DUAL-ATTENTION NEURAL TRANSDUCERS FOR EFFICIENT WAKE WORD SPOTTING IN SPEECH RECOGNITION

Saumya Y. Sahai, Jing Liu*, Thejaswi Muniyappa, Kanthashree M. Sathyendra, Anastasios Alexandridis, Grant P. Strimel, Ross McGowan, Ariya Rastrow, Feng-Ju Chang, Athanasios Mouchtaris, Siegfried Kunzmann*

Amazon Alexa AI, USA

ABSTRACT

We present dual-attention neural biasing, an architecture designed to boost Wake Words (WW) recognition and improve inference time latency on speech recognition tasks. This architecture enables a dynamic switch for its runtime compute paths by exploiting WW spotting to select which branch of its attention networks to execute for an input audio frame. With this approach, we effectively improve WW spotting accuracy while saving runtime compute cost as defined by floating point operations (FLOPs). Using an in-house de-identified dataset, we demonstrate that the proposed dual-attention network can reduce the compute cost by 90% for WW audio frames, with only 1% increase in the number of parameters. This architecture improves WW F1 score by 16% relative and improves generic rare word error rate by 3% relative compared to the baselines.

***Index Terms*—** Speech recognition, inference optimization, wake word spotting, attention, neural biasing, personalization

1. INTRODUCTION

End-to-end (E2E) ASR systems such as connectionist temporal classification (CTC) [1], listen-attend-spell (LAS) [2], recurrent neural network transducer (RNN-T) [3], transformer transducer [4–8], and their variants ConvRNN-T [9], conformer [10, 11] have become increasingly popular due to their superior performance over hybrid HMM-DNN systems, making them promising architectures for deployment in commercial virtual voice assistants. While hybrid models optimize the acoustic model (AM), pronunciation model (PM) and language model (LM) independently, E2E systems jointly optimize them to output word sequences directly from an input sequence. These fully neural E2E approaches are strong candidates for low resource settings due to their simplicity and unified compression capabilities. However, one of the major limitations of E2E ASR systems is that they have difficulty in accurately recognizing words that are uncommon in the paired audio-text training data, such as custom WW which are specified by the customer to address a virtual assistant (e.g. Ziggy, Hey Shaq), contact names, proper nouns, and other rare named entities [12, 13]. To address this issue, previous works [8, 14] have proposed attention-based neural biasing which apply a biasing adapter mechanism by scoring the similarity of encoded audio representations with personalized catalog embeddings. Attention-based neural biasing is a promising approach to boost personalized entity names; however, due to its compute complexity by application on the audio encodings frame-by-frame, the incurred runtime latency challenges scalable deployment of these attention-based biasing networks for on-device systems with hardware constraints (e.g. limited memory bandwidth and CPU constraints).

To address compute limitations for on-device ASR, model compression is a commonly used methodology. In general, model compression techniques can be divided into two categories: architecture modification and weight interpretation. The former reduces complex architectures to simplified alternatives while the latter interprets weights with low-bit representations. Our work belongs to the architecture modification category. Also in this category are CIFG [15] which simplifies the LSTM structure [16] by merging the input and forget gates which results in 25% fewer parameters; simple recurrent unit [17] introduces more efficient recurrent cells for Edge ASR; low-rank factorization [18], bifocal [19], dynamic encoders [20], amortized networks [21, 22], linformer [23], performer [24] and time-reduction layers [2, 25, 26] which are suggested to reduce runtime latency. In the second category, quantization [27–29], sparsity [30–32] are dominant paradigms used to interpret weights with lower-bit integer or sparse representations.

Our work is inspired by the bifocal neural transducer [19], that contains two audio encoder networks which are dynamically pivoted at run time. One major difference in our work is the compute cost amortized Multi-Head Attention (MHA) [4] biasing networks designed to simultaneously boost custom WW and personalized entities. In contrast to vanilla neural biasing [8, 14] which does not differentiate sentence segments, the proposed dual-attention network biases towards only WW embeddings at the sentence-beginning, and proper name embeddings (e.g. contact names, device names) at post-WW segments.

2. RELATED WORK

2.1. Bifocal RNN-T

Neural sequence transducers are streaming E2E ASR systems [3] that typically consist of an audio encoder, a text predictor and a joint network. The encoder, behaving like an AM, produces high level acoustic representations $\mathbf{h}_t^{enc}$ for each input audio frame $\mathbf{x}_T = (x_0, \ldots, x_T)$. The text predictor, acting like an LM, encodes previously predicted word-pieces $\mathbf{y}_{u-1} = (y_0, \ldots, y_{u-1})$ and outputs $\mathbf{h}_u^{pred}$, with

$$\mathbf{h}_t^{enc} = \text{AudioEncoder}(\mathbf{x}_t); \quad \mathbf{h}_u^{pred} = \text{TextPredictor}(\mathbf{y}_{u-1}).$$

The joint network fuses $\mathbf{h}_t^{enc}$ and $\mathbf{h}_u^{pred}$ and passes them through dense and then softmax layers to obtain output probability distributions over the word-pieces.

Bifocal RNN-T [19], is a special type of neural transducer, consisting of two audio encoders: a small/fast encoder trained for the buffered lead-in audio segments that contains pre-WW and WW audio frames; and a large/slow encoder for processing the remainder of the audio leveraging WW spotting to pivot between the two (Fig.

[*] Equal Contribution.

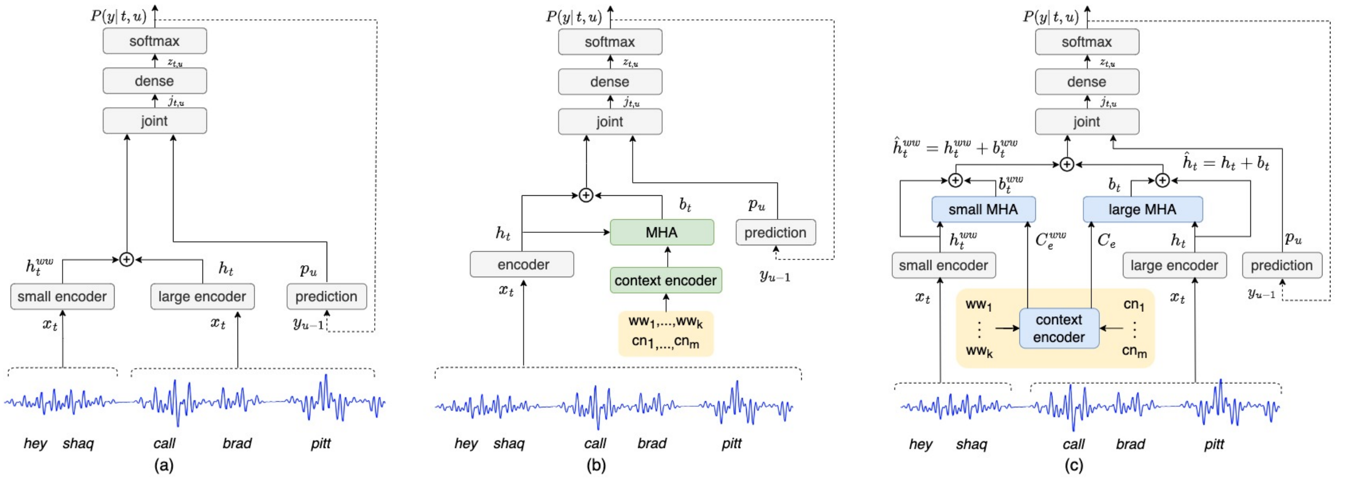

Fig. 1: Model architectures: (a) baseline bifocal neural transducer *pretrained-base* [19]; (b) baseline single-attention neural biasing *single-attn-base-128* [8, 14]; (c) proposed dual-attention neural biasing *dual-attn-λ*, where λ is the projection size of the small MHA; initialize the model with pretrained weights (grey blocks); add dual-attention modules (blue blocks); only train the blue blocks by freezing the grey ones.

1(a)). Bifocal architecture improves latency by diverting WW audio frames to its fast encoder branch. However, it has limited capacity to adapt itself to recognize new or rare words, particularly user-specified WW directed to its low-capacity small audio encoder. This drawback degrades user experience by falsely rejecting voice queries initiated from custom WW or diverting to another virtual assistant personality by mistake. The proposed dual-attention architecture is designed to mitigate this limitation using an attention network to "just focus on" boosting the desired WW.

2.2. Neural Biasing

To leverage a user's custom environment and preferences to improve recognition of personalized requests directed to voice assistants, both [8] and [14] suggest neural biasing (Fig. 1(b)) consisting of MHA layers [4] to measure the similarity of audio encoding with entity-name embedding. The attention weights are computed frame-by-frame to assess the relevance of user pre-defined entity-names with the current audio frame. Neural biasing effectively boosts personalized entity-names (e.g. *proper names* such as contacts and device names) because more relevant entity-names receive higher attention weights. However, due to quadratic complexity of dot-product attention [4], runtime latency has been a bottleneck to deploy neural biasing to embedded ASR systems.

3. DUAL-ATTENTION NEURAL BIASING

To address the on-device latency and compute limitations, and inspired by bifocal RNN-T and neural biasing (Sec. 2), we propose dual-attention neural biasing (Fig. 1(c)), which enables a dynamic pivot for its runtime compute paths, namely leveraging WW spotting to select the branch of the network to execute an input audio frame on. The motivation of dual attention is to introduce bifocal "lenses" engineered to focus on different segments of an utterance. The distinguishing feature of this design is training two alternative MHA networks (highlighted blue components in Fig. 1(c)). A small attention network $\mathcal{A}^s$, coupled with the small audio encoder, is trained for the lead-in segments and a large attention network $\mathcal{A}^l$ paired with the large audio encoder for the rest of the audio. $\mathcal{A}^s$ is designed to boost the ASR accuracy for user-specified custom WW (typically in the order of 10), while $\mathcal{A}^l$ is engineered to improve the recognition of personalized entity names (can scale to tens of thousands). $\mathcal{A}^s$ has a smaller number of hidden units than $\mathcal{A}^l$, enabling faster but coarser frame processing. In contrast, $\mathcal{A}^l$ has a larger capacity, but at the cost of more compute. The final component in this design is the context encoder, namely a BiLSTM encoder, which takes tokenized custom WW/proper names from a sentence-piece tokenizer [33]. The last state of this BiLSTM is used as the embedding C_e^{ww} or C_e. In Fig. 1 (c), we first pretrain the bifocal transducer (grey blocks), then fine tune only the context encoder and the two MHA models (blue blocks) by keeping the rest of the pretrained weights (grey blocks) frozen [14].

3.1. Dual-Attention Biasing Networks

The small MHA network $\mathcal{A}^s$ is trained to learn the correlation between the lead-in audio encoding and user enabled WW text embedding. It is a light-weight model thanks to the natural lower perplexity of the spoken words prior to the WW. In contrast, higher perplexity of the post WW segment requires an MHA model $\mathcal{A}^l$ with higher capacity. The objective of this dual-attention design is to match the accuracy of single-attention baseline (Sec. 2.2) and to reduce the FLOPs since this architecture emphasizes the reduction of the MHA inference cost as it is one of the primary runtime bottlenecks.

3.2. Dynamic Catalog Masking

In contrast to [8, 14] (Fig. 1(b)) which statically concatenates WW and *proper names* embeddings without differentiating sentence segments, one distinguishing feature in our design (Fig. 1(c)) is catalog masking which is dynamically determined by the frame index signaling the end of the WW. At inference time, we only apply WW embeddings C_e^{ww} in $\mathcal{A}^s$ by dynamically masking out *proper names* tokens. In this way, we effectively narrow down the biasing candidates from tens of thousands (i.e. *proper names*) to 6 (i.e. custom WW only) for better focusing, and to rule out less relevant catalogs like contacts/device names from appearing at the sentence-beginning. Similarly, for $\mathcal{A}^l$, we drop out WW catalogs and only apply *proper names* embeddings C_e. In this way, we enable the two

Model	Lead-in audio segment FLOPs	Biasing layer parameters	Lead-in audio catalog size	RNN-T			C-T		
				F1R	TRRR	TARR	F1R	TRRR	TARR
pretrained-base	–	–	–	–	–	–	–	–	–
single-attn-base-128	$3.3M$	$400K$	300	+14.72%	-14.80%	+28.73 %	+14.96%	-12.84%	+41.96%
single-attn-catalog-mask	$190K$ (-93.7%)	$400K$(+0.0%)	6	+17.03%	-15.17%	+31.39%	+16.93%	-11.11%	+42.77%
dual-attn-64	$95K$ (-96.9%)	$483K$(+20.7%)	6	+16.42%	-14.07%	+36.45%	+14.14%	-7.98%	+32.42%
dual-attn-32	$48K$ (-98.4%)	$441K$(+10.3%)	6	+16.00%	-12.43%	+26.07%	+15.34%	-12.15%	+42.51%
dual-attn-16	$23K$ (-99.2%)	$421K$(+5.1%)	6	+14.34%	-16.63%	+29.01%	+14.38%	-12.15%	+39.23%
dual-attn-8	$12K$ (-99.6%)	$410K$ (+2.5%)	6	+2.38%	-4.21%	+8.52%	+9.30%	-5.90%	+19.07%

Table 1: Compute cost measured in FLOPs (M=10^6, K=10^3); number of parameters in the biasing layers; relative changes in F1 score (denoted as F1R), True Reject Rate (TRRR) and True Accept Rate (TARR) for the proposed *dual-attn-*λ in Fig. 1(c), comparing with baseline *pretrained-base* as shown in Fig. 1(a); *single-attn-base-128* refers to Fig. 1(b); +/- sign means to improve/degrade.

MHAs to bias toward their own targets by masking out irrelevant catalogs. As shown in Fig. 1(c), the highlighted yellow box containing the context encoder runs offline to generate and cache neural embeddings C_e^{ww} for user-customized WW $ww_1, \ldots, ww_k$ and C_e for proper names $cn_1, \ldots, cn_m$. These cached embeddings C_e^{ww} and C_e are dynamically masked for runtime inference. We also introduce a special *no-bias* token into our catalog as in [14, 34] to help the dual-attention system to learn when not to bias.

4. EXPERIMENTS

4.1. Datasets

We use 114K hours of de-identified in-house voice assistant (*general*) dataset randomly sampled from live traffic across more than 20 domains (e.g. Music, Communications, SmartHome) to pretrain the baseline RNN-T and Conformer-Transducer (C-T) models[1]. For training the dual-attention networks, we use 290 hours of *proper names* (that contains mentions of named entities), *general* data which is mixed in the ratio of 1:2.5, and 3.6K hours of semi-supervised dataset containing 6 custom WW generated using a teacher model [35]. To evaluate the models, we use a 75-hour *general* testset and a 20-hour *proper names* testset which are both human-transcribed. For calculating WW true accept and true reject rates, we use 25 hours of human annotated data containing 6 WW. The training and test sets are de-identified and have no overlap.

4.2. Experimental Setup

We evaluate the dual-attention neural transducers with two pretrained ASR architectures, RNN-T and C-T.

RNN-T and C-T Pretraining. The input audio features are 64-dimensional LFBE features extracted every 10ms with a window size of 25ms resulting in 192 feature dimensions per frame. Ground truth tokens are passed through a 2.5K and 4K word-piece tokenizer [33, 36] for RNN-T and C-T, respectively. The RNN-T encoder has 5 LSTM layers and a time reduction layer with downsampling factor of 2 at layer 3. Each LSTM layer has 256 units each layer for the lead-in audio encoder and 1120 units for the large audio encoder (Fig. 1(a)). The prediction network has 2 LSTM layers with 1088 units each. The C-T encoder network consists of 2 convolutional layers with 128 kernels of size 3, and strides 2 and 1 for the first and second layer, respectively, followed by a dense layer to project input features to 512 dimensions. They are then fed into 14 conformer blocks, that contain layer normalizations and residual links between layers. Each conformer block has a 1024 unit feedforward layer, 1 transformer layer with 8 64-dimensional attention heads and 1 convolutional module with kernel size 15. The prediction network has 2 LSTM layers with 736 units each. Convolutions and attentions are computed on the current and previous audio frames to make it streamable. For both RNN-T and C-T, the encoder and prediction network outputs are projected through 512 units of a feedforward layer.

Baselines. The baseline *pretrained-base* shown in Fig. 1(a) is a bifocal neural transducer [19] (RNN-T or C-T) which has two audio encoders but no neural biasing layers [8, 14]. This model has fast inference but poor accuracy on *proper names*. The second baseline *single-attn-base-128*, displayed in Fig. 1(b), is a single-attention neural biasing transducer model as in [8, 14] with keys and values projected to 128 dimensions. This baseline has good accuracy on *proper names* but slower inference at runtime compared to the *pretrained-base*. The third model *single-attn-catalog-mask* is the same as *single-attn-base-128* (Fig. 1(b)) except that irrelevant catalogs (e.g. *proper names*) are removed from the small audio encoder via dynamic catalog masking (Sec. 3.2).

Configuration for dual-attention biasing networks. The context encoder is a BiLSTM with 64 units (for each forward and backward LSTM). The input and output have 64-dimensional projections. This context encoder is trained from scratch to generate embeddings for both WW and proper names. These embeddings are then fed into the dual-attention layers to bias the audio encoders outputs (Fig. 1(c)). More precisely, the large MHA network takes the large audio encoder outputs as query, and proper names embeddings as key and value and projects them to 128 dimensions. On the other hand, the small MHA is a light-weight model which takes query from the small audio encoder outputs, and WW embeddings as key and value and projects them to size λ, denoted as *dual-attn-*λ. We experiment with $\lambda = 64, 32, 16$, and 8. In the following experiments, both small and large MHA only use 1 attention head, since we did not observe accuracy gains with 2 or 4 heads.

5. RESULTS

Given a model A's WER (WER$_A$) and a baseline B's WER (WER$_B$), the relative Word Error Rate Reduction (WERR) of A over B is computed as WERR $= (\text{WER}_B - \text{WER}_A)/\text{WER}_B$. The WW accuracy is measured in terms of True Accept Rate (TAR), True Reject Rate (TRR) and F1 score, where TAR is the proportion of ground truth positives that are accepted correctly; TRR is the fraction of ground truth negatives that are rejected correctly and F1 score is the harmonic mean of TAR and TRR. We present F1, TAR and TRR relative improvement (denoted by F1R, TARR and TRRR respectively)

[1] As far as we know, there does not exist a large-scale public dataset that contains a variety of user-customized WW.

	RNN-T		C-T	
	general	*proper names*	*general*	*proper names*
pretrained-base	–	–	–	–
single-attn-base-128	+0.2%	+26.73%	+0.2%	+29.08%
single-attn-catalog-mask	+0.2%	+28.13%	-0.2%	+29.63%
dual-attn-64	+0.2%	+28.13%	0.0%	+29.23%
dual-attn-32	+0.8%	+27.24%	0.0%	+29.08%
dual-attn-16	+0.8%	+27.98%	0.0%	+30.42%
dual-attn-8	+0.2%	+28.64%	-0.2%	+29.23%

Table 2: WERR relative to *pretrained-base*; +/- sign implies an improvement/degradation in WER.

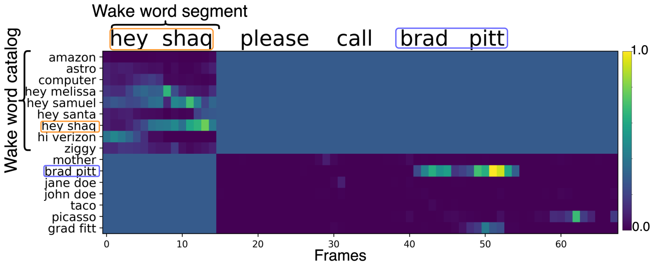

Fig. 2: Attention weight visualization over different catalog entities for *dual-attn-64* model (bright colors represent a greater weight). Each frame represents 60ms.

against a baseline model. Higher values of WERR, F1R, TARR and TRRR represent better performance. Negative values mean degradation. To measure compute cost, we report the total number of MHA-layer floating point operations [19, 21] per frame (FLOPs) required for the lead-in audio.

5.1. Wake word accuracy

As the model learns to bias towards the desired WW, TARs naturally improve. TRR may degrade since the model is biased to accept more queries. From Table 1, the proposed model *dual-attention-λ* improves WW TARR by up to 36% (RNN-T) and 42% (C-T) against their bifocal baselines *pretrained-base*. As we reduce the projection dimension λ from 64 to 16, we still see 29% improvement of TARR for RNN-T, and 39% for C-T. When λ=8, the TARR improvement is 8.5% and 19% for RNN-T and C-T respectively. On the other hand, we observed up to 14% and 12% regression in TRRs for RNN-T and C-T. In fact, the proposed dual-attention model *dual-attn-λ* slightly outperforms Fig. 1(b) baseline *single-attn-base-128* in TRRR for $\lambda = 64, 32, 8$ for RNN-T and all values of λ for C-T. Taking into account of both TAR and TRR, *dual-attention-λ* improves F1-score by up to 17% for both RNN-T and C-T.

5.2. ASR accuracy

Table 2 presents the WERR for the proposed *dual-attention-λ* (Fig. 1(c)) against the bifocal baseline *pretrained-base* (Fig. 1(a)). On *proper names* test set, dual-attention architectures reduce WER by up to 28.6% for RNN-T (vs. 26.7% for *single-attn-base-128*), and 30.4% for C-T (vs. 29.1% for *single-attn-base-128*) thanks to catalog masking (detailed in Sec. 3.2) and a specialized MHA trained to "focus on" more relevant catalogs. As we reduce λ from 64 to 8 for the WW MHA network, we do not observe accuracy degradation against *single-attn-base-128* on *proper names* or *general* test sets. This shows that *dual-attn-λ* reduces compute cost without hurting ASR accuracy for both RNN-T and C-T.

5.3. Compute Cost & Attention Visualization

Table 1 shows the compute cost measured in FLOPs. Using neural biasing [8, 14] *single-attn-base-128* as baseline (Fig. 1(b)), with a small increase of $83K$ parameters (from $400K$ to $483K$ or 1% of $\sim 80M$ parameters), FLOPs (with catalog size 300) decreases from $3.3M$ (*single-attn-base-128*) to $95K$ (*dual-attn-64*). As we reduce λ from 64 to 8, we further improve FLOPs from 95K to 12K. However, when $\lambda = 8$, we observe reduced gain in WW accuracy: a relative F1 score improvement of 1% and 6% respectively for RNN-T and C-T. It is worth noting that catalog masking plays an important role in reduce FLOPs (from $3.3M$ to $190K$) as we narrow down the biasing candidates from 300 to 6 for the lead-in audio segment. In figure 2, we visualize attention heat map for the small and large MHA layers with catalog masking. The small MHA layer shows high values for true WW *hey shaq* while less relevant *proper names* catalogs are masked out, whereas the large MHA biases towards the target proper name *brad pitt* while the WW catalogs are masked out.

6. CONCLUSION

We proposed a dual-attention neural transducer network which was inspired by bifocal RNN-T as well as attention-based neural biasing. This proposed architecture exploited WW spotting to dynamically select a biasing branch and efficiently boosted the ASR accuracy of proper names as well as custom WW, at the same time reducing runtime compute FLOPs and alleviated runtime latency of attention networks.

7. REFERENCES

[1] Alex Graves, Santiago Fernández, Faustino Gomez, and Jürgen 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.

[2] William Chan, Navdeep Jaitly, Quoc Le, and Oriol 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.

[3] A. Graves, "Sequence transduction with recurrent neural networks," *ICML Workshop on Representation Learning*, 2012.

[4] A Vaswani, N Shazeer, N Parmar, J Uszkoreit, L Jones, A N Gomez, L Kaiser, and I Polosukhin, "Attention is all you need," *Advances in neural information processing systems*, vol. 30, 2017.

[5] Z. Tian, J. Yi, J. Tao, Y. Bai, and Z. Wen, "Self-attention transducers for end-to-end speech recognition," *Interspeech*, 2019.

[6] C 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.

[7] 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.

[8] F. Chang, J. Liu, M. Radfar, A. Mouchtaris, M. Omologo, A. Rastrow, and S. Kunzmann, "Context-aware transformer transducer for speech recognition," in *2021 IEEE Automatic*

Speech Recognition and Understanding Workshop (ASRU). IEEE, 2021, pp. 503–510.

[9] M. Radfar, R. Barnwal, R. V. Swaminathan, F. Chang, G. P Strimel, N. Susanj, and A. Mouchtaris, "Convrnn-t: Convolutional augmented recurrent neural network transducers for streaming speech recognition," *arXiv preprint arXiv:2209.14868*, 2022.

[10] A. Gulati, J. Qin, C. Chiu, N. Parmar, Y. Zhang, J. Yu, W. Han, S. Wang, Z. Zhang, et al., "Conformer: Convolution-augmented transformer for speech recognition," *arXiv preprint arXiv:2005.08100*, 2020.

[11] B. Li, A. Gulati, J. Yu, T. N. Sainath, C. Chiu, A. Narayanan, S. Chang, R. Pang, Y. He, et al., "A better and faster end-to-end model for streaming asr," in *ICASSP 2021-2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2021, pp. 5634–5638.

[12] T. N. Sainath, R. Prabhavalkar, S. Kumar, S. Lee, A. Kannan, D. Rybach, V. Schogol, P. Nguyen, B. Li, et al., "No need for a lexicon? evaluating the value of the pronunciation lexicon in end-to-end models," in *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2018, pp. 5859–5863.

[13] A. Gourav, L. Liu, A. Gandhe, Y. Gu, G. Lan, X. Huang, S. Kalmane, G. Tiwari, D. Filimonov, A. Rastrow, et al., "Personalization strategies for end-to-end speech recognition systems," in *ICASSP 2021-2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2021, pp. 7348–7352.

[14] K. M. Sathyendra, T. Muniyappa, F. Chang, J. Liu, J. Su, G. P. Strimel, A. Mouchtaris, and S. Kunzmann, "Contextual adapters for personalized speech recognition in neural transducers," in *ICASSP 2022-2022 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2022, pp. 8537–8541.

[15] Klaus Greff, Rupesh K Srivastava, Jan Koutník, Bas R Steunebrink, and Jürgen Schmidhuber, "Lstm: A search space odyssey," *IEEE transactions on neural networks and learning systems*, vol. 28, no. 10, pp. 2222–2232, 2016.

[16] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[17] Tao Lei, Yu Zhang, Sida I Wang, Hui Dai, and Yoav Artzi, "Simple recurrent units for highly parallelizable recurrence," *arXiv preprint arXiv:1709.02755*, 2017.

[18] Daniel Povey, Gaofeng Cheng, Yiming Wang, Ke Li, Hainan Xu, Mahsa Yarmohammadi, and Sanjeev Khudanpur, "Semi-orthogonal low-rank matrix factorization for deep neural networks.," in *Interspeech*, 2018, pp. 3743–3747.

[19] J. Macoskey, G. P. Strimel, and A. Rastrow, "Bifocal neural asr: Exploiting keyword spotting for inference optimization," in *ICASSP 2021-2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2021, pp. 5999–6003.

[20] Y. Shi, V. Nagaraja, C. Wu, J. Mahadeokar, D. Le, R. Prabhavalkar, A. Xiao, C. Yeh, J. Chan, C. Fuegen, O. Kalinli, and M. L. Seltzer, "Dynamic encoder transducer: A flexible solution for trading off accuracy for latency," 2021.

[21] Y Xie, J Macoskey, M Radfar, F Chang, B King, A Rastrow, A Mouchtaris, and G P Strimel, "Compute cost amortized transformer for streaming asr," *arXiv preprint arXiv:2207.02393*, 2022.

[22] Jonathan Macoskey, Grant P Strimel, Jinru Su, and Ariya Rastrow, "Amortized neural networks for low-latency speech recognition," *Interspeech*, 2021.

[23] Sinong Wang, Belinda Z Li, Madian Khabsa, Han Fang, and Hao Ma, "Linformer: Self-attention with linear complexity," *arXiv preprint arXiv:2006.04768*, 2020.

[24] K. Choromanski, V. Likhosherstov, D. Dohan, X. Song, A. Gane, T. Sarlos, P. Hawkins, J. Davis, A. Mohiuddin, et al., "Rethinking attention with performers," *arXiv preprint arXiv:2009.14794*, 2020.

[25] Hagen Soltau, Hank Liao, and Hasim Sak, "Neural speech recognizer: Acoustic-to-word lstm model for large vocabulary speech recognition," *arXiv preprint arXiv:1610.09975*, 2016.

[26] Y He, T N Sainath, R Prabhavalkar, I McGraw, R Alvarez, D Zhao, D Rybach, A Kannan, 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.

[27] Itay Hubara, Matthieu Courbariaux, Daniel Soudry, Ran El-Yaniv, and Yoshua Bengio, "Quantized neural networks: Training neural networks with low precision weights and activations," *The Journal of Machine Learning Research*, vol. 18, no. 1, pp. 6869–6898, 2017.

[28] J. Choi, S. Venkataramani, V. V. Srinivasan, K. Gopalakrishnan, Z. Wang, and P. Chuang, "Accurate and efficient 2-bit quantized neural networks," *Proceedings of Machine Learning and Systems*, vol. 1, pp. 348–359, 2019.

[29] H D Nguyen, A Alexandridis, and A Mouchtaris, "Quantization aware training with absolute-cosine regularization for automatic speech recognition.," in *Interspeech*, 2020, pp. 3366–3370.

[30] Michael Zhu and Suyog Gupta, "To prune, or not to prune: exploring the efficacy of pruning for model compression," *arXiv preprint arXiv:1710.01878*, 2017.

[31] Vlad Niculae and Mathieu Blondel, "A regularized framework for sparse and structured neural attention," *Advances in neural information processing systems*, vol. 30, 2017.

[32] Aurko Roy, Mohammad Saffar, Ashish Vaswani, and David Grangier, "Efficient content-based sparse attention with routing transformers," *Transactions of the Association for Computational Linguistics*, vol. 9, pp. 53–68, 2021.

[33] Taku Kudo, "Subword regularization: Improving neural network translation models with multiple subword candidates," *arXiv preprint arXiv:1804.10959*, 2018.

[34] G Pundak, T N Sainath, R Prabhavalkar, A Kannan, and D Zhao, "Deep context: end-to-end contextual speech recognition," in *2018 IEEE spoken language technology workshop (SLT)*. IEEE, 2018, pp. 418–425.

[35] J. Liu, R. V. Swaminathan, S. H. K. Parthasarathi, C. Lyu, A. Mouchtaris, and S. Kunzmann, "Exploiting large-scale teacher-student training for on-device acoustic models," Berlin, Heidelberg, 2021, p. 413–424, Springer-Verlag.

[36] R. Sennrich, B. Haddow, and A. Birch, "Neural machine translation of rare words with subword units," *arXiv preprint arXiv:1508.07909*, 2015.