CaTT-KWS: A Multi-stage Customized Keyword Spotting Framework based on Cascaded Transducer-Transformer

Zhanheng Yang[1,2], Sining Sun[2*], Jin Li[2], Xiaoming Zhang[2], Xiong Wang[1],Long Ma[2], Lei Xie[1†]*

[1]Audio, Speech and Language Processing Group, School of Computer Science, Northwestern Polytechnical University, Xi'an, China
[2]Tencent Technology Co., Ltd, Beijing, China

zhhyang@mail.nwpu.edu.cn,{siningsun,hughjli,xiaomizhang,malonema}@tencent.com,
xwang@npu-aslp.org, lxie@nwpu.edu.cn

Abstract

Customized keyword spotting (KWS) has great potential to be deployed on edge devices to achieve hands-free user experience. However, in real applications, false alarm (FA) would be a serious problem for spotting dozens or even hundreds of keywords, which drastically affects user experience. To solve this problem, in this paper, we leverage the recent advances in transducer and transformer based acoustic models and propose a new multi-stage customized KWS framework named Cascaded Transducer-Transformer KWS (CaTT-KWS), which includes a transducer based keyword detector, a frame-level phone predictor based force alignment module and a transformer based decoder. Specifically, the streaming transducer module is used to spot keyword candidates in audio stream. Then force alignment is implemented using the phone posteriors predicted by the phone predictor to finish the first stage keyword verification and refine the time boundaries of keyword. Finally, the transformer decoder further verifies the triggered keyword. Our proposed CaTT-KWS framework reduces FA rate effectively without obviously hurting keyword recognition accuracy. Specifically, we can get impressively 0.13 FA per hour on a challenging dataset, with over 90% relative reduction on FA comparing to the transducer based detection model, while keyword recognition accuracy only drops less than 2%.

Index Terms: Customized Keyword Spotting, Transducer, Transformer, Multi-stage detection, Multi-task learning

1. Introduction

Keyword spotting (KWS) aims at detecting predefined keywords from consecutive audio stream. It has significant applications on edge devices to realize hands-free user experience. Besides detecting a single wake-up word (WuW), e.g., "Alexa" and "Hi Google" to initiate cloud-based speech interactions, on-device speech command recognition is also desired for command-and-control and privacy-sensitive applications. In these applications, an always-on KWS system runs locally on the resource-limited edge device, and thus needs to be small-footprint, prompt and resistant to false alarm (FA) and false rejection (FR). Recently, neural network approaches [1, 2, 3, 4, 5, 6, 7] have been widely adopted. To ensure good performance, these approaches usually require training data for the specific sets of keywords, and the addition of a keyword to the current system requires a new round of data collection and model training. Moreover, with the increase of keywords for always-on applications, false alarms are notoriously hard to control. To

trade-off, a typical solution is to cascade a wake-up word detection module with a windowed command recognition module. Once the wake-up word is triggered, another keyword spotting module is activated for a brief time slot only, for accepting user commands. In this paper, we aims to develop a new customized KWS framework with high accuracy as well as rare FA. More importantly, it is highly customizable to new keywords with no more effort on data collection and model training.

With natural streaming ability and its success in speech recognition [8, 9, 10, 11], recurrent neural network transducer (RNN-T) has recently been applied to KWS tasks as well [12, 13, 14, 15]. Besides the streaming nature with low latency, transducer based acoustic model is flexible for open-vocabulary customized KWS as the modeling units can be sub-words, such as phonemes, which opens space for keyword customization. Some prior works on transducer based KWS aim to improve keyword accuracy by attention based biasing with pre-defined keyword transcripts [12, 13, 14, 16]. For example, He et al. [12] proposed a technique to bias the search process towards a specific keyword using an attention mechanism, while Liu et al. [13] further improved the attention based biasing and added auxiliary loss function during model training. Recently, Tian et al. [14] explored CTC joint training, stateless prediction network and various training data configuration strategies to avoid the over-fitting problem of transducer based KWS.

Although transducer based KWS is flexible for keyword customization, mainly through biasing, there is still plenty of space for further improvement. First, previous studies mainly considered the wake-up scenario with a single keyword as target at runtime, while its ability in detecting multiple keywords simultaneously for speech command recognition scenario has not been explored. Second, transducer based methods have obtained impressively high wake-up rate, but FA is also a severe problem reported in the literature [13, 14]. This problem might be even severe for speech command recognition scenario, aiming at supporting dozens or hundreds of keywords at runtime. Multi-stage strategy has been previously adopted to alleviate the problem [13, 17, 18, 19]. In general, the first stage is a lightweight always-on keyword detector. Once a keyword candidate is detected, the corresponding audio segment is sent to the following stage(s) for further verification. In the multi-stage architecture, transducer is a great choice for the first stage due to its high recall on keywords. But how to design the following verification stages for FA reduction becomes crucial for the performance of the whole system. In [13], which is also the most related work to our paper, a multi-level detection (MLD) method was proposed. The detection stage computes posterior sum confidence in a sliding window without regarding the phone order

---

to detect keyword with a small computational cost. Subsequent verification stages compute edit distance probability confidence and approximate likelihood confidence as measurements to further verify the keyword. The MLD method is a statistics based verification method only relies on the transducer output.

In this paper, we propose a new neural network based multi-stage customized KWS framework named Cascaded Transducer-Transformer KWS (CaTT-KWS), which includes a transducer based keyword detector, a frame-level phone predictor for force alignment and a transformer decoder. The three modules are shaped as a multi-task learning framework, where the encoder is shared across all the modules. Aiming at accurately spotting keyword candidates, the first stage adopts a tiny transducer [11] cascaded with a WFST based decoding graph, using context independent (CI) phones as modeling units. Meanwhile, a rough time boundary of the triggered candidate will be generated for the following stages. This detection stage can be customized easily to accept user preferred keywords by simply modifying the search graph, without retraining the model. It is well-known that the output of streaming transducer has emission delay problem. Therefore, in the second stage, Viterbi based force alignment is used to refine the keyword boundary and generate a likelihood score as confidence measure to decide if the audio segment generated in the detection stage includes a keyword or not. Lastly, a light-weight transformer decoder serves as the final verification stage, which accepts the encoder outputs corresponding to the triggered keyword and performs a beam search to finish the keyword verification. The proposed CaTT-KWS framework is verified on a challenging dataset, with impressively high keyword detection accuracy and low false alarms.

2. Multi-stage Framework

In this section, we introduce our proposed CaTT-KWS framework, including overview of the multi-task training procedure and detailed description on the design of the three stages – the transducer detector, the force-alignment module and the transformer decoder.

2.1. Multi-task learning procedure

We shape our multi-stage framework as a multi-task learning procedure, as shown in Fig. 1. The overall framework can be regarded as cascaded transducer-transformer with a shared encoder. Furthermore, additional linear layer is added as a framewise phone predictor to predict phone label for each frame. On one hand, predicting frame-level phone label can accelerate and regularize the encoder training. On the other hand, during inference, once keywords are detected, the output of phone predictor will be used to obtain more accurate keyword boundary by force alignment. More details will be given in Section 2.3. The final loss function is a combination of transducer loss, transformer loss and frame-level cross-entropy (CE) loss, described as:

$$L = \alpha L_{\text{Transducer}} + \beta L_{\text{CE}} + \gamma L_{\text{Transformer}} \qquad (1)$$

where α, β and γ are hyper-parameters. In this paper, we empirically set them to 1.0, 0.8 and 0.5 respectively. We performer multi-task learning during training and prepare corresponding transcript and frame-level phone alignment as label.

2.2. Detection Stage: Tiny Transducer

As shown in Fig. 1 (a), the transducer based detection stage consists of a tiny transducer based streaming acoustic model and a WFST based decoder. Note that this is similar to the hybrid framework [20, 21] which is widely adapted in KWS for keyword customization and restricting the search path. In this stage, the model tries to recall more cases on keywords, but it will inevitable result in a large number of false alarms.

Specifically, We adopt a tiny transducer proposed in [11] which consists of a DFSMN-based [22] encoder and a casual Conv1d based stateless predictor [23]. The tiny transducer was specifically designed for edge device deployment with tricks for model compression and strategies for avoiding model overfitting.

During inference, greedy search is used and only the posterior probabilities of non-blank outputs are fed into the WFST decoder, which is "skip blank" described in [11]. Our decoding graph is composed by two separate WFSTs: lexicons (L) and predefined command sets as grammars (G). They are composed into the final LG WFST for decoding, which can be presented as

$$LG = \min(\det(L \circ G)) , \qquad (2)$$

where min and det represent minimize and determinize operations, respectively. Token passing algorithm is used to figure out the most likely triggered keyword and outputs a sketchy time boundaries of the keyword as well.

2.3. Verification Stage: Force Alignment

In our second stage, we aim to refine the keyword boundaries obtained in the previous detection stage for subsequent stage and reject parts of false alarms.

After the detection stage, we can get the keyword candidate's sketchy time boundary and the corresponding phone sequence during the Viterbi decoding. Hence in the verification stage, we aim to make use of the phone predictor to get a more accurate time stamp for the candidate location and further verify it, as shown in Fig. 1(b). Because of the well-known emission delay issue of streaming transducer, the boundary obtained from the detection stage is not accurate and this will lead to possible truncation of the candidate. To refine the boundary, as illustrated in Fig 1(d), where t_0 is the original start point obtained from the transducer decoder, we push the start point empirically t_d frame backward and feed the corresponding frames to the force alignment module with the phone sequence of the keyword candidate.

To implement force alignment, a simple linear WFST graph is constructed using the phone sequence of triggered candidate. Note that because we push the start point t_d frames forward, which may introduce extra garbage segment, an extra symbol (g) is inserted before the phone sequence to absorb the outputs that do not belong to the candidate keyword. Fig 1(d) gives an keyword candidate example composed of phone sequence "a, b, c" with extra garbage "(g)". Finally we get the accurate start point t_r of the candidate and the likelihood calculated during alignment is considered as the confidence score S_1 for this stage:

$$S_1 = \frac{-\sum_{l \in f, t \in T} \log(p_l^t)}{T} , \qquad (3)$$

where f denotes the framewise force alignment result and p_l^t denotes the phone predictor posterior of label l at t^{th} frame. And T is the accurate time stamp we get at this stage. Comparing S_1 with a pre-defined threshold τ can reject part of the false alarm examples effectively and positive examples with accurate boundaries are passed to the final verification stage.
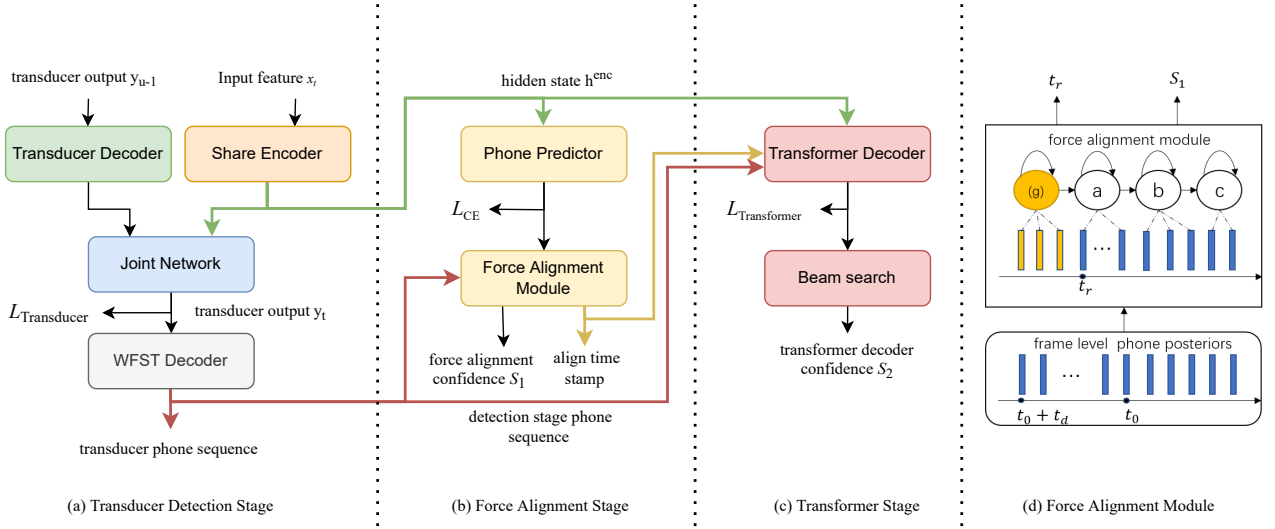
Figure 1: *The proposed CaTT-KWS framework, which is composed of three stages – transducer stage (a) for keyword detection and force alignment stage (b) as well as transformer stage (c) for keyword verification. Given the frame level phone posteriors generated by the phone predictor, the time stamp of the keyword candidate is further refined with more accurate start time stamp t_r and likelihood score S_1, as shown in (d).*

2.4. Verification Stage: Transformer Decoder

In the final stage, we further verify the trigger command segment and make the final decision. After the above force alignment stage, more accurate boundaries of keyword candidate are obtained, with which the shared encoder's outputs h^{enc} of the corresponding keyword candidate can be truncated out, which serves subsequently as the inputs of the transformer decoder, as shown in Fig. 1(c). The transformer decoder accepts h^{enc} and generates an N-best list autoregressively through beam search. Since we aim to confirm whether the segment is a real keyword or not, we do not need to search until <EOS> appears. Therefore, we force to stop the beam search procedure after M steps, where M is the length of the phone sequence of the keyword candidate. After decoding, we check if there is any output sequence that matches the phone sequence of the candidate. If none, it is rejected; otherwise, we compare score S_2 of the sequence to a preset threshold v. S_2 can be described as

$$S_2 = -\sum_{l \in B} \log(p_l) , \qquad (4)$$

where B denotes the triggered phone sequence in the beam. If S_2 is smaller than the threshold v, the candidate is eventually triggered as a keyword.

3. Experiments

In this section, we introduce the corpus we used and describe the experimental setup about the detail of our model configuration and evaluation metrics. Experimental results and analysis are also presented at last.

3.1. Corpus

Our models are trained on a set of 23,000-hour Mandarin ASR corpus which is collected from Tencent in-car speech assistant products. During training, the development set is randomly shuffled from the training set. We evaluated the accuracy of all models on a 6K utterances keyword set that cover 29 Mandarin commands (each has six or more phones), such as a command for starting GPS navigation that is consisted of phone sequence "d a3 k ai1 d ao3 h ang2". This 6K testing utterances are collected from both relatively 'clean' highway driving conditions and noisy downtown driving conditions, resulting in the *clean*

set with 3K utterances and the *noisy* set with another 3K utterances. For false alarm evaluation, we used an 84-hour audio set covering radio broadcast, chat and driving noise. All data we mentioned above are anonymous and hand-transcribed.

3.2. Experimental Setup

We used the 40-dim power-normalized cepstral coefficients (PNCC) [24] feature computed with a 25ms window and shifted every 10ms as input for all experiments. For the model configuration of our proposed model, the shared encoder consists of a convolution downsample layer to achieve a time reduction rate of 4 and a simple 1-layer 128-dim LSTM following a 6-layer 512-dim FSMN with a 320-dim linear projection. The left context of the FSMN module is 8 for all layer and right context is [2,2,1,2,2,1], respectively. The transducer decoder we used has a 1-D convolution layer with kernel size is 2. The joint network consists of a single feed-forward layer with 100 units following a hyperbolic tangent activation function. The output units include 210 context-independent (CI) phones and a blank symbol. The phone predictor only has one linear layer and a softmax layer, which maps the encoder outputs to phones. The transformer decoder includes several 512-dim self-attention blocks and we further explore the effect of the self-attention block's number in Section 3.4. The overall model size is about 3.8M, which is suitable for deploying on edge devices.

3.3. Effect of Two Verification Stages

Table 1 and Fig. 2 show the results of our proposed framework and MLD [13] method on clean and noisy evaluation sets. The force alignment stage threshold τ and transformer stage threshold v are set as 1.5 and 5, respectively. And we set t_d as 15 during the force alignment stage. The transformer decoder only has one attention block. From experiment S0 in Table 1, we find that in the detection stage, the accuracy rate of the tiny transducer can reach 96% on the clean test set, which is pretty high. However, it also reaches a relative high FA at 1.47 per hour. By using our proposed force alignment and transformer stages, as shown in experiment S3, the FA per hour drops from 1.47 to 0.13 with little accuracy decay. Besides, we used the experiment setup suggested in [13] to evaluate the MLD method, and our proposed method can get a much lower FA rate with similar detection accuracy.

Table 1: *Comparison on accuracy(%) and FA number per hour among different framework.*

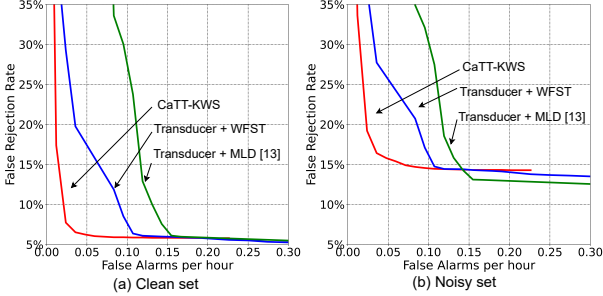| ID	Stage	Acc (%)		FA	
		Clean	Noisy	per hour	Gain (%)
S0	Transducer + WFST	96	87.65	1.47	-
S1	+ Transformer decoder	94.31	85.89	0.23	84.35
S2	+ Force alignment	95.48	87.34	0.5	65.98
S3	++ Transformer decoder	94.20	85.69	0.13	91.15
S4	Transducer + MLD	94.41	86.24	0.27	-

Figure 2: *ROC curves for different systems.*

Fig 2 shows the receiver operating characteristic (ROC) curves of our proposed CaTT-KWS framework and the MLD method on the clean set and noisy set. During decoding, the threshold of transformer stage is fixed and threshold of force alignment stage changes dynamically. For the MLD method, only the threshold of last stage is changed to draw ROC curves. It is obvious that our proposed method outperforms the MLD method significantly when the number of FA (per hour) is less than 0.15, at which the FRR of MLD method worsens rapidly. Moreover, as shown in Table 1, accurate time boundaries of speech command can improve the performance of transformer decoder based verification comparing the result between the experiment of S1 (two stages framework) and S3 (three stages framework). In order to further analyze the result, we compare the average timestamp offset compared with the ground truth of start point for the command phrase obtained by transducer detection stage and force alignment stage. As shown in Table 2, the timestamp generated from force alignment module is more accurate. And a accurate timestamp is very effective for reducing the number of FA

3.4. Impact of Transformer Decoder Size

To explore how transformer decoder size can affect performance, we adjust the transformer decoder size by adding more self-attention blocks. The ROC results are reported in Fig. 3. It denotes that adding transformer decoder size can not gain improvement on accuracy with similar FA levels while increasing the overall model size. The recognition performance even gets worse at low FA level because the overfitting of the model will be aggravated due to the use of more model parameters.

3.5. Impact of Different Transformer Modeling Units

Considering the scalability of our model (e.g. customized commands), we choose CI phones as the model units of both transducer and transformer decoder in the experiments above. Here we further use 6,000 common Chinese characters as the modeling unit for the transformer decoder and make a comparison with the counterpart using CI phones. The ROC curves in Fig. 4 show that the phone model has a lower FR rate at lower FA conditions while the character model leads to an even lower FR rate at higher FA conditions. As a result, we can select appro-

Table 2: *Comparison on start point error between transducer detection stage and force alignment stage.*

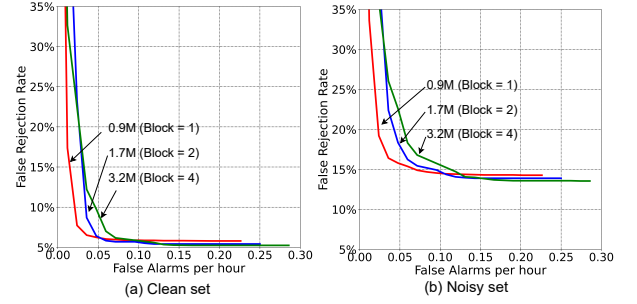| Stage	Clean set (s)	Noisy set (s)
Transducer detection	0.29	0.44
Force alignment	0.11	0.23

Figure 3: *ROC curves for systems with different transformer decoder size.*

priate modeling units according to different requirements, such as using modeling units with larger granularity like character in scenarios that require lower FRR. On the contrary, we can use units with smaller granularity like phone in scenarios that should pay more attention to FA. By the way, because a larger number of model units result in bigger input and output layers, the size of transformer decoder of the character system becomes 4 times larger than that of the phone system. In order to effectively avoid the increase of model size, we can discard the unused input and output nodes according to the phones/characters covered in the predefined keyword set.
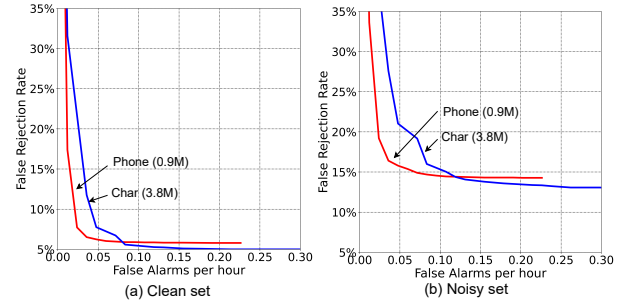

Figure 4: *ROC curves for character- and phone- systems. The transformer decoder size is shown in parentheses.*

4. Conclusions

In this paper, we proposed a new KWS framework CaTT-KWS, aiming to suppress FA without obviously reducing command recognition accuracy. The model size of CaTT-KWS is about 3.8M, which is small enough for deploying on edge devices. Our proposed framework is shaped as the cascaded transducer-transformer architecture with the force alignment module. The force alignment module can also provide a more accurate time stamp. We evaluate our method on a sizable dataset, showing considerable relative reduction of over 90% on FA, which achieves about 0.13 FA per hour, while the command recognition accuracy only drops less than 2%. And we further set up experiments to show the performance of our proposed CaTT-KWS framework, and explore the impact on transformer decoder size and model units as well.

5. References

[1] T. Sainath and C. Parada, "Convolutional neural networks for small-footprint keyword spotting," 2015.

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

[3] B. Wei, M. Yang, T. Zhang, X. Tang, X. Huang, K. Kim, J. Lee, K. Cho, and S.-U. Park, "End-to-end transformer-based open-vocabulary keyword spotting with location-guided local attention," pp. 361–365, 2021.

[4] C. Jose, Y. Mishchenko, T. Senechal, A. Shah, A. Escott, and S. Vitaladevuni, "Accurate detection of wake word start and end using a cnn," *arXiv preprint arXiv:2008.03790*, 2020.

[5] A. Berg, M. O'Connor, and M. T. Cruz, "Keyword transformer: A self-attention model for keyword spotting," *arXiv preprint arXiv:2104.00769*, 2021.

[6] J. Qi and J. Tejedor, "Exploiting hybrid models of tensor-train networks for spoken command recognition," *arXiv preprint arXiv:2201.10609*, 2022.

[7] J. Bae and D.-S. Kim, "End-to-end speech command recognition with capsule network." in *Interspeech*, 2018, pp. 776–780.

[8] A. Graves, "Sequence transduction with recurrent neural networks," *arXiv preprint arXiv:1211.3711*, 2012.

[9] Y. He, T. N. Sainath, R. Prabhavalkar, I. McGraw, R. Alvarez, D. Zhao, D. Rybach, A. Kannan, Y. Wu, R. Pang *et al.*, "Streaming end-to-end speech recognition for mobile devices," in *IEEE International Conference on Acoustics, Speech and Signal Processing*. IEEE, 2019, pp. 6381–6385.

[10] T. N. Sainath, Y. He, B. Li, A. Narayanan, R. Pang, A. Bruguier, S.-y. Chang, W. Li, R. Alvarez, Z. Chen *et al.*, "A streaming on-device end-to-end model surpassing server-side conventional model quality and latency," in *IEEE International Conference on Acoustics, Speech and Signal Processing*. IEEE, 2020, pp. 6059–6063.

[11] Y. Zhang, S. Sun, and L. Ma, "Tiny transducer: A highly-efficient speech recognition model on edge devices," in *IEEE International Conference on Acoustics, Speech and Signal Processing*. IEEE, 2021, pp. 6024–6028.

[12] 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*. IEEE, 2017, pp. 474–481.

[13] Z. Liu, T. Li, and P. Zhang, "Rnn-t based open-vocabulary keyword spotting in mandarin with multi-level detection," in *IEEE International Conference on Acoustics, Speech and Signal Processing*. IEEE, 2021, pp. 5649–5653.

[14] Y. Tian, H. Yao, M. Cai, Y. Liu, and Z. Ma, "Improving rnn transducer modeling for small-footprint keyword spotting," in *IEEE International Conference on Acoustics, Speech and Signal Processing*. IEEE, 2021, pp. 5624–5628.

[15] E. Sharma, G. Ye, W. Wei, R. Zhao, Y. Tian, J. Wu, L. He, E. Lin, and Y. Gong, "Adaptation of rnn transducer with text-to-speech technology for keyword spotting," in *IEEE International Conference on Acoustics, Speech and Signal Processing*. IEEE, 2020, pp. 7484–7488.

[16] T. Bluche and T. Gisselbrecht, "Predicting detection filters for small footprint open-vocabulary keyword spotting," *arXiv preprint arXiv:1912.07575*, 2019.

[17] S. Sigtia, P. Clark, R. Haynes, H. Richards, and J. Bridle, "Multi-task learning for voice trigger detection," 2020.

[18] R. Yang, G. Cheng, H. Miao, T. Li, P. Zhang, and Y. Yan, "Keyword search using attention-based end-to-end asr and frame-synchronous phoneme alignments," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 29, pp. 3202–3215, 2021.

[19] S. Sigtia, J. Bridle, H. Richards, P. Clark, E. Marchi, and V. Garg, "Progressive voice trigger detection: Accuracy vs latency," in *IEEE International Conference on Acoustics, Speech and Signal Processing*. IEEE, 2021, pp. 6843–6847.

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

[21] M. Sun, D. Snyder, Y. Gao, V. K. 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.

[22] S. Zhang, M. Lei, Z. Yan, and L. Dai, "Deep-fsmn for large vocabulary continuous speech recognition," in *IEEE International Conference on Acoustics, Speech and Signal Processing*. IEEE, 2018, pp. 5869–5873.

[23] M. Ghodsi, X. Liu, J. Apfel, R. Cabrera, and E. Weinstein, "Rnn-transducer with stateless prediction network," in *IEEE International Conference on Acoustics, Speech and Signal Processing*. IEEE, 2020, pp. 7049–7053.

[24] C. Kim and R. M. Stern, "Power-normalized cepstral coefficients (pncc) for robust speech recognition," *IEEE/ACM Transactions on audio, speech, and language processing*, vol. 24, no. 7, pp. 1315–1329, 2016.