recurrent neural network acoustic models for speech recognition," *arXiv preprint arXiv:1507.06947*, 2015.