Sub-band Convolutional Neural Networks for Small-footprint Spoken Term Classification

Chieh-Chi Kao, Ming Sun, Yixin Gao, Shiv Vitaladevuni, Chao Wang

Alexa Speech, Amazon
{chiehchi,mingsun,yixigao,shivnaga,wngcha}@amazon.com

Abstract

This paper proposes a Sub-band Convolutional Neural Network for spoken term classification. Convolutional neural networks (CNNs) have proven to be very effective in acoustic applications such as spoken term classification, keyword spotting, speaker identification, acoustic event detection, etc. Unlike applications in computer vision, the spatial invariance property of 2D convolutional kernels does not fit acoustic applications well since the meaning of a specific 2D kernel varies a lot along the feature axis in an input feature map. We propose a sub-band CNN architecture to apply different convolutional kernels on each feature sub-band, which makes the overall computation more efficient. Experimental results show that the computational efficiency brought by sub-band CNN is more beneficial for small-footprint models. Compared to a baseline full band CNN for spoken term classification on a publicly available Speech Commands dataset, the proposed sub-band CNN architecture reduces the computation by 39.7% on commands classification, and 49.3% on digits classification with accuracy maintained.

Index Terms: spoken term classification, convolutional neural network (CNN), sub-band feature

1. Introduction

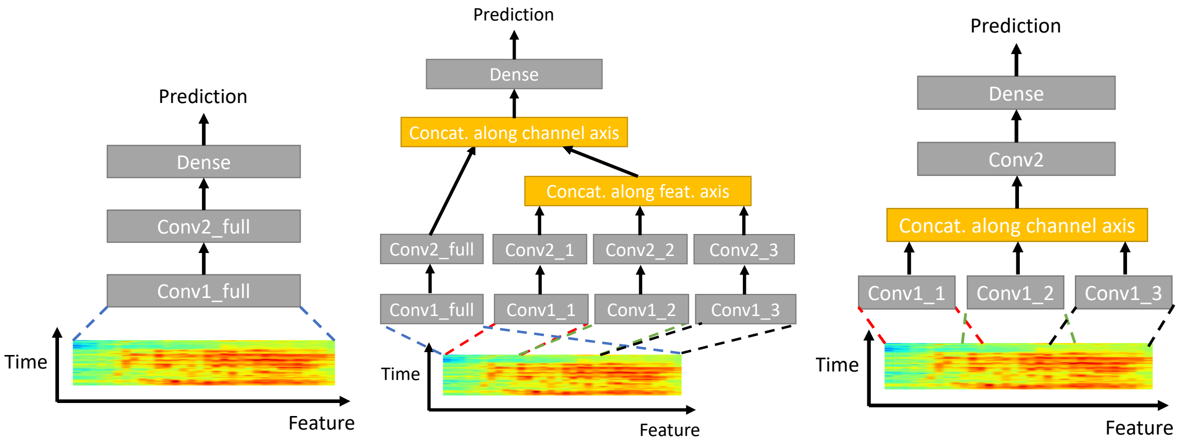
With the rapid development of public available datasets (e.g. spoken term classification [1], speaker identification [2, 3], acoustic event classification/detection [4, 5], etc.), state-of-the-art models for various acoustic applications can be trained with a large amount of annotated data. CNN-based architectures have achieved state-of-the-art performance in keyword spotting [6], speech recognition [7, 8], speaker identification [2, 3], acoustic event classification [9, 10, 11, 12], and rare acoustic event detection [13, 14]. CNNs have shown performance superior to feed-forward Deep Neural Networks (DNNs) in various acoustic applications due to the following reasons. First, DNN is not good at modeling the strong correlations in time and frequency of acoustic signal. Second, DNNs are not able to model shift of formants in speech signals. Instead, CNNs are able to capture local patterns that model the correlations properly and detect shift of formants by sharing the weights of 2D kernels (*time*×*feature*) across different locations in the input feature space.

An important property of CNN is shift invariance (also known as spatial invariance), which allows CNN to detect patterns even if it does not appear at exactly the same location as samples seen in the training set. Weights of learned 2D convolutional kernels are shared across different locations in the input feature map. This property is desirable for visual detection tasks, where the physical meaning of two dimensions in the input feature map are the same (i.e. x-axis and y-axis in an image). However, the spatial invariance property of 2D kernels does not fit acoustic applications well since the physical meaning of a specific 2D kernel holds only along the time axis in a input feature map, but varies a lot along the feature axis. To keep the shift invariance property locally, we propose an architecture of sub-band CNN by limiting the weight sharing within a certain sub-band on the feature axis.

In this work, we experimented the proposed architecture on Google Speech Commands dataset [1], which provides a common benchmark for keyword spotting and spoken term classification. The goal of both tasks is to detect a relatively small set of predefined keywords in an utterance. Different from keyword spotting that has been widely used for virtual assistants (e.g. Amazon Alexa, Google Assistant, Apple Siri), spoken term classification does not have the low-latency constraint since the classification is done at utterance level. Previous works [15, 16, 17, 18] showed that neural networks are very effective in keyword spotting. As tremendous efforts are dedicated into the discovery of effective CNN architectures for further advancing the performance, we argue that it is also important to investigate into effective ways for utilizing computational resource at inference time. Since most of the applications mentioned above run on mobile devices or smart speakers, a model with small memory footprint and low computational budget is required. While previous works used low-rank SVD [19, 20] and knowledge distillation [21, 22, 23] to make neural networks more compact, this work focuses on how to utilize the computational resource efficiently for CNNs. Compared with residual network for small-footprint keyword spotting (5.65M multiplications) [18], we explored a regime with much lower computational resource for spoken term classification, which is more than 20× reduction in the number of multiplications. [1]

We propose a simple approach to utilize the computational resource efficiently for CNNs that are dedicated for acoustic applications. Our approach applies different sets of convolutional kernels at each feature sub-band. Feature maps extracted from each sub-band are concatenated along the channel axis, and then fed into the next convolutional layer. Limited weight sharing (LWS) for convolutional layers has been proven very effective in speech recognition [7, 24]. LWS is explored for models with a single convolutional layer in these two works. A CNN that consists of one pair of convolution and max-pooling layers, and two fully connected hidden layers is used in [7]. LWS in either time or feature axis for models with a single layer convolutional layer are discussed in [24]. Although multi-layers CNN has been tested in [24], full weight sharing was used so that more than one convolutional layers can be stacked. Different from these works, we apply LWS to CNNs with multiple convolutional layers, and explore different ways to concatenate features

[1] We use FLOPS as the measurement of computation complexity in this paper, while the number of multiplication is used in [18]. For the proposed model, about half of the FLOPS are multiplications. We use this rate to convert FLOPS to the number of multiplication.



(a) *Full band (baseline)* (b) *Full band plus non-overlapped sub-band* (c) *Overlapped sub-band (proposed)*

Figure 1: *CNN models with different weight sharing methods. (a) The baseline model proposed in [6]. (b) Applying the multi-band approach proposed in [25] to the baseline model. (c) The proposed overlapped sub-band CNN. For the easiness of illustration, the x-axis is set as feature in this figure, which is different from conventional settings.*

Table 1: *Detailed architecture of full band CNN (Fig. 1a).*

Layer	Full band
Conv 1 (t×f, ch, stride)	20×8, K, 1×1
Activation 1	ReLU
Dropout 1	P
Maxpool (t×f, stride)	2×2, 2×2
Conv 2 (t×f, ch, stride)	10×4, K, 1×1
Activation 2	ReLU
Dropout 2	P
Dense (# outputs)	12

extracted from each sub-band.

Most similar to our approach of using sub-band CNN with multiple convolutional layers is the work of [25], which combines multiple non-overlapped sub-band sub-networks with a full band sub-network for audio source separation. Different from [25], we use overlapped sub-band networks without an extra full band sub-network. Overlapping between sub-bands helps to avoid information loss at the boundary between sub-bands. [25] used a full band sub-network to avoid this information loss brought by non-overlapped sub-bands. However, it may introduce redundant information as well as excessive computational costs. Our experimental results show that the proposed overlapped sub-band CNN performs better than the multi-band architecture proposed in [25] on spoken term classification. Recently, Phaye et. al [26] proposed an architecture using sub-spectrogram based CNN for acoustic scene classification. Different from these two works [25, 26] of applying LWS to CNNs with multiple convolutional layers, this work is the first one to concatenate features within a CNN rather than concatenating the feature extracted from the top most layer of a CNN. Experimental results in Sec. 3.2 show that concatenating features along channel axis within CNN outperforms other concatenation methods for sub-band CNN.

In the rest of this paper, we discuss our sub-band CNN approach in Section 2, demonstrate them on two tasks (commands and digits classification) of Speech Commands dataset [1] in Section 3, and provide conclusion remarks in Section 4.

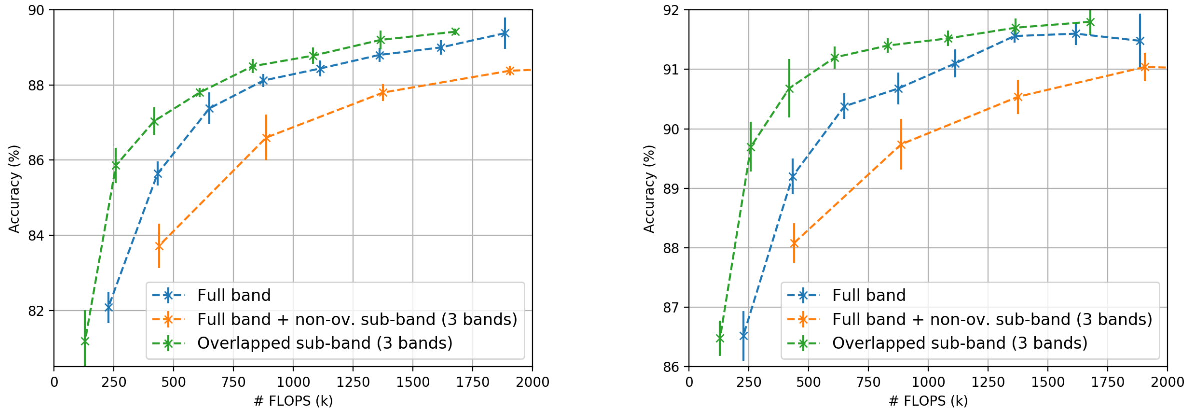
Table 2: *Detailed architecture of full band plus non-overlapped sub-band CNN (Fig. 1b).*

Layer	Band 1	Band 2	Band 3	Full band
Conv 1 (t×f, ch, str.)	20×8, K,1×1	20×8, K,1×1	20×8, K,1×1	20×8, K,1×1
Activ. 1	ReLU	ReLU	ReLU	ReLU
Dropout 1	P	P	P	P
Maxpool (t×f, str.)	2×2, 2×2	2×2, 2×2	2×2, 2×2	2×2, 2×2
Conv 2 (t×f, ch, str.)	10×4, K,1×1	10×4, K,1×1	10×4, K,1×1	10×4, K,1×1
Activ. 2	ReLU	ReLU	ReLU	ReLU
Dropout 2	P	P	P	P
Concat. (axis)	Feature			-
Concat. (axis)	Channel			
Dense (# o/p)	12			

2. Sub-band CNN

We show implementation details of the proposed sub-band CNN in this section. We chose the "`cnn-trad-fpool3`" model proposed in [6] as our baseline model. We used the implementation of "`cnn-trad-fpool3`" in Tensorflow official package [27] as the baseline, which is slightly different from the original model described in [6]. As shown in Fig. 1a, it consists of two convolutional layers followed by a dense layer. The detailed architecture of the baseline model is shown in Table 1. The convolutional layers are applied to the full band input feature map, which is equivalent to full weight sharing mentioned earlier.

We applied the proposed sub-band CNN idea to the baseline model, and Fig. 1c shows the architecture of the resulting model. First, the input feature map is split into B overlapped sub-bands (B=3 in Fig. 1c), and each sub-band has its own set of kernels at the first convolutional layer. The feature extracted from each sub-band after the first convolutional layer are concatenated along the channel axis, and then fed into the second convolutional layer. The high-level feature extracted by the second convolutional layer is then fed into a dense layer to generate the final prediction. Note that we set the number of kernels (K) in each convolutional block to be the same (i.e. there are K ker-



(a) *Commands classification*

(b) *Digits classification*

Figure 2: *Accuracy curve of different weight sharing methods on subsets of Google Speech Commands dataset [1]. Each data point represents an average of five trials, and the error bar is the sample standard deviation of five trials.*

Table 3: *Detailed architecture of overlapped sub-band CNN (Fig. 1c).*

Layer	Band 1	Band 2	Band 3
Conv 1 (t×f, ch, stride)	20×8, K, 1×1	20×8, K, 1×1	20×8, K, 1×1
Activation 1	ReLU	ReLU	ReLU
Dropout 1	P	P	P
Maxpool (t×f, stride)	2×2, 2×2	2×2, 2×2	2×2, 2×2
Concat. (axis)	Channel		
Conv 2 (t×f, ch, stride)	10×4, K, 1×1		
Activation 2	ReLU		
Dropout 2	P		
Dense (# outputs)	12		

nels in ‘Conv1_1’, ‘Conv1_2’, ..., ‘Conv1_B’, and ‘Conv2’ respectively). The detailed architecture is shown in Table 3.

We experimented with different number of sub-bands ({2,3,4}) on commands classification task. As shown in Fig. 3a, the model with 3 bands is comparable to the one with 4 bands, and both of them outperform the model with 2 bands. We chose the model with 3 bands for further experiments per Occam's razor. For comparison, we applied the multi-band approach proposed in [25] to the baseline model. It is shown in Fig. 1b, and the detailed architecture is shown in Table 2.

3. Experimental Results

3.1. Datasets

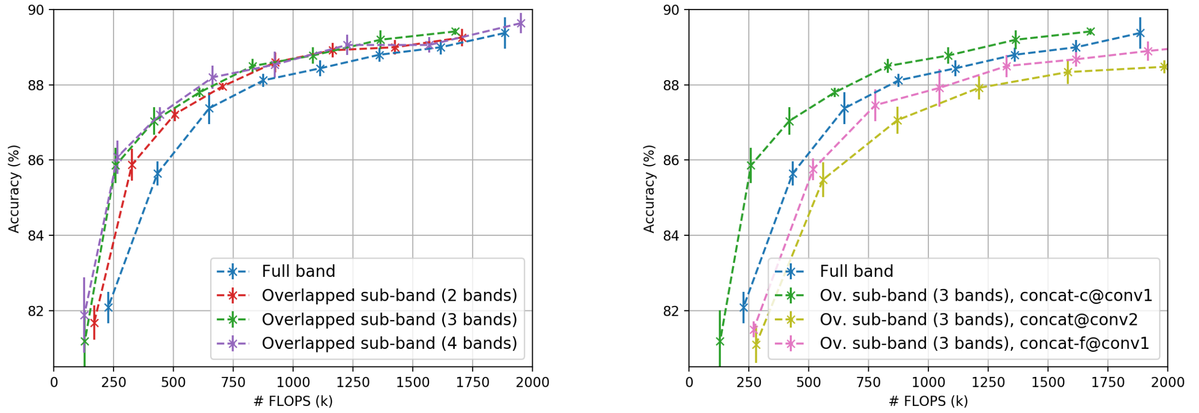
We tested the proposed model on Google Speech Commands dataset [1], which has 35 words in the latest version (v0.02). We chose two subsets as our testbed for spoken term classification tasks. For the formulation of subsets, we use the same setup as the “Audio Recognition” tutorial in the official Tensorflow package [28]. The task is formulated as a twelve-way classification. For each task, the subset consists of ten keywords, silence, and unknown (i.e. words not belong to the ten selected keywords). The first task is commands classification, which contains ten keywords as: “yes”, “no”, “up”, “down”, “left”, “right”, “on”, “off”, “stop”, or “go”. The second task is digits classification, which uses digits zero to nine as ten keywords. There are 36,923, 4,445, 4,890 samples for commands classification, and 37,390, 4,373, 4,929 samples for digits classification in train, dev, test sets respectively.

Feature extraction Each utterance is an one second clip with mono audio signals sampled at 16kHz. The acoustic features used in this work are mel-frequency cepstral coefficients (MFCCs), and the inputs fed to each sub-band sub-network are actually MFCC sub-vectors. For each one-second clip, we extract 40 dimensional MFCCs from frames of 30 ms duration with shifts of 10 ms.

3.2. Experimental Setups

We compare three weight sharing methods for CNN on spoken term classification: (1) full band (Fig. 1a), (2) full band plus non-overlapped sub-band (Fig. 1b), (3) overlapped sub-band (Fig. 1c). To investigate the performance of different model sizes, we experimented with different number of kernels (K) in each convolutional block. Every curve in Fig. 2 and Fig. 3 consists of data points generated with different values of K: {8, 16, 24, 32, 40, 48, 56, 64}. For each model with a specific K, we use floating point operations per second (FLOPS) as the measure of model complexity. Number of FLOPS is measured by *float_operation* function in the official Tensorflow profiler tool. We apply the same dropout probability (P=0.5) to all the models in this paper. All models are trained with stochastic gradient descent (SGD) optimizer with a minibatch size of 100. We train the models with an intial learning rate of 0.001 for 24k iterations, and drop the learning rate to 0.0001 for another 3k iterations. For the overlapped sub-band models shown in Fig. 3a, the bands are {[0,26), [14,40)} for 2 bands, {[0,16), [12,28), [24,40)} for 3 bands, and {[0,14), [8,22), [16,30), [26,40)} for 4 bands, respectively.

The evaluation metric used for spoken term classification in this work is accuracy. All the accuracies reported in this paper are the average of five random trials to reduce the effect caused by randomness during the training of CNN models. Error bars in all figures are the sample standard deviation of five trials. We chose accuracy as the evaluation metric rather than a Detection Error Tradeoff (DET) of false reject and false accept rate for the easiness to compare a large number of trained models in a plot. It's more succinct to represent five trials using one data point



(a) *Different number of bands* (b) *Different concatenation methods for sub-band features*

Figure 3: *Accuracy curve of experiments on number of sub-bands and concatenation methods for sub-band features. Commands classification is used as the testbed. Each data point represents an average of five trials, and the error bar is the sample standard deviation of five trials.*

with an error bar compared to plotting tens of DETs in a figure.

3.3. Results

Fig. 2a and Fig. 2b show the accuracy curves for different weight sharing methods on commands and digits classification. We have two major observations from these plots. First, overlapped sub-band outperforms the other two methods in for both cases. It shows that using overlapped sub-band CNNs to limit weight sharing within a narrow band works better on spoken term classification. This observation is aligned with the motivation of this work: spatial invariance property of CNN does not fit acoustic applications well, and feature axis should be treated different from time axis. Interestingly, full band plus non-overlapped sub-band CNN does not work better than the baseline model. We hypothesize that the full band sub-network and the non-overlapped sub-band sub-network may extract similar features, which causes redundancy in computation. Second, the computational efficiency brought by overlapped sub-band CNN is more beneficial for small-footprint models (i.e. in the region with lower FLOPS). If we set the target accuracy as the baseline model (full band) with 500k FLOPS, the proposed sub-band CNN architecture reduces the computation (in terms of FLOPS) by 39.7% on commands classification, and 49.3% on digits classification. Similarly, if the target accuracy is set as the baseline model with 1,000k FLOPS, the reduction is 23.7% on commands classification, and 50.1% on digits classification. Given the same K, we found that the decrease in FLOPS of the proposed method comes from the decrease of number of points in the feature map generated by *conv2*. Feed inputs with less number of points to the final dense layer significantly reduce the required computation. From the trend of curves shown in Fig. 2, we suspect that full band model and overlapped sub-band model may have similar performance when unlimited computational budget is given.

3.4. Concatenation of Sub-band Features

To investigate the effect of different methods to concatenate feature maps from each sub-band, we tested three settings as following: (1) *concat-c@conv1*: Concatenate along channel axis after the first convolutional layer. This is the *overlapped sub-band CNN* in all other sections throughout this paper. (2) *concat@conv2*: Each sub-band sub-network has two convolutional layers, and we concatenate all feature maps after the second convolutional layer. The axis for concatenation does not matter under this setting since the concatenated feature is further fed into a dense layer. (3) *concat-f@conv1*: Concatenate along feature axis after the first convolutional layer. As shown in Fig. 3b, *concat-c@conv1* outperforms the other two concatenation methods. By concatenating along channel axis, the receptive field of each point in the concatenated feature map after the first convolutional layer has been tripled in feature axis (from 21×9 to 21×27). Larger receptive field enables the feature map responds to large enough areas in the input feature map to capture information about acoustic signals spread across different feature dimensions. Traditionally, larger receptive field is achieved by stacking more convolutional layers, which is less feasible for mobile devices or smart speakers. The proposed sub-band CNN provides an alternative way to achieve larger receptive field, which is suited for small-footprint model on spoken term classification.

4. Conclusions

In this paper, we proposed a sub-band CNN architecture and explored it for spoken term classification. We compare the proposed sub-band CNNs to full band CNNs and another weight sharing approach on two spoken term classification tasks. The proposed architecture of sub-band CNNs reduces the computation by 39.7% on commands classification, and 49.3% on digits classification, to achieve the same accuracy as the baseline full band CNN with 500k FLOPS. We found that the computational efficiency brought by sub-band CNN is more beneficial for small-footprint models. Potential applications for the proposed architecture include on-device speech command recognition, acoustic event detection, etc.

5. Acknowledgement

The authors would like to thank Weiran Wang, Krishna Puvvada, and Wei-Ning Hsu for useful discussions.

6. References

[1] P. Warden, “Speech commands: A dataset for limited-vocabulary speech recognition,” *CoRR*, vol. abs/1804.03209, 2018.

[2] A. Nagrani, J. S. Chung, and A. Zisserman, “Voxceleb: a large-scale speaker identification dataset,” in *INTERSPEECH*, 2017, pp. 2616–2620.

[3] J. S. Chung, A. Nagrani, and A. Zisserman, “Voxceleb2: Deep speaker recognition,” in *INTERSPEECH*, 2018, pp. 1086–1090.

[4] J. F. Gemmeke, D. P. W. Ellis, D. Freedman, A. Jansen, W. Lawrence, R. C. Moore, M. Plakal, and M. Ritter, “Audio set: An ontology and human-labeled dataset for audio events,” in *IEEE ICASSP*, 2017, pp. 776–780.

[5] A. Mesaros, T. Heittola, A. Diment, B. Elizalde, A. Shah, E. Vincent, B. Raj, and T. Virtanen, “DCASE 2017 challenge setup: Tasks, datasets and baseline system,” in *Proceedings of the Detection and Classification of Acoustic Scenes and Events 2017 Workshop (DCASE2017)*, November 2017, pp. 85–92.

[6] T. N. Sainath and C. Parada, “Convolutional neural networks for small-footprint keyword spotting,” in *INTERSPEECH*, 2015, pp. 1478–1482.

[7] O. Abdel-Hamid, A. Mohamed, H. Jiang, and G. Penn, “Applying convolutional neural networks concepts to hybrid nn-hmm model for speech recognition,” in *IEEE ICASSP*, 2012, pp. 4277–4280.

[8] Y. Qian, M. Bi, T. Tan, and K. Yu, “Very deep convolutional neural networks for noise robust speech recognition,” *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 24, no. 12, pp. 2263–2276, Dec 2016.

[9] N. Takahashi, M. Gygli, B. Pfister, and L. V. Gool, “Deep convolutional neural networks and data augmentation for acoustic event recognition,” in *INTERSPEECH*, 2016, pp. 2982–2986.

[10] S. Hershey, S. Chaudhuri, D. P. W. Ellis, J. F. Gemmeke, A. Jansen, R. C. Moore, M. Plakal, D. Platt, R. A. Saurous, B. Seybold, M. Slaney, R. J. Weiss, and K. Wilson, “Cnn architectures for large-scale audio classification,” in *IEEE ICASSP*, 2017, pp. 131–135.

[11] B. Shi, M. Sun, C. Kao, V. Rozgic, S. Matsoukas, and C. Wang, “Semi-supervised acoustic event detection based on tri-training,” in *IEEE ICASSP*, 2019, pp. 750–754.

[12] Q. Tang, M. Sun, C. Kao, V. Rozgic, and C. Wang, “Hierarchical residual-pyramidal model for large context based media presence detection,” in *IEEE ICASSP*, 2019, pp. 3312–3316.

[13] H. Lim, J. Park, and Y. Han, “Rare sound event detection using 1D convolutional recurrent neural networks,” DCASE2017 Challenge, Tech. Rep., September 2017.

[14] C. Kao, W. Wang, M. Sun, and C. Wang, “R-CRNN: region-based convolutional recurrent neural network for audio event detection,” in *INTERSPEECH*, 2018, pp. 1358–1362.

[15] G. Chen, C. Parada, and G. Heigold, “Small-footprint keyword spotting using deep neural networks,” in *IEEE ICASSP*, 2014, pp. 4087–4091.

[16] Y. He, R. Prabhavalkar, K. Rao, W. Li, A. Bakhtin, and I. McGraw, “Streaming small-footprint keyword spotting using sequence-to-sequence models,” in *IEEE Automatic Speech Recognition and Understanding Workshop (ASRU)*, 2017, pp. 474–481.

[17] S. O. 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,” in *INTERSPEECH*, 2017, pp. 1606–1610.

[18] R. Tang and J. Lin, “Deep residual learning for small-footprint keyword spotting,” in *IEEE ICASSP*, 2018, pp. 5484–5488.

[19] 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.

[20] M. Sun, D. Snyder, Y. Gao, V. Nagaraja, M. Rodehorst, S. Panchapagesan, N. Strom, S. Matsoukas, and S. Vitaladevuni, “Compressed time delay neural network for small-footprint keyword spotting,” in *INTERSPEECH*, 2017, pp. 3607–3611.

[21] L. Lu, M. Guo, and S. Renals, “Knowledge distillation for small-footprint highway networks,” in *IEEE ICASSP*, 2017, pp. 4820–4824.

[22] R. Pang, T. Sainath, R. Prabhavalkar, S. Gupta, Y. Wu, S. Zhang, and C.-C. Chiu, “Compression of end-to-end models,” in *INTERSPEECH*, 2018, pp. 27–31.

[23] B. Shi, M. Sun, C. Kao, V. Rozgic, S. Matsoukas, and C. Wang, “Compression of acoustic event detection models with quantized distillation,” to appear in INTERSPEECH, 2019.

[24] O. Abdel-Hamid, L. Deng, and D. Yu, “Exploring convolutional neural network structures and optimization techniques for speech recognition,” in *INTERSPEECH*, 2013, pp. 3366–3370.

[25] N. Takahashi and Y. Mitsufuji, “Multi-scale multi-band densenets for audio source separation,” in *IEEE Workshop on Applications of Signal Processing to Audio and Acoustics (WASPAA)*, 2017, pp. 21–25.

[26] S. S. R. Phaye, E. Benetos, and Y. Wang, “Subspectralnet - using sub-spectrogram based convolutional neural networks for acoustic scene classification,” in *IEEE ICASSP*, 2019, pp. 825–829.

[27] M. Abadi, A. Agarwal et al., “TensorFlow: Large-scale machine learning on heterogeneous systems,” 2015, software available from tensorflow.org. [Online]. Available: https://www.tensorflow.org/

[28] “TensorFlow: Simple audio recognition.” [Online]. Available: https://www.tensorflow.org/tutorials/sequences/audio_recognition/