INTERSPEECH 2021
30 August – 3 September, 2021, Brno, Czechia

An Integrated Framework for Two-pass Personalized Voice Trigger

Dexin Liao[1]*, Jing Li*[1]*, Yiming Zhi*[1]*, Song Li*[2]*, Qingyang Hong*[1]*, Lin Li*[2]

[1]School of Informatics, Xiamen University, China
[2]School of Electronic Science and Engineering, Xiamen University, China
qyhong@xmu.edu.cn, lilin@xmu.edu.cn

Abstract

In this paper, we present the XMUSPEECH system for Task 1 of 2020 Personalized Voice Trigger Challenge (PVTC2020). Task 1 is a joint wake-up word detection with speaker verification on close talking data. The whole system consists of a keyword spotting (KWS) sub-system and a speaker verification (SV) sub-system. For the KWS system, we applied a Temporal Depthwise Separable Convolution Residual Network (TDSC-ResNet) to improve the system's performance. For the SV system, we proposed a multi-task learning network, where phonetic branch is trained with the character label of the utterance, and speaker branch is trained with the label of the speaker. Phonetic branch is optimized with connectionist temporal classification (CTC) loss, which is treated as an auxiliary module for speaker branch. Experiments show that our system gets significant improvements compared with baseline system.

Index Terms: Keyword spotting, Speaker verification, TDSC-ResNet, Multi-task, CTC

1. Introduction

Voice wake-up is becoming popular in people everyday life as smart devices are widely used. Usually, detecting a trigger phrase including two stages, the first is a KWS system to detect the prefixed keywords, and the second is a SV system to conform the identity of the speaker. PVTC2020 [1] includes joint wake-up word detection with speaker verification on close talking data (task 1) and joint wake-up detection with speaker verification on far-field multi-channel microphone array data (task 2). More details about the dataset, task and evaluation metrics refer to [1]. In this paper, we mainly focus on task 1. From a practical standpoint, the whole system should satisfy the requirements of low latency. Typically, SV task needs lager networks than KWS task, an always-on SV model will lead to unnecessary computation and high latency. So we employed a cascaded 2-stage architecture, where a low cost KWS system decides whether triggered and then the keyword segment is fed to a larger SV system to do further identification.

Currently, the Deep-KWS [2] system has become more popular for its simpler implementation and higher accuracy. Many Convolutional Neural Networks (CNNs) architectures [3–9] have been explored for KWS system to achieve high accuracy with limited model size and computational resources. Temporal Convolution (TConv) [6] extracts the whole level frequency features to address the problem that conventional 2D CNNs struggle with concentrating the relation between high and low frequencies. Depthwise Separable Convolution Neural Network [10] considers the channel realm and space realm separately, reduces the number of the parameters of the standard convolution without significant performance degradation, and is well applied on KWS [5]. To achieve a fast and accurate KWS system, we applied TConv and depthwise separable convlolution on our Deep-KWS system.

In SV field, x-vector framework proposed by Snyder et al. [11] is the most popular framework recently. X-vector can make full use of deep neural network and significantly improve the performance of SV systems. In recent years, multi-task (MT) frameworks have been applied to combine speech with speaker information to improve the performance of SV systems [12–14]. These works showed that we can add frame-level phonetic information learned from multitasking before the pooling layer. It can help the SV system to distinguish speaker-specific information more easily. [15, 16] attempted to jointly optimize KWS and SV with a single network. In the speaker dependent voice trigger task, the SV system aims to verify the speaker identity of utterance containing prefixed keywords. Due to the above, we reproduced the frame-level MT framework [13] to better utilize the phonetic information in SV system. However, this method needs to get the phoneme labels of the inputs on frame-level. Usually the alignment is generated by a well trained automatic speaker recognition (ASR) model. The procedure is complex and the accuracy of labels is subject to the performance of ASR model. What's more, the ASR model might have decoding failure when processing the audio which is domain mismatched or noisy. CTC [17] can automatically label unsegmented sequence data and has achieved great success in many end-to-end applications [18–20]. Motivated by [15], we considered that CTC could be a simpler approach to embed frame-level phonetic information into SV system via MT learning. Thus, we proposed a multi-CTC x-vector network for PVTC2020 task 1 and significantly improved the performance compared with the official baseline.

We demonstrate the whole system in Section 2. Experimental details are presented in Section 3, and results are reported and analyzed in Section 4. Finally, we conclude this work in Section 5.

2. System description

2.1. KWS sub-system

The KWS system consists of three modules:(i) a feature extraction module, (ii) a deep neural network, and (iii) a posterior handling module. The features are 80-dimensional log-mel filterbank (FBank) computed with a 25ms window and shifted every 10ms. And we apply a window of 40 frames, which contain context information of sub-keywords, to generate training samples as the input of the model. We choose CNN as the acoustic model of KWS system. In the posterior handling module, the sequence of acoustic features is projected to a posterior probabilities sequences of the sub-words after the acoustic model. We smooth the posteriors over a fixed window of size 50 by taking average, and the confidence score is computed within a sliding window of size 150. The method of confidence computation is proposed in [21], in which the confidence score subject to the

http://dx.doi.org/10.21437/Interspeech.2021-2161

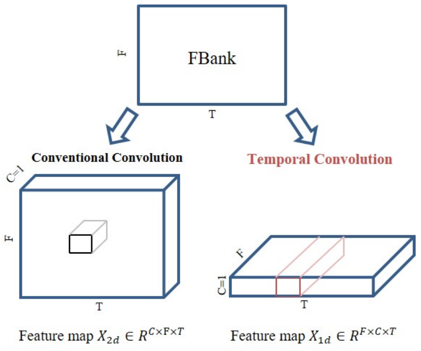
F
FBank
T
Conventional Convolution
Temporal Convolution
C=1
F
T
F
C=1
T
Feature map $X_{2d} \in R^{C\times F\times T}$
Feature map $X_{1d} \in R^{F\times C\times T}$

Figure 1: *Schematic diagram of the difference between Conventional Convolution and Temporal Convolution.*

constraint that the sub-keywords are uttered in the same order as in the specified keyword. The system triggers whenever the confidence score is higher than a predefined threshold.

2.1.1. *Temporal convolution*

The Temporal Convolution ResNet (TC-ResNet) model [6] is composed of sequence of residual blocks which use one dimensional convolution named Temporal Convolution (TConv). We treat the input FBank as time series. Conventionally, the FBank features are denoted as $\mathbf{I} \in \mathbb{R}^{F\times T}$, where F represents the dimension of FBank feature, and T denotes the number of frames. As shown in Figure 1. Conventional CNNs reshape the input from $\mathbf{I}$ to $\mathbf{X_{2d}} \in \mathbb{R}^{C\times F\times T}$, where $C = 1$ means the channel dimension. In TConv, the feature dimension is equal to the channels of the input feature map, so the input feature map X is transformed to $\mathbf{X_{1d}} \in \mathbb{R}^{F\times C\times T}$, where $F = 80$ and $C = 1$. Therefore, all convolutions in the model are along the temporal dimension, avoiding stacking a large number of layers to capture high-level frequency features. In addition, TConv requires a smaller number of computations compared with the conventional 2d convolution. What's more, it can decrease the output feature map (i.e., the input feature map of the next layer) of the TConv and dramatically reduce the computations in the following layers.

2.1.2. *Model architecture overview*

Our Temporal Depthwise Separable Convolution ResNet (TDSC-ResNet) is based on MatchboxNet [22]. As shown in Figure 2, the model uses TConv and depthwise and pointwise convolution. We first apply SpecAugment [23] on the bottom of the architecture thus enable the acoustic model to become more robust. After SpecAugment, the input feature map $\mathbf{I} \in \mathbb{R}^{F\times T}$ is transformed to $\mathbf{X_{1d}} \in \mathbb{R}^{F\times 1\times T}$ as illustrated in Sec. 2.1.1. Then TConv and Squeeze-and-Excitation (SE) [24] block can better accommodate feature maps. Inside the repeated blocks are separable convolution, 1×1 pointwise convolutions, batch norm, ReLU, and dropout (see the right part of Figure 2), and we set the $n = 3$ which means there is 3 blocks in the architecture. Finally we position two layers of additional TConv followed by an average pooling layer and then a fully connected layer and a softmax activation layer are applied to obtain the sub-words occurrence probability of the keyword. The detail settings of the model is listed in Table 1. The receptive field of the model is designed according to the length of input samples.

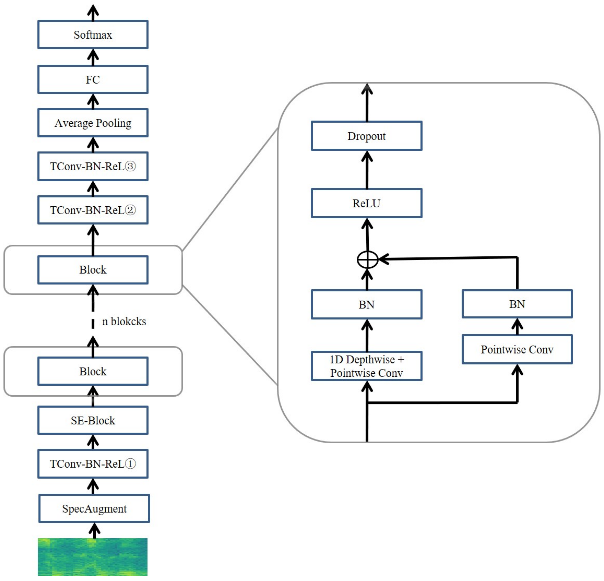

Figure 2: *Schematic diagram of TDSC-ResNet.*

Table 1: *The configuration of TDSC-ResNet. The number after Block denotes the index of the repeated blocks.*

Layers	Kernel size	Channels
TConv①	1x6	128
Block① Depthwise Conv	1x7	128
Block① Pointwise Conv	1x1	64
Block② Depthwise Conv	1x8	64
Block② Pointwise Conv	1x1	64
Block③ Depthwise Conv	1x9	64
Block③ Pointwise Conv	1x1	64
TConv②	1x14	128
TConv③	1x1	128

2.2. SV sub-system

The standard x-vector system supports the input to have fixed length. We first modified the x-vector system to allow various lengths input to better fit the CTC loss training. The feature is 40-dim MFCCs with 3-dim Pitch, 25ms frame length and 10ms shift. Cepstral mean-normalization (CMN) with a sliding window of 3 seconds and voice active detection (VAD) are also applied. In the training phase, the training data contains keyword and other contents. But in the evaluation phase, the SV system extract discriminative speaker embeddings of the keyword segments, which are much shorter than training data. Hence, we optimize the model through two steps. First the model is trained with all the training data. Then only the keyword segments are used to fine-tune the model again. SpecAugment is applied during training. In enrollment stage, the average vector of the three utterances' embedding extracted from the target speaker is saved as the enrollment speaker embedding vector.

2.2.1. *Connectionist temporal classifier*

Unlike cross entropy, in which the alignments between input and target sequences should be given, the CTC loss automatically learn the alignments. Suppose there are N modeling la-

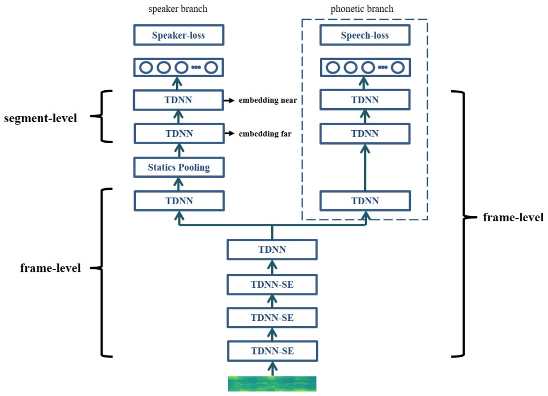

Figure 3: *Frame-level phonetic MT x-vector framework.*

bel units, the softmax layer before CTC consists of N units and a blank unit. The introduction of blank unit enable outputs to define the probabilities of all possible alignments of the input sequence. Given a T frames utterance, the CTC path $\pi = (\pi_0, ..., \pi_{T-1})$ contains repetitions of no-blank labels and the blank unit. After a many-to-one mapping function β, which means removing the repeated labels and blanks, the CTC path can be mapped to the corresponding label sequence (e.g. β(a-ab-) = β(-aa- -abb) = aab). Summing all the possibilities of CTC paths has heavy computing burden, and a forward-backward dynamic programming algorithm [17] was proposed to solve this problem. Given the label sequence, CTC naturally tends to align each label prediction. In this way, the sequence of frame-level posterior probability distribution before CTC loss is similar to that before CE loss, while the later requires a predetermined alignment.

2.2.2. *The frame-level multi-task learning*

We adopt time delay neural network (TDNN) with SE-block as the speaker embedding extractor, which is optimized by AAM-Softmax [25] loss. The detailed configuration is shown in Table 2. The MT system is depicted in Figure 3 and the configuration of TDNNs in phonetic branch is the same as that in speaker part. Unlike the MT system in [15], the phonetic branch and the speaker branch in our MT system are trained jointly by a total loss function (1) for each examples, and the phonetic branch is more inclined to assist SV branch to achieve a better performance. After training, the phonetic part is removed from the model. For the back-end, the $embedding\ far$ in Figure 3 is for PLDA scoring and $embedding\ near$ for cosine scoring. PLDA model is trained corresponding to the train set. Submean and normalization are applied during scoring.

$$L_{loss} = L_{speaker} + \alpha L_{speech} \tag{1}$$

For the multi-Cross-Entropy x-vector (MT-CE-XV) system, the phonetic branch in Figure 3 is trained by CE loss. To get the aligned label, we used the PVTC training data to train a lattice-free MMI based ASR [26] refer to Kaldi [27] recipe[1]. Then, the frame-level phoneme alignment is obtained through the ASR model. Assume an utterance with T frames, the L_{speech} in (1) is defined as (2)

$$L_{speech} = \frac{1}{T}\sum_{t=1}^{T} CE(O_t^p, Y_t^p) \tag{2}$$

Table 2: *The configuration of the speaker embedding extractor.*

Layer-index	Layer	Context	SE-ratio	Size
1	TDNN-SE	{t-2:t+2}	4	512
2	TDNN-SE	{t - 2,t,t + 2}	8	512
3	TDNN-SE	{t - 3,t,t + 3}	8	512
4	TDNN	{t}		512
5	TDNN	{t}		1500
6	Stats pooling	[0,T]		2 x 1500
7	TDNN	{0}		512
8	TDNN	{0}		512
9	AAM-Softmax	{0}		Num.spk

where O_t^p denotes the output of the phonetic branch at time t, Y_t^p is its corresponding frame-level phoneme label.

For the multi-CTC x-vector (MT-CTC-XV) system, the L_{speech} is (3)

$$L_{speech} = L_{CTC}^{\pi}(Y_l) \tag{3}$$

where π is the possibility distribution permutation before the CTC and Y_l is the label sequence.

A multi-CE-CTC x-vector (MT-CTC-CE-XV) system is a combination of CE and CTC loss in MT learning. That is, there are two phonetic branches, one for CE loss and another for CTC. Here the L_{speech} is (4)

$$L_{speech} = \frac{1}{T}\sum_{t=1}^{T} CE(O_t^p, Y_t^p) + L_{CTC}^{\pi}(Y_l) \tag{4}$$

3. Experimental setup

All of the models are implemented based on our open source toolkits:ASV-Subtool [28]. In the KWS system, The validation set is randomly selected 10 out of 300 speakers' data in the training set. The keyword 'xiao le xiao le' is split to three sub-keywords (i.e., '1_xiao le', '2_le xiao', '3_xiao le'), and each sub-keyword contains an interval between Chinese characters inside the keyword. The labels for each input is determined by force-alignment with a large ASR system as the official pipeline. In the evaluation period, a simple VAD is applied to detect voice activity, and only the audio clips which trigger the KWS system are saved to be further transformed to speaker embeddings.

To train the SV system, the model is first pre-trained by the all training data and then fine-tuned by the keyword segments. Here are some details about training strategy:

- KWS: Stochastic gradient descent (SGD) is used with a batch size of 128 for 100 epochs. The initial learning rate is set as 0.01 and decreases by a factor of 0.5 when the model reaches a valid loss plateau. Early stopping is employed when the valid loss is not decreasing.
- Pre-train SV: The Radam optimizer [29] is used to optimize the models with a batch size of 32 for 20 epochs.
- Fine-tune SV: SGD optimizer with a batch size of 128 for 20 epochs. The learning rate is initialized as 0.03 and decreases by a factor of 0.5 when the model reaches a valid loss plateau.

4. Results and analysis

4.1. Performance of KWS sub-system

We choose the false rejection rate under one false alarm per hour as the KWS system performance criterion. In a real scenario,

[1] https://github.com/kaldi-asr/kaldi/blob/master/egs/aishell/s5/run.sh

Table 4: *Results of different systems.*

System index	**KWS system**	**SV system**	**SV-dev-EER%**	**SV-dev-minDCF**	$score_{dev}$	$score_{test}$
S1	KWS-baseline [1]	SV-baseline [1]	1.32	0.16	0.10	0.37
S2 [30]						0.075
S3 [31]					0.042	0.081
S4	XMU-KWS	SV-baseline	1.368	0.162	0.090	0.161
S5	XMU-KWS	x-vector	1.572	0.195	0.099	0.189
S6	XMU-KWS	MT-CE-XV	1.159	0.152	0.073	0.168
S7	XMU-KWS	MT-CTC-XV-phoneme	1.291	0.143	0.078	0.162
S8	XMU-KWS	MT-CTC-XV-char	**0.974**	**0.112**	**0.062**	**0.161**
S9	XMU-KWS	MT-CTC-CE-XV	1.237	0.123	0.073	0.168
S10	XMU-KWS	MT-CTC-XV-char + score-fusion with PLDA	**0.808**	**0.086**	**0.051**	**0.140**
S11		+ Data augmentation	0.786	0.083	0.049	0.126
S12		+ model-fusion with SV-baseline	**0.570**	**0.061**	**0.038**	**0.098**

Table 3: *Performance of the KWS models on dev set (the false rejection (FR) rate [%] under one false alarm (FA) per hour).*

Models	**Params**	**FR**	**RTF**
Baseline [1]	240K	2.00	0.076
TDSC-ResNet	231K	0.72	0.043

the main cost of time comes from the always-on KWS system. We first calculate the process time T_{kws} of KWS system which consists of the time to trigger the system and the time to generate the keyword clips. The real time factor (RTF) is calculated as follows:

$$RTF = T_{kws}/T_{total\_dev} \qquad (5)$$

where T_{total_dev} is the total duration of the development set audios, and here $T_{total\_dev} = 20.13$. The KWS systems are evaluated on an Intel(R) Xeon(R) E5-2643 v4 CPU clocked at 3.4 GHz. As shown in Table 3, our KWS system has higher accuracy and lower latency compared with the baseline.

4.2. Performance of SV models

After KWS system, the keyword segments of the whole utterance in development set compose a new dataset, which will determine the threshold of the following SV system. We choose the mean threshold of Equal Error Rate (EER) and minDCF [32] as the threshold of the SV system.

The statistic $score$ is calculated from the miss rate and the FA rate according to the following equation:

$$score = Miss + alpha * FA \qquad (6)$$

where $Miss$ represents the proportion of errors in all positive label samples, and *FA* refers to the rate of errors in all negative label samples. The *alpha* constant is set as 19, which is calculated by the assumption that the probability of the positive samples is 0.05.

The results of different systems are shown in Table 4. From System S4 to S10, all SV models apply a cosine scoring except the one in S10, which also uses PLDA scoring with the $embedding\ far$ in Figure 3. As the official baseline selected data on OpenSLR[2] as pre-training data, our x-vector system in S5 performs worse than S4. We observe that all the multi-task learning SV models from S6 to S10 achieve better performance than S5. It proves the point that the frame-level phonetic information obtained by multi-task benefits SV system. We choose the MT-CTC-XV-char SV system and use PLDA to estimate scores, then the cosine scores and PLDA scores of S10 are fused. Compared with SV-baseline, the single SV system in S10 gets better performance without other pre-training data. To make SV models more robust, MUSAN [33] and RIRs [34] noises are added to the training data as a method of data augmentation (S11). Finally, with the score fusion on SV systems in S11 and SV-baseline (S4), we get a competitive result (S12) compared with the top 2 teams (S2,S3), and the score gap on the evaluation set may stem from the additional information of other pre-training data.

4.3. Effect of multi-task learning

Meanwhile, different types of multi-task learning are also investigated. Compared with MT-CE-XV (S6), MT-CTC-XV (S7,S8) trains with the label sequence directly and automatically utilizes phonetic information. We find that S8 outperforms S7, and the difference between them is the CTC modeling unit. The Chinese char unit contains more context information compared with phoneme unit, and contributes better performance. However, combining CTC loss and CE loss in multi-CE-CTC (S9) does not lead to further improvement, suggesting that the two phonetic branches may have learned duplicate frame-level phonetic information.

5. Conclusions

This paper presents a two-pass system for personalized voice trigger. We applied TDSC-ResNet to Deep-KWS system and made it more accurate and faster. In addition, various multi-task learning framework are explored to jointly learn frame-level phonetic information in SV system. What's more, a MT-CTC-XV system, which is simpler than MT-CE-XV system, is proposed to utilize the phonetic information for SV task. The system can be easily applied to transfer learning and get further improvement. After score fusion, the single MT-CTC-XV sub-system gets a better performance compared with official SV system. At last, the fusion system achieves a score of 0.098 on evaluation set.

6. Acknowledgements

The work is supported by the National Natural Science Foundation of China (Grant No.61876160 and No.62001405).

[2] http://openslr.org/resources.php

7. References

[1] Y. Jia, X. Wang, X. Qin, Y. Zhang, X. Wang, J. Wang, and M. Li, "The 2020 personalized voice trigger challenge: Open database, evaluation metrics and the baseline systems," *arXiv preprint arXiv:2101.01935*, 2021.

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

[3] R. Tang and J. Lin, "Deep residual learning for small-footprint keyword spotting," in *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2018, pp. 5484–5488.

[4] A. Coucke, M. Chlieh, T. Gisselbrecht, D. Leroy, M. Poumeyrol, and T. Lavril, "Efficient keyword spotting using dilated convolutions and gating," in *ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2019, pp. 6351–6355.

[5] M. Xu and X.-L. Zhang, "Depthwise separable convolutional resnet with squeeze-and-excitation blocks for small-footprint keyword spotting." in *Interspeech 2020*, 2020, pp. 2547–2551.

[6] S. Choi, S. Seo, B. Shin, H. Byun, M. Kersner, B. Kim, D. Kim, and S. Ha, "Temporal convolution for real-time keyword spotting on mobile devices." in *Interspeech 2019*, 2019, pp. 3372–3376.

[7] C. Yang, X. Wen, and L. Song, "Multi-scale convolution for robust keyword spotting," *Proc. Interspeech 2020*, pp. 2577–2581, 2020.

[8] X. Li, X. Wei, and X. Qin, "Small-footprint keyword spotting with multi-scale temporal convolution," *arXiv preprint arXiv:2010.09960*, 2020.

[9] T. Higuchi, M. Ghasemzadeh, K. You, and C. Dhir, "Stacked 1d convolutional networks for end-to-end small footprint voice trigger detection," *arXiv preprint arXiv:2008.03405*, 2020.

[10] A. G. Howard, M. Zhu, B. Chen, D. Kalenichenko, W. Wang, T. Weyand, M. Andreetto, and H. Adam, "Mobilenets: Efficient convolutional neural networks for mobile vision applications," *arXiv preprint arXiv:1704.04861*, 2017.

[11] D. Snyder, D. Garcia-Romero, D. Povey, and S. Khudanpur, "Deep neural network embeddings for text-independent speaker verification." in *Interspeech*, 2017, pp. 999–1003.

[12] Y. Liu, L. He, J. Liu, and M. T. Johnson, "Speaker embedding extraction with phonetic information," *arXiv preprint arXiv:1804.04862*, 2018.

[13] S. Wang, J. Rohdin, L. Burget, O. Plchot, Y. Qian, K. Yu, and J. Cernockỳ, "On the usage of phonetic information for text-independent speaker embedding extraction." in *Interspeech*, 2019, pp. 1148–1152.

[14] M. Zhao, R. Li, S. Yan, Z. Li, H. Lu, S. Xia, Q. Hong, and L. Li, "Phone-aware multi-task learning and length expanding for short-duration language recognition," in *2019 Asia-Pacific Signal and Information Processing Association Annual Summit and Conference (APSIPA ASC)*. IEEE, 2019, pp. 433–437.

[15] S. Sigtia, E. Marchi, S. Kajarekar, D. Naik, and J. Bridle, "Multi-task learning for speaker verification and voice trigger detection," in *ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2020, pp. 6844–6848.

[16] M. Jung, Y. Jung, J. Goo, and H. Kim, "Multi-task network for noise-robust keyword spotting and speaker verification using ctc-based soft vad and global query attention," *arXiv preprint arXiv:2005.03867*, 2020.

[17] 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*, 2006, pp. 369–376.

[18] S. Kim, T. Hori, and S. Watanabe, "Joint ctc-attention based end-to-end speech recognition using multi-task learning," in *2017 IEEE international conference on acoustics, speech and signal processing (ICASSP)*. IEEE, 2017, pp. 4835–4839.

[19] S. Karita, N. E. Y. Soplin, S. Watanabe, M. Delcroix, A. Ogawa, and T. Nakatani, "Improving transformer-based end-to-end speech recognition with connectionist temporal classification and language model integration," in *20th Annual Conference of the International Speech Communication Association: Crossroads of Speech and Language, INTERSPEECH 2019*, 2019, pp. 1408–1412.

[20] T. N. Sainath, R. Pang, D. Rybach, Y. He, R. Prabhavalkar, W. Li, M. Visontai, Q. Liang, T. Strohman, Y. Wu *et al.*, "Two-pass end-to-end speech recognition," *arXiv preprint arXiv:1908.10992*, 2019.

[21] B. Liu, S. Nie, Y. Zhang, S. Liang, Z. Yang, and W. Liu, "Focal loss and double-edge-triggered detector for robust small-footprint keyword spotting," in *ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2019, pp. 6361–6365.

[22] S. Majumdar and B. Ginsburg, "Matchboxnet–1d time-channel separable convolutional neural network architecture for speech commands recognition," *arXiv preprint arXiv:2004.08531*, 2020.

[23] D. S. Park, W. Chan, Y. Zhang, C.-C. Chiu, B. Zoph, E. D. Cubuk, and Q. V. Le, "Specaugment: A simple data augmentation method for automatic speech recognition." in *Interspeech 2019*, 2019, pp. 2613–2617.

[24] J. Hu, L. Shen, and G. Sun, "Squeeze-and-excitation networks," in *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2018, pp. 7132–7141.

[25] J. Deng, J. Guo, N. Xue, and S. Zafeiriou, "Arcface: Additive angular margin loss for deep face recognition," in *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019, pp. 4690–4699.

[26] D. Povey, V. Peddinti, D. Galvez, P. Ghahremani, V. Manohar, X. Na, Y. Wang, and S. Khudanpur, "Purely sequence-trained neural networks for asr based on lattice-free mmi." in *Interspeech*, 2016, pp. 2751–2755.

[27] D. Povey, A. Ghoshal, G. Boulianne, L. Burget, O. Glembek, N. Goel, M. Hannemann, P. Motlicek, Y. Qian, P. Schwarz, J. Silovsky, G. Stemmer, and K. Vesely, "The kaldi speech recognition toolkit," in *IEEE 2011 Workshop on Automatic Speech Recognition and Understanding*, 2011.

[28] F. Tong, M. Zhao, J. Zhou, H. Lu, Z. Li, L. Li, and Q. Hong, "Asv-subtools: Open source toolkit for automatic speaker verification," in *ICASSP 2021-2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2021, pp. 6184–6188.

[29] L. Liu, H. Jiang, P. He, W. Chen, X. Liu, J. Gao, and J. Han, "On the variance of the adaptive learning rate and beyond," in *ICLR 2020 : Eighth International Conference on Learning Representations*, 2020.

[30] https://www.pvtc2020.org/leaderboard.html.

[31] J. Hou, L. Zhang, Y. Fu, Q. Wang, Z. Yang, Q. Shao, and L. Xie, "The NPU system for the 2020 personalized voice trigger challenge," *arXiv preprint arXiv:2102.13552*, 2021.

[32] C. S. Greenberg, V. M. Stanford, A. F. Martin, M. Yadagiri, G. R. Doddington, J. J. Godfrey, and J. Hernandez-Cordero, "The 2012 nist speaker recognition evaluation." in *INTERSPEECH*, 2013, pp. 1971–1975.

[33] D. Snyder, G. Chen, and D. Povey, "Musan: A music, speech, and noise corpus," *arXiv preprint arXiv:1510.08484*, 2015.

[34] T. Ko, V. Peddinti, D. Povey, M. L. Seltzer, and S. Khudanpur, "A study on data augmentation of reverberant speech for robust speech recognition," in *2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2017, pp. 5220–5224.