INTERSPEECH 2020
October 25–29, 2020, Shanghai, China

Multilingual Speech Recognition with Self-Attention Structured Parameterization

Yun Zhu, Parisa Haghani, Anshuman Tripathi, Bhuvana Ramabhadran*, Brian Farris*, Hainan Xu*, Han Lu*, Hasim Sak*, Isabel Leal*, Neeraj Gaur*, Pedro J. Moreno*, Qian Zhang**

Google

yunzhu@google.com, parisah@google.com

Abstract

Multilingual automatic speech recognition systems can transcribe utterances from different languages. These systems are attractive from different perspectives: they can provide quality improvements, specially for lower resource languages, and simplify the training and deployment procedure. End-to-end speech recognition has further simplified multilingual modeling as one model, instead of several components of a classical system, have to be unified. In this paper, we investigate a streamable end-to-end multilingual system based on the Transformer Transducer [1]. We propose several techniques for adapting the self-attention architecture based on the language id. We analyze the trade-offs of each method with regards to quality gains and number of additional parameters introduced. We conduct experiments in a real-world task consisting of five languages. Our experimental results demonstrate $\sim$8% to $\sim$20% relative gain over the baseline multilingual model.

Index Terms: speech recognition, multilingual, RNN-T, Transformer Transducer, language id

1. Introduction

With more than 7000 languages in the world, there is considerable interest and importance to support multiple languages in a single automatic speech recognition (ASR) system. However, not all languages are widely available, with several falling under the category of low-resource languages. Many of these languages also share acoustic and/or linguistic structures. To take advantage of the commonality and harvest rich-resource languages, multilingual models have been studied over the last two decades and have outperformed monolingual models for ASR in low-resource languages [2, 3, 4, 5, 6]. Conventional ASR concentrates on the area of multilingual acoustic modeling with a variety of approaches, including new network structure [6, 7], multi-task learning [8], layer sharing [4], residual learning [9], knowledge distillation [10], and data augmentation [11]. However, language-specific pronunciation and language models are still needed.

In recent years, end-to-end (E2E) ASR has emerged as the state-of-the-art. E2E multilingual models have been proposed using different architectures including CTC [12, 13], LAS [14, 15], and RNN-T [16, 17]. These models replace the acoustic, pronunciation, and language models of n different languages with a single model. One key finding in the literature is that multilingual ASR models benefit significantly from the identity of the language being spoken, i.e., when language ID (LID) is provided as an input to the model. There have been a variety of approaches proposed for leveraging LID. Language gating was proposed in [13], while [17] used adapter layers to reflect the amount of training material. LID features have also been combined with either the input acoustic feature itself [14, 15, 16, 17] or with the target label embedding [15].

More recently, transformer-based architectures [18] have been applied to ASR systems [19, 20, 1, 21] successfully. In [22], the author showed that transformers also outperform RNNs in multilingual end-to-end ASR. In [23], the multilingual transformer was trained with language information in the output targets, i.e., the target labels were annotated the beginning and end of sub-words with LID information. This transformer achieved a significant improvement over their state-of-the-art residual learning based LSTM model.

In this work, we introduce a streaming E2E multilingual model based on the work presented in [1]. We introduce several techniques to parameterize the self-attention module of transformers using language ID, while adhering to the latency constraints required for interactive applications. We address the challenges of training such a model with large-scale real world data by a set of languages consisting of 5 languages, namely, Danish, Norwegian, Swedish, Finnish and Dutch.

We propose two novel approaches to include LID information that take advantage of the unique structure of the self attention layer: multi-head and attention span.

- A language specific head that assigns each language to a specific set of heads during training and inference.
- A language dependent, left-context, self-attention span that learns the optimal limited left context per language.

We also explore two existing approaches previously used in other encoder-decoder based E2E architectures and incorporate them into the transformer transducer (T-T). These include concatenating LID with the input acoustic features and language adaptor modules [17]. Finally, we investigate the combination of all these approaches and demonstrate that the best result is obtained by LID concatenation to self-attention inputs in all layers, and using language-specific attention heads. We demonstrate relative gains of up to 20% over the baseline multilingual model with no language-specific parameters.

2. Model Structure

The multilingual and monolingual transformer architecture and training objective are based on [1]. The model is composed of three main components: an audio encoder, a label encoder, and a joint network. The audio encoder maps the input acoustic features to higher level representations. The label encoder similarly encodes the model's previous non-blank label prediction. The two high level representations are then combined in

*: Authors sorted alphabetically.

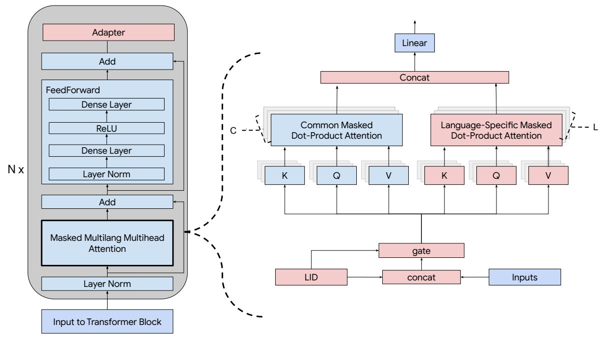
Adapter
Add
FeedForward
Dense Layer
ReLU
Dense Layer
Layer Norm
Add
Masked Multilang Multihead Attention
Layer Norm
Input to Transformer Block
N x
Linear
Concat
C
Common Masked Dot-Product Attention
Language-Specific Masked Dot-Product Attention
L
K
Q
V
K
Q
V
gate
LID
concat
Inputs

Figure 1: *Using language id in the multi-headed attention layer. The blue components are from basic Transformer while the red components are language specific.*

the joint network, followed by a softmax layer. This architecture is very similar to the RNN-T model in [24] with the audio encoder replaced by transformer blocks. The model is trained end-to-end using the RNN-T training objective [24]. When used with a unidirectional audio encoder, the RNN-T model provides a streaming implementation as the output does not depend on future acoustic frames [25]. Here we provide a brief recap of the RNN-T training objective, for details please see [24, 1]. Given the input sequence of real-valued vectors of length T, $\mathbf{x} = (x_1, x_2, ..., x_T)$, and the target sequence of labels $\mathbf{y} = (y_1, y_2, ..., y_U)$ of length U, RNN-T training objective defines a conditional distribution $P(\mathbf{y}^*|\mathbf{x})$ over all the possible alignments of x and y, where y_i^* can optionally be a blank label or real label $\mathbf{y}$. The probability of the target label sequence $\mathbf{y}$ is computed by marginalizing this conditional distribution over all possible alignments $\mathbf{y}^*$ as described in Equation (1),

$$P(\mathbf{y}|\mathbf{x}) = \sum_{\mathbf{y}^* \in \mathcal{Z}(\mathbf{y},T)} P(\mathbf{y}^*|\mathbf{x}), \qquad (1)$$

where $\mathcal{Z}(\mathbf{y}, T)$ is the set of valid alignments of length T for the label sequence.

Each transformer block in the audio encoder comprises of a multi-headed self-attention layer followed by a feed-forward layer. The left side of Figure 1 illustrates the components of a transformer block. To enable streaming, we use relative positional embedding [26] inside the self-attention layer and include only limited left context when computing the current attention weights. The label encoder is a stack of two LSTM layers. This is motivated by the findings in [27], where it was established that the label encoder merely helps the model align input audio and has little impact on the overall performance. The joint network is a feed-forward network [1].

3. Leveraging Language ID in Transformer

In this section, we present the different approaches to parameterize the self-attention module based on LID.

3.1. Language Embedding Concatenation

We use an one-hot vector to represent language id and append it to the input of multi-headed attention block. Recall that in the self attention network, Query, Key, and Value are calculated based on the input features and positional embedding. We define the Query, Key and Value vectors $\mathrm{Q_i}$, $\mathrm{K_i}$, and $\mathrm{V_i}$ using Equation 2, where $\mathrm{X_i}$ is the addition of acoustic features and positional embedding, $\mathrm{Q_w}$, $\mathrm{K_w}$, and $\mathrm{V_w}$ are the projection matrices, and $\mathrm{d_l}$, is the LID embedding vector. As shown, on the right hand side of the equation, concatenating LID to the input features is equivalent to having language-specific bias terms: $\mathrm{Q_l}$, $\mathrm{K_l}$, and $\mathrm{V_l}$.

$$\begin{aligned} Q_i &= Q_w * [X_i|d_l] + b_q = Q_x * X_i + Q_l * d_l + b_q, \\ K_i &= K_w * [K_i|d_l] + b_k = K_x * X_i + K_l * d_l + b_k, \\ V_i &= V_w * [X_i|d_l] + b_v = V_x * X_i + V_l * d_l + b_v, \end{aligned} \qquad (2)$$

The final attention is given in Equation 3 where $\mathrm{d_k}$ is the input dimension. Note that the LID is not concatenated to the residue connect (i.e.$(x + \mathrm{AttentionLayer}(\mathrm{concat}(x, d)))$).

$$Attention(Q, K, V) = softmax\left(\frac{QK^T}{\sqrt{d_k}}\right)V \qquad (3)$$

3.2. Language Specific Attention Head

In a typical self attention network, multiple heads are used to learn different perspectives of the inputs [18]. The attention network is defined in Equation 4,

$$head_j = Attention_j(Q, K, V), j \in [1, H] \qquad (4)$$

where H is the number of heads in the attention network. Each head has a different set of projection metrics.

In this work, we split the attention heads into language-specific and shared components. Weights associated with the language-specific heads are trained using data from the associated language only while the remaining heads are shared across all languages. Let L and K denote the number of languages and number of language specific heads. For language $\mathrm{l} \in [0, \mathrm{L} - 1]$ the language specific heads are given by Equation 5.

$$\mathrm{LangSpecificHeads_l} = concat_{k\in[1,K]}(head_{l*K+k}) \qquad (5)$$

The remaining $\mathrm{H} - \mathrm{L} * \mathrm{K}$ heads are common among all languages and are given by Equation 6.

$$\mathrm{CommonHeads} = concat_{c\in[L*K,H]}(head_c) \qquad (6)$$

Finally, the language-specific and common heads are concatenated as shown in Equation 7 and projected to a higher dimension as the final output of the self-attention network.

$$\mathrm{MultiHead} = concat(\mathrm{LangSpecificHeads_l}, \mathrm{CommonHeads}) \qquad (7)$$

3.3. Language Dependent Attention Span

A streaming model is achieved by using left-context only in the self-attention network. Furthermore, to lower the computing resources during inference we do not use full left context. While it is possible to manually specify the length of the left context, in practice we believe there is a language dependency that can be learned. Different languages might be sensitive to different left contexts and thus we introduce a learn-able variable to represent it. Then for each layer and each language, we add a masking function that could be imposed on the attention weight. This function maps a distance between two time-step to a real

Table 1: *Train and Test data per language (in number of utterances)*

Language	Train	Test
da-dk	3.7M	4.2K
fi-fi	3.8M	15K
nb-no	4.9M	3K
nl-nl	9.3M	12K
sv-se	8.2M	15K

value between 0 and 1 similar to [28]. Denote by M_l the mask generated by the function for language l, we can formulate the final attention value for language l as Equation 8.

$$Attention_l(Q, K, V) = softmax\left(\frac{QK^T \cdot M_l}{\sqrt{d_k}}\right)V \quad (8)$$

3.4. Language Adapter Layers

The concept of adapter layers was introduced in [29] for natural language processing. In [17], language adapter layers were used in a LSTM-based speech encoder. Both these approaches used adapter modules during language-specific fine-tuning. In this paper, we append language adapter layer right after the Feed-forward network in the transformer block.

4. Experiments

4.1. Experiment Setup

4.1.1. Languages and Data

We conduct our experiments on a set of five languages that consist of Danish(da-dk), Norwegian(nb-no), Swedish(sv-se), Finnish(fi-fi) and Dutch(nl-nl). This set represents a mix of languages that are generally mutually intelligible (The Nordics family) in addition to having a large alphabet intersection. For all of these languages, the training data is anonymized and transcribed by humans. The test sets, sampled from the same distribution, are similarly anonymized and transcribed by humans. The amounts of train and test data used per language are summarized in Tables 1. We use two forms of data augmentation to mitigate overfitting and improve generalization: First, we augment the data by corrupting the original utterances using a room simulator, and varying degrees of noise and reverberation [30]. We augment all languages' data by this method by a factor of 25. Second, we apply spectral-augmentation [31] with frequency masking ($F = 50$, $mF = 2$) and time masking ($T = 30$, $mT = 10$).

4.1.2. Model architecture and Training Configuration

We use the same acoustic features across all experiments: 80-dimensional log-Mel filterbanks, computed with a 25 msec window, shifted every 10 msec. Features from 3 contiguous frames are stacked, resulting in a 240-dimensional vector. These stacked features are downsampled by a factor of 3 generating inputs at 30ms frame rate that the audio encoder operates on. The audio encoder in our model consists of 14 transformer blocks. See Table 2 for more details on the sizes of the layers inside the attention block. Every two frames are stacked into one on the 3rd layer and then un-stacked at 11th. The label encoder consists of 2 LSTM layers with 2048 hidden units and a 640-dimensional projection per layer. The outputs of the audio and label encoders are fed to a joint-network consisting of 640 hidden units whose output is fed to the softmax layer. We train this model to output word-piece [32] units in all our experiments. The multilingual output inventory is constructed by combining training data from all languages and building a wordpiece inventory of size 4k.

All the models for experiments presented in this paper are trained on 8x8 TPU with a per-core batch size of 32 (effective batch size of 4096). The learning rate schedule is ramped up linearly from 0 to $4.0e{-4}$ during first 4K steps, it is then held constant till 50K steps and then decays exponentially to $4.0e{-6}$ till 200K steps. During training we also added a gaussian noise($\mu = 0, \sigma = 0.01$) to model weights [33] starting at 10K steps. We did not find dropout to be useful so the models are trained without dropout.

Table 2: *Transformer encoder parameter setup.*

Input feature	240
Dense layer 1	2048
Dense layer 2	1024
Number of attention heads	8
Head dimension	64
Attention layer	1024

Table 3: *Comparison of variations of Language Embedding Concatenation and baseline. The second column shows the relative increase in number of parameters from the baseline model.*

	Params Inc.(%)	WER da-dk	fi-fi	nb-no	nl-nl	sv-se
Baseline	-	14.9	19.7	17.7	14.7	17.9
First	0.02	12.9	19.1	14.7	14.2	16.4
All	0.3	**12.3**	**18.6**	**14.3**	**13.8**	**15.9**
All-KQ	0.2	13.2	19.0	14.7	14.3	16.4

4.2. Results

As the baseline model, we train a Transformer Transducer model with all languages. Each training mini batch consists of utterances from all languages, sampled according to their natural training data distribution.

4.2.1. Language Embedding Concatenation

We use a one-hot vector of size 16 as our LID vector. We tried three different variations of concatenating this vector to the self-attention network: 1/ only injecting this information into the first layer of the transformer. In RNN-T with LSTM encoders this approach performed the best [16]. 2/ Injecting LID to all layers (14 in total) of the transformer stack. And 3/ injecting to all layers, similar to 2/ but only projecting the LID information to Query, Key, and not Value. The result is summarized in Table 3.

Concatenating LID to transformer inputs improves WER for all languages by an average of 9%. Among the different variations, injecting LID into every layer has the highest gain and achieves on average 3% gain over concatenating to the first layer only. This finding is in contrast to the findings of a similar approach on an LSTM acoustic encoder [16]. This may be due to the transformer encoder being much deeper than the LSTM encoder, causing the LID information to be gradually

Table 4: *Comparison of the Language Specific Attention head approach when varying total number of attention heads (H) and language specific ones (K).*

H(K)	Params Inc.(%)	WER da-dk	fi-fi	nb-no	nl-nl	sv-se
8(0)	-	14.9	19.7	17.7	14.7	17.9
8(1)	-3.01	13.1	19.0	14.7	14.7	17.2
16(0)	30.07	14.4	18.4	16.8	14.1	17.1
16(1)	27.07	**12.5**	**18.2**	**14.4**	**14.0**	**16.2**
16(2)	24.06	12.6	18.3	14.5	14.2	16.2
16(3)	21.05	14.6	20.9	16.5	15.8	18.4

"forgotten" by the network in the higher layers. Meanwhile, all-layer-KQ approach stays in between the baseline and all-layer, showing that the LID impacts more than the weight distribution of the attention network.

From the perspective of the extra parameters, concatenating LID to inputs introduces per-language bias terms in the self-attention layer. The overall increase in the number of parameters remains very small (0.3% for the best approach).

4.2.2. *Language Specific Head*

To investigate the effectiveness of our proposed method, we explored several language-specific head configurations. We increase the total number of heads to 16 to allow experimenting with more than 1 language-specific head. The results are summarized in Table 4. The baseline configuration is represented as 8(0), denoting 8 total attention heads and 0 language-specific ones.

We observe that including a language-specific head results in significant WER improvements across all languages (compare 8(0) with 8(1), and 16(0) with 16(1)). Furthermore, the use of language-specific heads results in fewer total parameters as it affects the size of the Feed-forward layer that projects the concatenated heads to the output: for each utterance, only its language-specific and common heads are concatenated, thereby resulting in fewer parameters in the projection matrix.

The other noteworthy result is that increasing the number of language-specific heads results in regressions (compare 16(2) and 16(3) to 16(1)). This can be explained by noting that in this case, a larger portion of the attention weights are trained with a smaller effective batch size as language-specific weights only receive gradients from utterances from their language in the batch. Also it is worth noting that increasing the total number of attention heads results in modest improvements (comparing 8(0) and 16(0)) but at the cost of 30% increase in the number parameters.

4.2.3. *Language Dependent Left Context Span*

The adaptive span method learns the left context in the range [0, 128], per language, while the baseline is fixed to 128. As seen in table 5, this approach results in an average of 7% WER improvement across all languages while also saving on compute resources during inference. Memory-wise, this approach will only introduce negligible constant variables.

4.2.4. *Language Adapter Layer*

The use of adapter layers helps improve overall performance. The gains are consistent for all languages as seen from the results in Table 6, especially for smaller-resource languages.

Table 5: *WER results for Language Dependent Left Context approach. Increase in number of parameters is negligible.*

	WER da-dk	fi-fi	nb-no	nl-nl	sv-se
Baseline	14.9	19.7	17.7	14.7	17.9
Lang-Specific Span	**13.6**	**19.5**	**15.0**	**13.9**	**16.9**

Table 6: *WER results when using Adapter layers.*

	Params Inc.(%)	WER da-dk	fi-fi	nb-no	nl-nl	sv-se
Baseline	-	14.9	19.7	17.7	14.7	17.9
Adapter	2.15	**13.7**	**18.7**	**14.7**	**14.4**	**16.9**

4.2.5. *Combination of the Approaches*

We investigate whether the combination of the approaches described thus far would result in further improvements in performance. We find that concatenating LID and learning language-specific attention heads can result in further improvements. However, the addition of adapter modules or adaptive language context causes a regression in performance, suggesting further tuning of parameters is needed. Table 7 summarizes the results for the best performing model and compares that to the single-lang and baseline multi-lang models.

Table 7: *Comparison of single-lang and multi-lang models with the best approach that combines LID concatenation and language-specific heads.*

Model	WER da-dk	fi-fi	nb-no	nl-nl	sv-se
Single-lang	12.9	19.3	15.6	**11.6**	**15.1**
Multi-lang	14.9	19.7	17.7	14.7	17.9
Concat+Head	**12.1**	**17.7**	**14.0**	13.5	15.8

The baseline multilingual model suffers from regressions in all languages compared to the single-language counterpart. Our best proposed approach, the combination of LID concatenation and language-specific attention heads, can mitigate this regression. In three of the five languages, it results in improvements of up to 10% compared to the single-language model. In the two higher resource languages, the regressions are decreased from an average of 20% to 10%.

5. Conclusions

We study the use of a Transformer based E2E model as a streaming multilingal ASR system. We show that careful parameterization of the self-attention module based on language ID can significantly improve the performance of the multilingual model. Lower-resource languages in particular see improvements over their single language counter parts, but more work is needed to match or improve the performance of such a model for higher resource languages.

6. References

[1] 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," *arXiv preprint arXiv:2002.02562*, 2020.

[2] A. Waibel, H. Soltau, T. Schultz, T. Schaaf, and F. Metze, "Multilingual speech recognition," in *Verbmobil: Foundations of Speech-to-Speech Translation*. Springer, 2000, pp. 33–45.

[3] J.-T. Huang, J. Li, D. Yu, L. Deng, and Y. Gong, "Cross-language knowledge transfer using multilingual deep neural network with shared hidden layers," in *2013 IEEE International Conference on Acoustics, Speech and Signal Processing*. IEEE, 2013, pp. 7304–7308.

[4] A. Ghoshal, P. Swietojanski, and S. Renals, "Multilingual training of deep neural networks," in *2013 IEEE International Conference on Acoustics, Speech and Signal Processing*. IEEE, 2013, pp. 7319–7323.

[5] S. Thomas, M. L. Seltzer, K. Church, and H. Hermansky, "Deep neural network features and semi-supervised training for low resource speech recognition," in *2013 IEEE international conference on acoustics, speech and signal processing*. IEEE, 2013, pp. 6704–6708.

[6] F. Grézl, M. Karafiát, and K. Veselỳ, "Adaptation of multilingual stacked bottle-neck neural network structure for new language," in *2014 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2014, pp. 7654–7658.

[7] T. Sercu, G. Saon, J. Cui, X. Cui, B. Ramabhadran, B. Kingsbury, and A. Sethy, "Network architectures for multilingual speech representation learning," in *2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2017, pp. 5295–5299.

[8] A. Mohan and R. Rose, "Multi-lingual speech recognition with low-rank multi-task deep neural networks," in *2015 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2015, pp. 4994–4998.

[9] S. Zhou, Y. Zhao, S. Xu, B. Xu *et al.*, "Multilingual recurrent neural networks with residual learning for low-resource speech recognition," 2017.

[10] J. Cui, B. Kingsbury, B. Ramabhadran, G. Saon, T. Sercu, K. Audhkhasi, A. Sethy, M. Nussbaum-Thom, and A. Rosenberg, "Knowledge distillation across ensembles of multilingual models for low-resource languages," in *2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2017, pp. 4825–4829.

[11] C. Liu, Q. Zhang, X. Zhang, K. Singh, Y. Saraf, and G. Zweig, "Multilingual asr with massive data augmentation," *arXiv preprint arXiv:1909.06522*, 2019.

[12] S. Watanabe, T. Hori, and J. R. Hershey, "Language independent end-to-end architecture for joint language identification and speech recognition," in *2017 IEEE Automatic Speech Recognition and Understanding Workshop (ASRU)*. IEEE, 2017, pp. 265–271.

[13] S. Kim and M. L. Seltzer, "Towards language-universal end-to-end speech recognition," in *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2018, pp. 4914–4918.

[14] S. Toshniwal, T. N. Sainath, R. J. Weiss, B. Li, P. Moreno, E. Weinstein, and K. Rao, "Multilingual speech recognition with a single end-to-end model," in *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2018, pp. 4904–4908.

[15] B. Li, T. N. Sainath, K. C. Sim, M. Bacchiani, E. Weinstein, P. Nguyen, Z. Chen, Y. Wu, and K. Rao, "Multi-dialect speech recognition with a single sequence-to-sequence model," in *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2018, pp. 4749–4753.

[16] A. Waters, N. Gaur, P. Haghani, P. Moreno, and Z. Qu, "Leveraging language id in multilingual end-to-end speech recognition," in *2019 IEEE Automatic Speech Recognition and Understanding Workshop (ASRU)*. IEEE, 2019, pp. 928–935.

[17] A. Kannan, A. Datta, T. N. Sainath, E. Weinstein, B. Ramabhadran, Y. Wu, A. Bapna, Z. Chen, and S. Lee, "Large-scale multilingual speech recognition with a streaming end-to-end model," *arXiv preprint arXiv:1909.05330*, 2019.

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

[19] L. Dong, S. Xu, and B. Xu, "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.

[20] 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.

[21] N. Moritz, T. Hori, and J. L. Roux, "Streaming automatic speech recognition with the transformer model," *arXiv preprint arXiv:2001.02674*, 2020.

[22] S. Karita, N. Chen, T. Hayashi, T. Hori, H. Inaguma, Z. Jiang, M. Someki, N. E. Y. Soplin, R. Yamamoto, X. Wang *et al.*, "A comparative study on transformer vs rnn in speech applications," *arXiv preprint arXiv:1909.06317*, 2019.

[23] S. Zhou, S. Xu, and B. Xu, "Multilingual end-to-end speech recognition with a single transformer on low-resource languages," *arXiv preprint arXiv:1806.05059*, 2018.

[24] A. Graves, "Sequence transduction with recurrent neural networks," in *Proceedings of the 29th International Conference on Machine Learning*, 2012.

[25] Y. He, T. Sainath, R. Prabhavalkar, I. McGraw, R. Alvarez, D. Zhao, D. Rybach, A. Kannan, Y. Wu, R. Pang, Q. Liang, D. Bhatia, Y. Shangguan, B. Li, G. Pundak, K. C. Sim, T. Bagby, S. yiin Chang, K. Rao, and A. Gruenstein, "Streaming end-to-end speech recognition for mobile devices," 2019. [Online]. Available: https://arxiv.org/abs/1811.06621

[26] Z. Dai, Z. Yang, Y. Yang, W. W. Cohen, J. Carbonell, Q. V. Le, and R. Salakhutdinov, "Transformer-xl: Attentive language models beyond a fixed-length context," in *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, 2019, p. 29782988.

[27] M. Ghodsi, X. Liu, J. Apfel, R. Cabrera, and E. Weinstein, "Rnn-transducer with stateless prediction network," 2020.

[28] S. Sukhbaatar, E. Grave, P. Bojanowski, and A. Joulin, "Adaptive attention span in transformers," *arXiv preprint arXiv:1905.07799*, 2019.

[29] N. Houlsby, A. Giurgiu, S. Jastrzebski, B. Morrone, Q. De Laroussilhe, A. Gesmundo, M. Attariyan, and S. Gelly, "Parameter-efficient transfer learning for nlp," *arXiv preprint arXiv:1902.00751*, 2019.

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

[31] 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," *arXiv preprint arXiv:1904.08779*, 2019.

[32] M. Schuster and K. Nakajima, "Japanese and korean voice search," in *2012 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2012, pp. 5149–5152.

[33] A. Graves, "Practical variational inference for neural networks," in *Advances in neural information processing systems*, 2011, pp. 2348–2356.