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

FSR: Accelerating the Inference Process of Transducer-Based Models by Applying Fast-Skip Regularization

Zhengkun Tian[1,2], *Jiangyan Yi*[1], *Ye Bai*[1,2], *Jianhua Tao*[1,2,3], *Shuai Zhang*[1,2], *Zhengqi Wen*[1]

[1]NLPR, Institute of Automation, Chinese Academy of Sciences, Beijing, China
[2]School of Artificial Intelligence, University of Chinese Academy of Sciences, Beijing, China
[3] CAS Center for Excellence in Brain Science and Intelligence Technology, Beijing, China
{zhengkun.tian, jiangyan.yi, ye.bai, jhtao, shuai.zhang, zqwen}@nlpr.ia.ac.cn

Abstract

Transducer-based models, such as RNN-Transducer and transformer-transducer, have achieved great success in speech recognition. A typical transducer model decodes the output sequence conditioned on the current acoustic state and previously predicted tokens step by step. Statistically, The number of blank tokens in the prediction results accounts for nearly 90% of all tokens. It takes a lot of computation and time to predict the blank tokens, but only the non-blank tokens will appear in the final output sequence. Therefore, we propose a method named fast-skip regularization, which tries to align the blank position predicted by a transducer with that predicted by a connectionist temporal classification (CTC) model. During the inference, the transducer model can predict the blank tokens in advance by a simple CTC project layer without many complicated forward calculations of the transducer decoder and then skip them, which will reduce the computation and improve the inference speed greatly. All experiments are conducted on a public Chinese mandarin dataset AISHELL-1. The results show that the fast-skip regularization can indeed help the transducer model learn the blank position alignments. Besides, the inference with fast-skip can be speeded up nearly 4 times with only a little performance degradation.

Index Terms: Transducer-based Models, Speech Recognition, Fast-Skip Regularization, CTC model

1. Introduction

Transducer-based models [1, 2, 3, 4], such as the RNN-Transducer (RNN-T) [5, 6, 7] and Transformer-Transducer (T-T) [8, 9, 10], have been widely applied for speech recognition. Compared with the attention-based encoder-decoder models [11, 12, 13, 14, 15], the transducer-based models predict a new token conditioned on the previous output tokens and the current encoded frame rather than the whole acoustic encoded sequence, which make it can be directly applied for streaming speech recognition. Different from the CTC model [16, 17, 18], transducer-based models can model the linguistic dependencies between the output tokens.

A typical transducer-based model consists of three components, an acoustic encoder (also named transcription network), a linguistic predictor, and a joint network, as shown in Fig.1. As the same as the CTC models, the transducer models adopt the forward-backward algorithm to optimize all possible alignment paths and decode the output sequence frame by frame. This characteristic affects the inference speed of the transducer models from two aspects. On the one hand, the frame-wise operation forces the transducer to pass through the transducer decoder (including the linguistic predictor and the joint network)

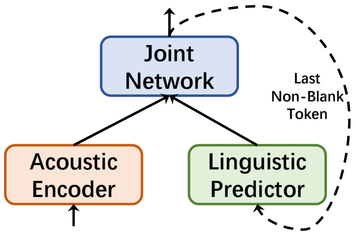


Figure 1: *The Architecture of Transducer-Based Models*

at each step of inference, which leads to a lot of calculations. On the other hand, due to the fact that the frame-level output probability lattice contains a lot of alignment paths with the same prefixes, the transducer models have to merge the probability of these duplicated paths. To our knowledge, there are four kinds of methods to address these two problems. Firstly, the neural transducer [19] and sync-transformer [20] split the acoustic encoded states into some fixed-length chunk and predict the output sequence chunk by chunk, which has made a significant change in the structure of the transducer model. Secondly, the neural aligner [21, 22, 23] forces the model to decode at most one token based on one acoustic frame, which simplifies the output probability lattice of both the training and inference but it still adopts frame-by-frame inference. Thirdly, [24] also directly applied the one-step constraint to the inference of transducer-based model, which is a simple and limited method. Finally, a recently proposed model named State-Less RNN-T [25, 26] utilizes a simple embedding layer instead of the traditional multiple RNN layer as the linguistic predictor, which improves the inference speed by reducing the computation of predictor. However, this method also makes it difficult for the transducer to capture the long-range linguistic dependencies.

We notice that the transducer model will produce a lot of blank tokens during inference. The blank token stands for silence or duplicated token, which will not appear in the final predicted result. The prediction of each token must depend on the forward calculation of the transducer decoder. Furthermore, we find the predicted blank tokens account for over 90% of all the tokens. Based on this observation, we assume that the inference process can be speeded up by predicting the blank tokens in advance with some simple computation and then skipping them. In this paper, we propose a novel method named fast-skip regularization (FSR) to make the transducer model depend on a CTC project layer to predict the blank tokens in advance. The FSR introduces a CTC module into the transducer model. And it forces the transducer model to align the blank position with that predicted by the CTC project layer. During the inference, the transducer model can skip the blank token based on the prediction of the CTC project layer, which makes the output probability lattice be sparse and improve the inference speed greatly. All the experiments are conducted on a public Chinese mandarin dataset AISHELL-1. The results show that the FSR can align the blank position predicted by the CTC and transducer accurately. The inference process can be accelerated up to 4 times with only a little performance degradation.

2. Fast-Skip Regularization for Transducer-based Models

2.1. Transducer

Transducer models can transcribe the acoustic feature sequence into the target sequence directly. As the same as the CTC model, the transducer introduces a blank token $\varnothing$ to represent the duplicated tokens and silence. Given an input sequence $\boldsymbol{x} = (x_1, x_2, ..., x_T)$ with length T and the target sequence $\boldsymbol{y} = (y_1, y_2, ..., y_U)$ with length U, the transducer aims to maximize the log-probability $P(\boldsymbol{y}|\boldsymbol{x})$:

$$\mathcal{L}_{transducer} = -\ln P(\boldsymbol{y}|\boldsymbol{x}) = -\ln \sum_{\pi \in \mathcal{B}^{-1}(\boldsymbol{y})} P(\pi|\boldsymbol{x}) \quad (1)$$

where π indicates any one alignment path and $\mathcal{B}_{-1}(\boldsymbol{y})$ represents the space which contains all possible alignment path, as shown in Fig.2(a). It is very challenging to calculate $P(\boldsymbol{y}|\boldsymbol{x})$ by enumerating all possible alignment paths. Therefore, the transducer introduces an efficient forward-backward algorithm to optimize all possible alignment paths [1, 27].

The *forward variable* $\alpha(t, u)$ means the sum of the probability of all the possible paths, which begins with the start token y_0 $(= \varnothing)$ and ends with y_u at the t-th frame. Given the t-th frame and the predicted token sequence $y_{0:u-1}$, the probabilities of predicting $\varnothing$ and y_u are represented as $\varnothing(t, u)$ and $y(t, u)$ respectively. For all $1 \leq t \leq T$ and $0 \leq u \leq U$, the forward variables can be calculated recursively using

$$\alpha(t, u) = \alpha(t-1, u)\varnothing(t, u) + \alpha(t, u-1)y(t, u-1) \quad (2)$$

And all paths begin with y_0 $(= \varnothing)$, it means $\alpha(1, 0) = 1$. The conditional probability $P(\boldsymbol{y}|\boldsymbol{x})$ can be expressed by the forward variable at the terminal node.

$$P(\boldsymbol{y}|\boldsymbol{x}) = \alpha(T, U)\phi(T, U) \quad (3)$$

Similarly, the *backward variable* $\beta(t, u)$ means the sum of the probabilities of all possible paths, which begin with y_u at t-th frame and end with y_U(=$\varnothing$) in the last frame. The backward variables can be expressed as

$$\beta(t, u) = \beta(t+1, u)\varnothing(t, u) + \beta(t, u+1)y(t, u) \quad (4)$$

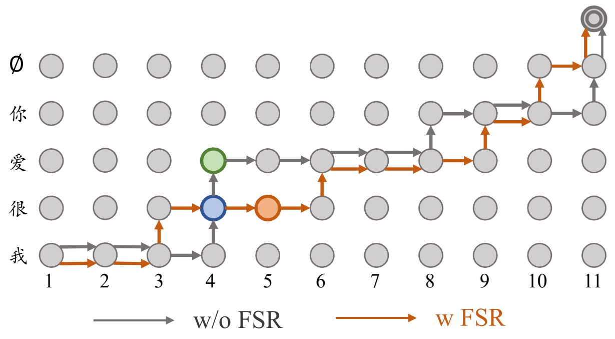
where the initial condition $\beta(T, U) = \varnothing(T, U)$.

Given an input feature sequence $\boldsymbol{x}$ and a target sequence $\boldsymbol{y}$, the probability $p(\boldsymbol{y}|\boldsymbol{x})$ is equal to the sum of $\alpha(t, u)\beta(t, u)$ over any top-left to bottom-right diagonal through the nodes. That is, $\forall n : 1 \leq n \leq U + T$

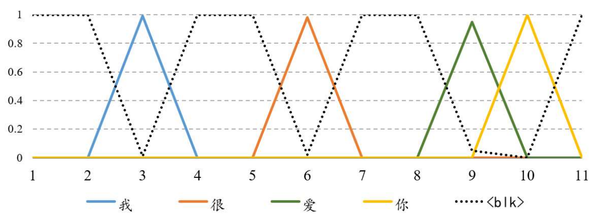
$$P(\boldsymbol{y}|\boldsymbol{x}) = \sum_{(t,u):t+u=n} \alpha(t, u)\beta(t, u) \quad (5)$$

2.2. Fast-Skip Regularization

We try to make the transducer model able to predict the blank tokens $\varnothing$ in advance without depending on the forward of the transducer decoder. Therefore, we propose a method named



(a) The Output Probability Lattice of Transducer-based Models



(b) The Probability Spikes of CTC-based Models

Figure 2: *Illustration of Fast-Skip Regularization. (a) indicates the output probability lattice of the transducer-based model. The gray transfer path means the alignment sequence without FSR. The orange transfer path means our expected alignment sequence with FSR. The symbol $\varnothing$ means the blank token. (b) shows the spikes graph generate by the CTC project layer. The FSR will force the vertical non-blank transfer of the transducers to align with the non-blank spikes generated by the CTC models. And the horizontal transfers are consistent with the blank positions of the CTC model.*

Fast-Skip Regularization (FSR), which forces the transducer model to align the predicted blank position with the blank position of the CTC model.

Inspired by the FastEmit [28, 29, 30], we utilize FSR to adjust the probability of possible alignment paths on the output lattice of the transducer model. We define a variable $\mathcal{A}_{t,u}$ to represent all the possible alignment paths that pass thought the *node* (t, u), as the blue node shown in Fig.2(a). The conditional probability $P(\mathcal{A}_{t,u}|\boldsymbol{x})$ can be decomposed into two terms:

$$\begin{aligned} &P(\mathcal{A}_{t,u}|\boldsymbol{x}) = \alpha(t, u)\beta(t, u) \\ &= P(\mathcal{A}^{nb}_{t,u}|\boldsymbol{x}) + P(\mathcal{A}^{b}_{t,u}|\boldsymbol{x}) = \\ &\underbrace{\alpha(t, u)y(t, u)\beta(t, u+1)}_{\text{predict non-blank token}} + \underbrace{\alpha(t, u)\varnothing(t, u)\beta(t+1, u)}_{\text{predict blank token}} \end{aligned} \quad (6)$$

where the first term $P(\mathcal{A}^{nb}_{t,u}|\boldsymbol{x})$ indicates the probability of all the alignment path transferred from the blue cirle to the green circle (vertical direction) and the second term $P(\mathcal{A}^{b}_{t,u}|\boldsymbol{x})$ means the probability of all the alignment paths that transferred from the bule circle to the orange circle (horizontal direction).

Based on the observation that the transducer loss aims to maximize the log-probability regardless of their emission position [28], we try to make the blank token positions predicted by these two models be consistent. We introduce a CTC module into the transducer model by putting a project linear layer at the top of the acoustic encoder. Afterwards, we define the blank probability predicted by the CTC module based on the t-th frame as $\mathcal{C}^b_t$, as shown by the black dot line in Fig.2(b).

Similarly, $\mathcal{C}_t^{nb}$, as shown by the colorful spikes in Fig.2(b), indicates the non-blank probability. And there is the following equation relationship between them: $\mathcal{C}_t^{nb} = 1 - \mathcal{C}_t^b$. We introduce the token position information of the CTC model into the training process of transducer. For $\forall n : 1 \leq n \leq U + T$, there is

$$\begin{aligned}\mathcal{L}_{transducer+fsr} &= \mathcal{L}_{transducer} + \mathcal{L}_{fsr} \\ &= -\ln \sum_{(t,u):t+u=n} [P(\mathcal{A}_{t,u}|\boldsymbol{x}) \\ &+ \lambda \mathcal{C}_t^{nb} P(\mathcal{A}_{t,u}^{nb}|\boldsymbol{x}) + \lambda \mathcal{C}_t^b P(\mathcal{A}_{t,u}^b|\boldsymbol{x})]\end{aligned} \tag{7}$$

where λ is the FSR weight utilized to balance the regularization term and transducer loss. The second term $\mathcal{C}_t^{nb} P(\mathcal{A}_{t,u}^{nb}|\boldsymbol{x})$ encourages the transducer model to predict the non-blank tokens (vertical transfer in the Fig.2(a)) at the spike position. The third term $\mathcal{C}_t^b P(\mathcal{A}_{t,u}^b|\boldsymbol{x})$ encourages the transducer model to learn the blank position predicted by the CTC model (horizontal transfer in the Fig.2(a)). The gradients of transducer model with respect to blank and non-blank prediction can be further rewritten as

$$\begin{aligned}\frac{\partial \mathscr{L}}{\partial P(k = y_u|t,u)} &= -\frac{(1 + \lambda \mathcal{C}_t^{nb})\alpha(t,u)\beta(t,u+1)}{P(\boldsymbol{y}|\boldsymbol{x})} \\ \frac{\partial \mathscr{L}}{\partial P(k = \varnothing|t,u)} &= -\frac{(1 + \lambda \mathcal{C}_t^{b})\alpha(t,u)\beta(t+1,u)}{P(\boldsymbol{y}|\boldsymbol{x})}\end{aligned} \tag{8}$$

During the training process, the FSR regularization will make the transducer models tend to choose the possible alignment paths matching with the position prior knowledge from the CTC module. Now that we introduce an external CTC project layer, the final joint loss function can be expressed as:

$$\mathcal{L}_{joint} = \mathcal{L}_{CTC} + \mathcal{L}_{transducer+fsr} \tag{9}$$

2.3. The Fast-Skip Inference

The traditional transducer models decode the output sequence frame by frame along the time axis. At each decoding step, the linguistic predictor and joint network can be calculated once respectively, which takes a lot of time and resources. The FSR can assist the transducer model to learn how to predict the blank tokens at the same positions as the CTC model. This makes the transducer model can predict the blank tokens in advance and skip them.

During the inference, the fast-skip transducer first computes the blank probability $\mathcal{C}_t^b$ by the CTC project layer. If the $\mathcal{C}_t^b$ is great than the skip-trigger threshold δ, the t-frame will be skipped. Otherwise, the t-th frame will be marked as triggered and the transducer model will continue to decode based on the current acoustic encoded frame. In order to avoid the deletion error caused by the misalignment of the predicted blank tokens, we set a fixed-length spike-window (W_{left}, W_{right}) based on the left and right frames of the triggered position. The W_{left} and W_{right} is the number of the left and right frames. The operation of expanding will sacrifice the inference speed to keep the performance degradation of the model as small as possible.

3. Experiments and Results

3.1. Dataset

In this work, all experiments are conducted on a public Mandarin speech corpus AISHELL-1[1]. The training set contains about 150 hours of speech (120,098 utterances) recorded by 340 speakers. The development set contains about 20 hours (14,326 utterances) recorded by 40 speakers. And about 10 hours (7,176 utterances / 36109 seconds) of speech is used as the test set. The speakers of different sets are not overlapped.

[1]https://openslr.org/33/

3.2. Experimental Setup

For all experiments, we use 80-dimensional Mel-filter bank coefficients (FBank) features computed on a 25ms window with a 10ms shift. Each feature is re-scaled to have zero mean and unit variance for each speaker. We chose 4233 characters (including a blank token '∅', a unknown token `<UNK>` and a padding token `<PAD>`) as model units.

In this paper, we only compared the experiments based on the transformer-transducer(T-T) model [8]. The T-T consists of 12 encoder blocks and 6 decoder blocks. There are 4 heads in multi-head attention. Both the output size of the multi-head attention and the feed-forward layers are 320. The hidden size of feed-forward layers is 1280. The 2D convolution front end utilizes two-layer time-axis CNN with ReLU activation, stride size 2, channels 320, and kernel size 3. The joint network contains an acoustic project linear layer, a linguistic project linear layer, and an output linear layer with input size 640. We adopt an Adam optimizer with warmup steps 12000 and the learning rate scheduler reported in [31]. After 150000 epochs, we average the parameters saved in the last 20 checkpoints. We also use the time-mask and frequency-mask method proposed in [32] instead of speed perturbation. We just apply the simple greedy search method for the following experiments. The more results based on other decoding methods will be reported in the future work. Besides, we did not utilize the external language models.

We use the character error rate (CER) to evaluate the performance of different models. For evaluating the inference speed of different models, we decode utterances one by one to compute real-time factor (RTF) on the test set. The RTF is the time taken to decode one second of speech. All experiments are conducted on a GeForce RTX 2080Ti 12G GPU.

3.3. Results

3.3.1. *Comparison of the Models with Different Weight λ*

Table 1: *Comparison of the Models with Different Weight λ (CER %).*

No.	FSR Weight λ	Dev	Test
1	0.000	13.18	13.95
2	0.001	7.56	8.04
3	0.003	7.07	7.78
4	0.005	6.95	7.52
5	0.01	**6.80**	**7.36**
6	0.02	6.91	7.58

In this section, we compare the models trained with different FSR weights λ. All the evaluated models set skip-trigger threshold δ to 0.5 and spike-window to *(1, 1)*. The results show that the model with an FSR weight of 0.01 (No.5) can achieve the best performance on both the development and test set. When the weight λ is equal to 0 (No.1), the regularization is abandoned, which is equivalent to utilize a multi-task training method to accelerate the convergence of the transducer model. Under this condition, directly decoding with fast-skip inference may make the model mistakenly skip some key frames and lead to large performance degradation. In the following experiments, we adopt the FSR weight 0.01 as the default parameter.

3.3.2. *Comparison of the Length of Expanding Window*

Even applying the fast-skip method, we can't guarantee that the transducer model can learn the blank alignment accurately. We

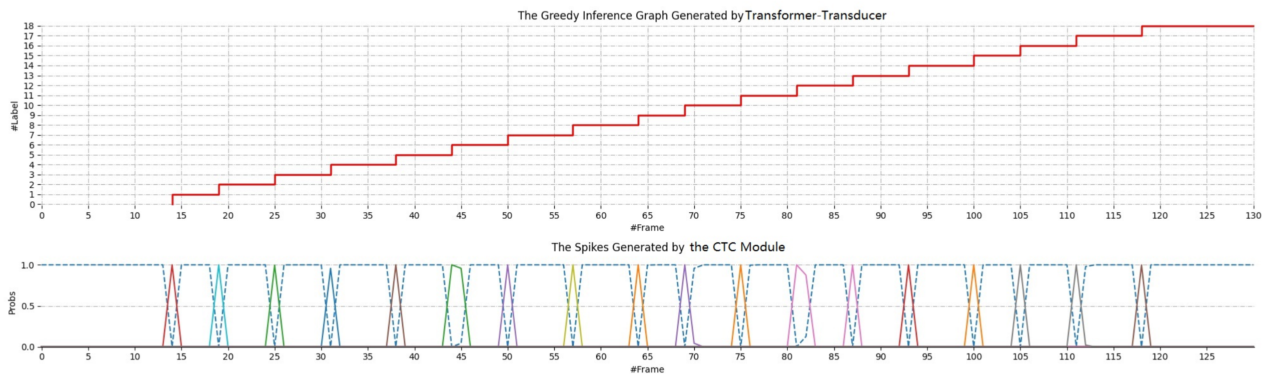


Figure 3: *The Visualization of Alignment Relationship between the Transducer Model and CTC Model. The upper figure captures the most possible alignment path of the transformer-transducer during the greedy search. The lower one is the spikes graph predicted by the CTC module. The spikes with different colors stand for different tokens. The blue dot line indicates the blank probability.*

construct a fixed-length window around the trigger spike, and all the acoustic frames in the window will be triggered, which could avoid missing some key frames due to the misalignment. W_{left} and W_{right} means the number of left and right frames around the triggered position respectively. As shown in Table.2, the expanding window is indeed able to improve the model performance. When the window width reaches a certain level, the performance of the model is no longer improved, which also confirms that the FSR could make the blank position predicted by these two models close. As the window becomes wider, the computation of the model increases gradually, which leads to the decrease of the inference speed.

Table 2: *Comparison of the Length of Expanding Window (CER %).*

No.	W_{left}	W_{right}	Dev	Test	RTF
1	0	0	7.03	7.75	0.0352
2	1	0	6.92	7.50	0.0473
3	0	1	6.90	7.49	0.0480
4	1	1	6.80	7.36	0.0601
5	2	1	6.64	7.21	0.0738
6	1	2	6.64	7.20	0.0727

3.3.3. Ablation Study

Table 3: *Ablation Study on the Inference Process (CER %).*

Model	No.	FSR	FSI	Dev	Test	RTF
T-T	A	✔	✔	6.80	7.36	0.0601
	B	✔	✗	6.40	7.15	0.1983
	C	✗	✔	13.18	13.95	0.0615
	D	✗	✗	6.32	7.12	0.2122

This section focuses on the importance of different components. All the evaluated models with Fast-Skip inference(FSI) set skip-trigger threshold δ to 0.5 and spike-window to *(1, 1)*. In order to facilitate the description, we mark the above four models as `A`, `B`, `C` and `D` respectively. Comparing model `A` with `B`, we find that the FSI could improve the inference speed nearly 4 times although he also brings a little performance degradation. The performance degradation can be alleviated by enlarging the window in the inference process. In the comparison of model `A` and `C`, it's easy to observe that the transducer model without FSR has a great performance degradation, which also proves that the fast-skip regularization is effective. The model `D` trained jointly with a CTC module can be regarded as a baseline model. Compared with the model `D`, the transducer models with FSR (`A` and `B`) just have a little performance degradation.

3.3.4. The Visual analysis of Alignment

To further study the impact of the FSR, we draw the picture as shown in Figure.3, which captures the positional alignment relationship between the transducer model and the CTC model. There is a frame-by-frame correspondence between these two subgraphs. We notice that the predicted alignment path in the upper part of Fig.3 starts at the 15-th acoustic frame. And the CTC model predicts a spike based on the same frame. It is no coincidence that each vertical shift of the T-T corresponds to a spike predicted by the CTC module. This proves our assumption that the transducer model could skip the blank position predicted by the CTC model in advance to reduce the computation and accelerate the inference.

4. Conclusions and Future Works

A typical transducer model decodes the output sequence conditioned on the current acoustic state and previously predicted tokens step by step. Statistically, the number of blank tokens in the prediction results accounts for nearly 90% of all tokens. It takes a lot of computation and time to predict the blank label, but it will not appear in the final output sequence. Consequently, we assume that the inference process of the transducer can be accelerated greatly if the blank tokens can be predicted in advance just depending on simple computation, and then skipped. This paper proposes a method named fast-skip regularization, which tries to align the blank position predicted by a transducer with that predicted by a CTC model. During the inference, the transducer model can skip the frames predicted by a simple CTC project layer as blank tokens. All experiments are conducted on a public Chinese mandarin dataset AISHELL-1. The results show that the fast-skip regularization can indeed help the transducer model learn the blank position predicted by the CTC module. The inference with fast-skip can be speeded up nearly 4 times with only a little performance degradation. In the future, we will focus on how to accelerate the training and inference simultaneously by applying the fast-skip regularization. We also provide more results based on the different inference methods and different dataset.

5. Acknowledge

This work is supported by the National Key Research & Development Plan of China (No. 2018YFB1005003), the National Natural Science Foundation of China (NSFC) (No.61831022, No.61901473, No.61771472, No.61773379), and Inria-CAS Joint Research Project (No.173211KYSB20190049).

6. References

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

[2] K. Rao, H. Sak, and R. Prabhavalkar, "Exploring architectures, data and units for streaming end-to-end speech recognition with rnn-transducer," in *2017 IEEE Automatic Speech Recognition and Understanding Workshop (ASRU)*. IEEE, 2017, pp. 193–199.

[3] T. Bagby, K. Rao, and K. C. Sim, "Efficient implementation of recurrent neural network transducer in tensorflow," in *2018 IEEE Spoken Language Technology Workshop (SLT)*. IEEE, 2018, pp. 506–512.

[4] S. T. N, R. Pang, R. David, Y. He, P. Rohit, W. Li, V. Mirkó, Q. Liang, S. Trevor, Y. Wu *et al.*, "Two-pass end-to-end speech recognition," *arXiv preprint arXiv:1908.10992*, 2019.

[5] J. Li, R. Zhao, H. Hu, and Y. Gong, "Improving rnn transducer modeling for end-to-end speech recognition," in *2019 IEEE Automatic Speech Recognition and Understanding Workshop (ASRU)*. IEEE, 2019, pp. 114–121.

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

[7] 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 *ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2019, pp. 6381–6385.

[8] Z. Tian, J. Yi, J. Tao, Y. Bai, and Z. Wen, "Self-Attention Transducers for End-to-End Speech Recognition," in *Proc. Interspeech 2019*, 2019, pp. 4395–4399.

[9] Q. Zhang, H. Lu, H. Sak, A. Tripathi, E. McDermott, S. Koo, and S. Kumar, "Transformer transducer: A streamable speech recognition model with transformer encoders and rnn-t loss," in *ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2020, pp. 7829–7833.

[10] C.-F. Yeh, J. Mahadeokar, K. Kalgaonkar, Y. Wang, D. Le, M. Jain, K. Schubert, C. Fuegen, and M. L. Seltzer, "Transformer-transducer: End-to-end speech recognition with self-attention," *arXiv preprint arXiv:1910.12977*, 2019.

[11] D. Bahdanau, K. Cho, and Y. Bengio, "Neural machine translation by jointly learning to align and translate," *arXiv preprint arXiv:1409.0473*, 2014.

[12] C. J. K, B. Dzmitry, S. Dmitriy, C. Kyunghyun, and B. Yoshua, "Attention-based models for speech recognition," in *Advances in neural information processing systems*, 2015, pp. 577–585.

[13] W. Chan, N. Jaitly, Q. Le, and O. Vinyals, "Listen, attend and spell: A neural network for large vocabulary conversational speech recognition," in *2016 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2016, pp. 4960–4964.

[14] K. Suyoun, H. Takaaki, and W. Shinji, "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.

[15] D. Linhao, X. Shuang, and X. Bo, "Speech-transformer: a no-recurrence sequence-to-sequence model for speech recognition," in *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2018, pp. 5884–5888.

[16] G. Alex, F. Santiago, G. Faustino, and S. Jürgen, "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.

[17] A. Y. Hannun, C. Case, J. Casper, B. Catanzaro, G. Diamos, E. Elsen, R. Prenger, S. Satheesh, S. Sengupta, A. Coates, and A. Y. Ng, "Deep speech: Scaling up end-to-end speech recognition," *arXiv preprint arXiv:1412.5567*, 2014.

[18] L. Jason, L. Vitaly, G. Boris, L. Ryan, K. Oleksii, C. Jonathan M., N. Huyen, and G. Ravi Teja, "Jasper: An end-to-end convolutional neural acoustic model," in *Interspeech 2019*, 2019, pp. 71–75.

[19] J. Navdeep, S. David, L. Q. V, V. Oriol, S. Ilya, and B. Samy, "A neural transducer," *arXiv preprint arXiv:1511.04868*, 2015.

[20] Z. Tian, J. Yi, Y. Bai, J. Tao, S. Zhang, and Z. Wen, "Synchronous transformers for end-to-end speech recognition," in *ICASSP 2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2020, pp. 5666–5670.

[21] S. Hasim, S. Matt, R. Kanishka, and B. Françoise, "Recurrent neural aligner: An encoder-decoder neural network model for sequence to sequence mapping." in *Interspeech*, vol. 8, 2017, pp. 1298–1302.

[22] L. Dong, S. Zhou, W. Chen, and B. Xu, "Extending recurrent neural aligner for streaming end-to-end speech recognition in mandarin," *arXiv preprint arXiv:1806.06342*, 2018.

[23] L. Dong, F. Wang, and B. Xu, "Self-attention aligner: A latency-control end-to-end model for asr using self-attention network and chunk-hopping," in *ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2019, pp. 5656–5660.

[24] J. Kim and Y. Lee, "Accelerating rnn transducer inference via one-step constrained beam search," *arXiv preprint arXiv:2002.03577*, 2020.

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

[26] Y. Zhang, S. Sun, and L. Ma, "Tiny transducer: A highly-efficient speech recognition model on edge devices," *arXiv preprint arXiv:2101.06856*, 2021.

[27] L. Rabiner, "A tutorial on hidden markov models and selected applications in speech recognition," *Proceedings of the IEEE*, vol. 77, no. 2, pp. 257–286, 1989.

[28] J. Yu, C.-C. Chiu, B. Li, S.-y. Chang, T. N. Sainath, Y. He, A. Narayanan, W. Han, A. Gulati, Y. Wu *et al.*, "Fastemit: Low-latency streaming asr with sequence-level emission regularization," *arXiv preprint arXiv:2010.11148*, 2020.

[29] B. Li, A. Gulati, J. Yu, T. N. Sainath, C.-C. Chiu, A. Narayanan, S.-Y. Chang, R. Pang, Y. He, J. Qin *et al.*, "A better and faster end-to-end model for streaming asr," *arXiv preprint arXiv:2011.10798*, 2020.

[30] A. Narayanan, T. N. Sainath, R. Pang, J. Yu, C.-C. Chiu, R. Prabhavalkar, E. Variani, and T. Strohman, "Cascaded encoders for unifying streaming and non-streaming asr," *arXiv preprint arXiv:2010.14606*, 2020.

[31] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, "Attention is all you need," in *Advances in neural information processing systems*, 2017, pp. 5998–6008.

[32] P. D. S, W. Chan, Y. Zhang, C.-C. Chiu, B. Zoph, C. E. D, and Q. V. Le, "Specaugment: A simple data augmentation method for automatic speech recognition," *arXiv preprint arXiv:1904.08779*, 2019.