INTERSPEECH 2023
20-24 August 2023, Dublin, Ireland

Modular Domain Adaptation for Conformer-Based Streaming ASR

Qiujia Li, Bo Li, Dongseong Hwang, Tara N. Sainath, Pedro M. Mengibar

Google LLC

{qiujia,boboli,dongseong,tsainath,pedro}@google.com

Abstract

Speech data from different domains has distinct acoustic and linguistic characteristics. It is common to train a single multidomain model such as a Conformer transducer for speech recognition on a mixture of data from all domains. However, changing data in one domain or adding a new domain would require the multidomain model to be retrained. To this end, we propose a framework called *modular domain adaptation* (MDA) that enables a single model to process multidomain data while keeping all parameters domain-specific, *i.e.*, each parameter is only trained by data from one domain. On a streaming Conformer transducer trained only on video caption data, experimental results show that an MDA-based model can reach similar performance as the multidomain model on other domains such as voice search and dictation by adding per-domain adapters and per-domain feed-forward networks in the Conformer encoder.

Index Terms: speech recognition, domain adaptation, Conformer transducer

1. Introduction

Automatic speech recognition (ASR) systems have been used across various applications, such as video captioning [1], dictation [2], voice search [3, 4], voice assistant [5] and telephony [6, 7, 8]. Each domain has its own acoustic and linguistic characteristics. For example, video captioning data has diverse acoustic environments and speaking styles; a dictation utterance is likely to have little background noise; voice search data tends to have short queries with named entities.

The performance of ASR models deteriorates significantly when the model is trained on a particular domain but evaluated on another [9]. Given sufficient model capacity and training data, it is desirable to build a single ASR model to serve all application domains. To ensure the model performs well for all domains, one simple and effective approach is to mix all data during training to obtain a multidomain model [10, 11]. Since all model parameters are shared across all domains, the multidomain approach has some shortcomings. First, the entire model needs to be retrained if the training data from a certain domain changes or a new domain is added. Second, finding the right balance to mix data from various domains is nontrivial. On the other end of the spectrum, it is also possible to build an ASR model for each domain. This means all parameters are domain-specific where each parameter is trained on data from a single domain. However, this is very expensive for training and serving as multiple models need to be maintained. Therefore, it is ideal to build a single model that serves all domains while keeping all model parameters domain-specific.

The contribution of this work is as follows. First, we proposed a framework called *modular domain adaptation* (MDA) that can process multidomain data with all parameters being domain-specific. As illustrated in Fig. 1, the backbone model is trained on a particular domain (in red), and all traffic passes through the common backbone model except certain parts of the model where the traffic is split and routed through the per-domain parameters. Second, we identified the most effective components of the Conformer backbone model to integrate per-domain parameters for parameter efficiency. Also, two types of per-domain parameters are explored, *i.e.*, per-domain components that replace the existing components in the backbone model and per-domain adapters that add lightweight adapter modules to modify intermediate representations. Third, the final recipe was validated on three different domains with a large amount of data and minimum word error rate (MWER) training [12]. The MDA-based model achieved similar performance as the multidomain model across all domains with 0.2–0.4% absolute degradation in word error rates (WERs). The number of per-domain parameters is 22% of the backbone model.

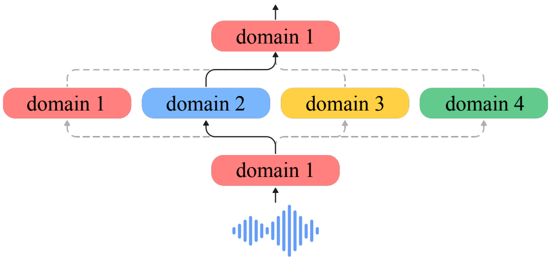


Figure 1: *Modular domain adaptation (MDA). Different colors indicate parameters trained on different domains. In this example, speech from domain 2 passes through the backbone trained on domain 1 except when per-domain parameters are available.*

Although domain adaptation has been widely studied for ASR [9] with various paradigms including input feature adaptation [13, 14, 15], model-based adaptation [16, 17, 18, 19, 20] and multi-task learning [21, 22], our work pays particular attention to modularity. Unlike general adaptation techniques, MDA satisfies the constraint that all model parameters are domain-specific. This brings many advantages of modularity [23]: similar functions of the ASR model are encoded with the same module while allocating distinct functions to per-domain parameters; per-domain parameters can be constructed separately and updated locally; parameter efficiency is much higher than fine-tuning the entire model or have multiple single-domain models.

In this paper, two methods under the framework of MDA are described in Sec. 2. Extensive experimental studies are then conducted in Sec. 3 & 4, which offer insights into the Conformer transducers, especially in terms of modularity. Conclusions are drawn in Sec. 5.

10.21437/Interspeech.2023-2215

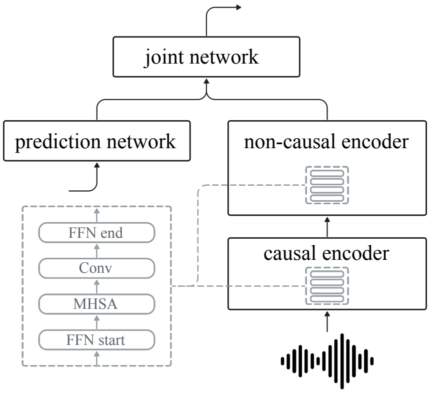


Figure 2: *Conformer transducer with cascaded encoders.*

2. Methods

In this section, two types of methods for MDA are introduced. Both approaches enable the inputs to be transformed differently based on their corresponding domains. Per-domain components replace a particular part of the backbone model with separate parameters for each domain. Per-domain adapters are additional domain-specific modules added to the backbone model.

2.1. Per-Domain Components

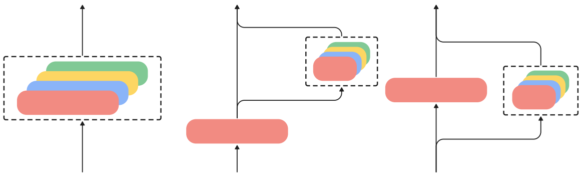
As illustrated in Fig. 2, a Conformer transducer [24] has an encoder with a stack of Conformer blocks and a decoder containing a prediction network and a joint network. A Conformer block contains a sequence of four modules - a feed-forward network (FFN), a multi-head self-attention (MHSA) module, a convolution module (Conv), and another feed-forward network. Each module in the Conformer block has a residual connection, which is omitted in the figure for simplicity. To enable processing streaming audio, a limited amount of future context is available for MHSA and Conv modules [25].

To perform MDA, some components of the Conformer transducer can be customized for each domain, *i.e.*, same architecture but with different parameters for each domain. As shown in Fig. 3(a), the input examples are routed through different components corresponding to their domains in different colors. The granularity of a "component" can vary greatly. A component can be a module in a Conformer block, an entire Conformer block, or even the entire encoder. At one extreme, it is equivalent to having a model for each domain if all components of the model are separate for each domain. Since some components will have a greater impact on the model quality than others, the effect of per-domain Conformer components will be examined in Sec. 4.

2.2. Per-Domain Adapters

Another approach to MDA is to have per-domain adapters in the Conformer model. Adapters are commonly used as efficient modules to adapt large foundation models for particular tasks without finetuning all model parameters [26, 27, 28, 29]. For MDA, per-domain adapters can be used to augment the hidden representations of the backbone model accordingly. For each domain, the adapter first uses a weight matrix $\mathbf{W}_{\text{down}} \in \mathbb{R}^{d\times b}$ to project the input representation $\mathbf{h} \in \mathbb{R}^{1\times d}$ to a lower-dimensional space with bottleneck dimension b, followed by a nonlinear activation function $f(\cdot)$, and then projects up to the original dimension using $\mathbf{W}_{\text{up}} \in \mathbb{R}^{b\times d}$. A residual connection is also applied around the adapter:

$$\mathbf{h} \leftarrow \mathbf{h} + f(\mathbf{h}\mathbf{W}_{\text{down}})\mathbf{W}_{\text{up}}. \tag{1}$$



(a) *Per-domain components.* (b) *Per-domain sequential adapters.* (c) *Per-domain parallel adapters.*

Figure 3: *Various modular domain adaptation (MDA) methods.*

There are different ways to configure adapters for the backbone model. The per-domain adapters can be either inserted in between two adjacent modules of the model, *i.e.*, sequential adapters as shown in Fig. 3(b), or positioned in parallel to some components of the backbone model, *i.e.*, parallel adapters as shown in Fig. 3(c). Experimental results for transfer learning have shown parallel adapters outperform sequential ones and modifying FFN representations is more effective than other modules in Transformer models [28]. In Sec. 4, experiments will be carried out on Conformers for MDA.

The aforementioned two approaches have their respective advantages. Per-domain components do not increase the computation cost because certain components of the backbone model are simply replaced. However, per-domain adapters have a small increase in computation cost as extra parameters are inserted into the backbone model. On the other hand, per-domain components generally need to store a much larger number of domain-specific parameters than per-domain adapters as a Conformer component is generally much larger than an adapter.

3. Experimental Setup

3.1. Data

The training set contains three domains: YouTube (YT), voice search (VS) and dictation (DT). In total, there are 755M utterances or 1.1M hours. For YT, the training set has around 220M utterances or 600k hours, and the test set has 20k utterances or 380 hours. For VS, the training set has 520M utterances or 490k hours, and the test set has 9.4k utterances or 12 hours. For DT, the training set has 10M utterances or 23k hours, and the test set has 17k utterances or 39 hours. Utterances from all domains were anonymized and hand-transcribed, except for the YT training set where the transcriptions were obtained in a semi-supervised fashion [30]. Multi-condition training [31], random data down-sampling to 8 kHz [32] and SpecAugment [33] were used during training to increase data diversity. 128-dimensional filterbank features were used as model inputs.

3.2. Model

The backbone model is a Conformer transducer model [24]. The encoder is based on the cascaded structure [34, 35] which consists of 7 causal Conformer blocks without future context followed by 10 non-causal Conformer blocks that use 900 ms of future acoustic context. The first two causal blocks do not have MHSA modules. The model dimensions are 512 and 640 for causal and non-causal encoders. For all Conformer blocks, MHSA modules have 8 attention heads; Conv modules have a convolution kernel size of 15; and the dimension of the FFN module is four times the model dimension. The causal and non-causal encoders have 47M and 99M parameters respectively. The outputs of both encoders are projected to 384 dimensions. There are two separate hybrid autoregressive transducer (HAT) decoders [36] for causal and non-causal encoders. The output vocabulary has 4096 wordpieces [37]. The joint network that combines features from the encoder and the embedding prediction network [38] has 640 units. Each decoder has 9.5M parameters. The total number of parameters of the entire model is 165M. During training, two encoders are selected randomly with the same probability and FastEmit loss [39] is applied with a scale of 0.005. All models were trained using the Lingvo toolkit [40] for 300k steps on 8 × 8 tensor processing units (TPUs) with a batch size of 4096.

4. Experimental Results

In this section, various MDA methods will be explored and compared with the multidomain baseline. Note that all WERs reported are from the non-causal decoder without using the end pointer (EP) [41] unless specified otherwise.

4.1. Baseline Conformer Model

The multidomain (MD) model is trained on a mixture of data from all five domains. Only for the MD baseline model, the domain ID encoded by a 16-dimensional one-hot vector is appended to the acoustic features as input to the model.

Table 1: *WERs (%) of the multidomain and single-domain models. YT-only model will be used as the backbone model.*

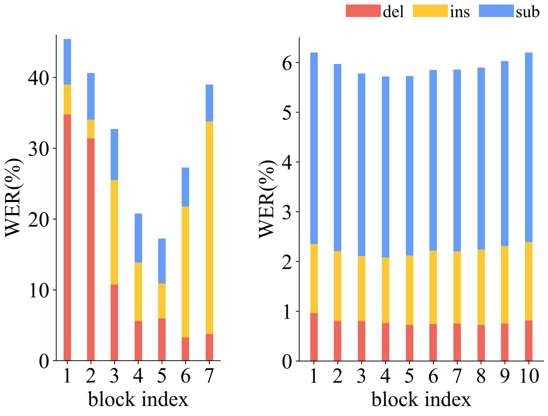
	YT	VS	DT
multidomain (MD)	14.0	4.7	3.4
YT-only	**13.9**	25.1	8.3
VS-only	43.0	**4.5**	14.7
DT-only	37.2	7.4	**3.7**

As shown in Table 1, the single-domain model performs well on in-domain testsets but struggles on out-of-domain testsets. The performance of the MD model is about the same as the single-domain models on their respective domains for YT and VS as they both have a large amount of training data. The MD model has a lower WER than the DT-only model on the DT testset because the size of the DT training set is an order of magnitude smaller than YT or VS. Since YT is considered to be the most diverse domain containing speech in a wide range of forms and acoustic conditions, the YT-only model is used as the backbone model in the rest of the paper. Note that the parameters of the backbone model will not be updated and per-domain parameters will be initialized randomly from scratch.

4.2. Per-Domain Components

In this section, various components in the backbone YT-only model are replaced by per-domain ones. For simplicity, only results on the VS testset are reported and the observations on the DT domain are similar.

First, per-domain Conformer encoder blocks are examined. In Fig. 4, each bar shows the WER of the model where a particular Conformer block is initialized from scratch and trained on VS while the rest of the backbone model is frozen. Fig. 4(a) shows that having per-domain Conformer blocks in the causal encoder is not effective as the WER is far from the multidomain baseline (4.7%). The trend line shows an interesting behavior where earlier blocks result in high deletion error rates but later blocks cause high insertion error rates. This may be attributed to the domain characteristics. The backbone YT-only model is trained to transcribe not only the primary speakers, but also speakers in the background. For VS, only the speech from the primary speaker is supposed to be recognized. Additionally, the average duration of a YT training utterance is 3–4 times greater than VS. We hypothesize that the earlier blocks of the encoder learn to suppress or enhance background signals. By training the earlier blocks on VS, the model tends to drop the acoustic content too aggressively whereas the later blocks cannot prevent the model to transcribe all speech content. However, the picture changes greatly for per-domain Conformer blocks in the non-causal encoder. As shown in Fig. 4(b), with the help of future context, the WERs drop to around 6% for all non-causal Conformer blocks. Further experiments show that having two per-domain Conformer blocks yields marginal improvement over a single one.



(a) *Causal encoder.* (b) *Non-causal encoder.*

Figure 4: *WERs on VS testset using per-domain Conformer blocks. Note that the scales of the y-axis are different. A causal / non-causal Conformer block has 6M / 10M parameters.*

Instead of using per-domain Conformer blocks, it is also possible to have per-domain Conformer modules across multiple Conformer blocks in the encoder. Table 2 shows the WERs when using different domain-specific modules, which are initialized from scratch and then trained on VS while the rest of the backbone YT-only model remains fixed. For each Conformer module, two WER results are presented where the "NC" column means only the non-causal encoder has per-domain modules and "C+NC" means both encoders have per-domain modules. Three conclusions can be drawn from the results in Table 2. First, by comparing the last row, "all params", with the MD baseline, most modeling power of the model resides in the encoders. Second, by comparing different Conformer modules in the "C+NC" column, the majority of the modeling capability is within the FFN start or FFN end module. Third, by comparing the "NC" and "C+NC" columns, non-causal blocks hold much more significance than causal blocks in terms of the recognition quality on the target domain. Moreover, compared with WERs shown in Fig. 4, having per-domain modules across all blocks is more effective than having a per-domain Conformer block.

In Table 3, per-domain decoder components are also tested. Having a per-domain prediction network is not useful as it has a similar WER as the backbone YT-only model (25.1%). This

Table 2: *WERs on VS testset using different per-domain encoder components. "C / NC" mean "causal / non-causal" encoders.*

	# trainable params	WER (%)	
	C / NC (M)	NC	C+NC
MD baseline		4.7	
FFN start	16.8 / 32.8	**4.9**	**4.7**
MHSA	4.2 / 20.5	5.5	5.3
Conv	5.0 / 12.4	5.4	5.0
FFN end	16.8 / 32.8	5.0	**4.7**
all params	46.8 / 99.1	4.8	4.6

is expected because the prediction network only provides the token embeddings of the two previous tokens for the joint network [38]. The joint network plays the most important role within the decoder as it transforms the encoder features, but the WER of the per-domain joint network is far from the MD baseline. Further experiments have also been carried out to combine domain-specific Conformer modules and per-domain joint networks, the improvement over the results in Table 2 is small, which further validates the finding that the main modeling capacity is within the encoder.

Table 3: *WERs on VS testset using different per-domain decoder components.*

	# trainable params (M)	WER (%)
MD baseline		4.7
prediction network	6.1	27.0
joint network	3.3	**9.2**
all params	9.3	9.3

4.3. Per-Domain Adapters

Using per-domain adapters is a more parameter-efficient approach for MDA as the number of additional domain-specific parameters is very small. Table 4 shows the WERs on the VS testset for both sequential adapters as in Fig. 3(b) and parallel adapters as in Fig. 3(c). Since the FFN module is the most important module in the Conformer as found in Sec. 4.2, both types of per-domain adapters are applied around all the FFN (start & end) modules in the encoder. There are three findings from Table 4. First, by comparing sequential and parallel adapters of the same bottleneck dimension, parallel adapters consistently outperform sequential ones. This is consistent with the literature in other fields [28]. Second, a larger bottleneck dimension helps for both types of adapters. Third, similar to Table 2, the adapters in the non-causal encoder are very significant and the adapters in the causal encoder bring small but consistent gains. Compared with the per-domain Conformer blocks in Sec. 4.2, per-domain adapters use much fewer domain-specific parameters while achieving lower WERs.

4.4. Final Recipe

Given that FFN modules have the most impact on WERs in the target domain and domain adapters are very parameter efficient, together with the observation that adapting the non-causal encoder is more effective than the causal encoder, we can take advantage of both MDA approaches. Table 5 shows the WERs on the VS testset with and without end pointing for three different setups. Depending on the budget of per-domain parameters and the tolerance of quality degradation compared to the MD baseline, a trade-off can be determined. As our final recipe, per-domain adapters are added to the causal encoder to ensure reasonable end-pointing latency [41, 35] by the causal encoder and per-domain FFN modules are used in the non-causal encoder to improve the recognition quality of the target domains.

Table 4: *WERs on VS testset using various per-domain adapters. "C / NC" mean "causal / non-causal" encoders.*

position	dim	# trainable params	WER (%)	
		C / NC (M)	NC	C+NC
MD baseline			4.7	
sequential	64	1.1 / 1.7	6.2	5.9
	128	2.1 / 3.3	5.9	5.7
	256	4.2 / 6.6	5.7	5.4
parallel	64	1.1 / 1.7	6.0	5.6
	128	2.1 / 3.3	5.8	5.4
	256	4.2 / 6.6	**5.5**	**5.1**

Table 5: *WERs on VS testset by various MDA methods with and without end pointing. PA refers to per-domain parallel adapters and FFN refers to per-domain FFN end.*

C	NC	# trainable params	WER (%)	
		C / NC (M)	w/o EP	w/ EP
MD baseline			4.7	6.1
PA	PA	4.2 / 6.6	5.1	6.6
PA	FFN	4.2 / 32.8	**4.7**	6.4
FFN	FFN	16.8 / 32.8	**4.7**	**6.3**

Finally, as shown in Table 6, the recipe is evaluated on all three domains and MWER training [12] is applied afterwards. During the MWER training, all model parameters are updated for the MD baseline, but only per-domain parameters are updated for the MDA recipe. The total number of per-domain parameters is around 37M and the WER degradation compared to the MD baseline after MWER training is 0.2–0.4% absolute.

Table 6: *WERs of the final recipe on all three domains and the effect of the MWER training. WERs on VS are with EP.*

	MWER	YT	VS	DT
MD baseline	✗	14.0	6.1	3.4
	✓	13.5	5.7	3.1
MDA recipe	✗	14.1	6.4	3.7
	✓	13.7	5.9	3.5

5. Conclusions

For Conformer models, modular domain adaptation (MDA) approaches such as per-domain feed-forward networks and per-domain adapters can achieve similar recognition quality as a multidomain model while having the benefits of modularity by using all domain-specific parameters for the entire model. In the future, we plan to explore other MDA methods to further improve the WERs and achieve better parameter efficiency.

6. References

[1] H. Liao, E. McDermott, and A. W. Senior, "Large scale deep neural network acoustic modeling with semi-supervised training data for YouTube video transcription," Proc. *ASRU*, 2013.

[2] J. Li, A. Mohamed, G. Zweig, and Y. Gong, "LSTM time and frequency recurrence for automatic speech recognition," Proc. *ASRU*, 2015.

[3] Y. Wang, D. Yu, Y. Ju, and A. Acero, "An introduction to voice search," *IEEE Signal Processing Magazine*, vol. 25, 2008.

[4] C. Shan, J. Zhang, Y. Wang, and L. Xie, "Attention-based end-to-end speech recognition on voice search," Proc. *ICASSP*, 2017.

[5] V. Kepuska and G. Bohouta, "Next-generation of virtual personal assistants (Microsoft Cortana, Apple Siri, Amazon Alexa and Google Home)," Proc. *CCWC*, 2018.

[6] T. Hain, P. C. Woodland, G. Evermann, and D. Povey, "The CU-HTK March 2000 Hub5E transcription system," Proc. *Speech Transcription Workshop*, 2000.

[7] W. Xiong, L. Wu, F. Alleva, J. Droppo, X. Huang, and A. Stolcke, "The Microsoft 2017 conversational speech recognition system," Proc. *ICASSP*, 2017.

[8] Z. Tuske, G. Saon, and B. Kingsbury, "On the limit of English conversational speech recognition," Proc. *Interspeech*, 2021.

[9] P. Bell, J. Fainberg, O. Klejch, J. Li, S. Renals, and P. Swietojanski, "Adaptation algorithms for neural network-based speech recognition: An overview," *IEEE Open Journal of Signal Processing*, vol. 2, 2020.

[10] A. Narayanan, A. Misra, K. C. Sim, G. Pundak, A. Tripathi, M. G. Elfeky, P. Haghani, T. Strohman, and M. Bacchiani, "Toward domain-invariant speech recognition via large scale training," Proc. *SLT*, 2018.

[11] W. Chan, D. S. Park, C. Lee, Y. Zhang, Q. V. Le, and M. Norouzi, "SpeechStew: Simply mix all available speech recognition data to train one large neural network," *arXiv:2104.02133*, 2021.

[12] R. Prabhavalkar, T. N. Sainath, Y. Wu, P. Nguyen, Z. Chen, C. Chiu, and A. Kannan, "Minimum word error rate training for attention-based sequence-to-sequence models," Proc. *ICASSP*, 2017.

[13] M. J. F. Gales, "Maximum likelihood linear transformations for HMM-based speech recognition," *Computer Speech & Language*, vol. 12, 1998.

[14] J. Fainberg, S. Renals, and P. Bell, "Factorised representations for neural network adaptation to diverse acoustic environments," Proc. *Interspeech*, 2017.

[15] T. N. Sainath, Y. He, B. Li, A. Narayanan, R. Pang, A. Bruguier, S. Chang, W. Li, R. Álvarez, Z. Chen *et al.*, "A streaming on-device end-to-end model surpassing server-side conventional model quality and latency," Proc. *ICASSP*, 2020.

[16] J. Li, M. L. Seltzer, X. Wang, R. Zhao, and Y. Gong, "Large-scale domain adaptation via teacher-student learning," Proc. *Interspeech*, 2017.

[17] T. Asami, R. Masumura, Y. Yamaguchi, H. Masataki, and Y. Aono, "Domain adaptation of DNN acoustic models using knowledge distillation," Proc. *ICASSP*, 2017.

[18] V. Manohar, P. Ghahremani, D. Povey, and S. Khudanpur, "A teacher-student learning approach for unsupervised domain adaptation of sequence-trained ASR models," Proc. *SLT*, 2018.

[19] L. Samarakoon, B. K. Mak, and A. Y. S. Lam, "Domain adaptation of end-to-end speech recognition in low-resource settings," Proc. *SLT*, 2018.

[20] K. C. Sim, A. Narayanan, A. Misra, A. Tripathi, G. Pundak, T. N. Sainath, P. Haghani, B. Li, and M. Bacchiani, "Domain adaptation using factorized hidden layer for robust automatic speech recognition," Proc. *Interspeech*, 2018.

[21] P. Denisov, N. T. Vu, and M. Ferras, "Unsupervised domain adaptation by adversarial learning for robust speech recognition," Proc. *ITG Symposium on Speech Communication*, 2018.

[22] Z. Meng, J. Li, Y. Gong, and B. Juang, "Adversarial teacher-student learning for unsupervised domain adaptation," Proc. *ICASSP*, 2018.

[23] J. Pfeiffer, S. Ruder, I. Vulić, and E. M. Ponti, "Modular deep learning," *arXiv:2302.11529*, 2023.

[24] A. Gulati, J. Qin, C. Chiu, N. Parmar, Y. Zhang, J. Yu, W. Han, S. Wang, Z. Zhang, Y. Wu, and R. Pang, "Conformer: Convolution-augmented Transformer for speech recognition," Proc. *Interspeech*, 2020.

[25] B. Li, A. Gulati, J. Yu, T. N. Sainath, C. Chiu, A. Narayanan, S. Chang, R. Pang, Y. He, J. Qin *et al.*, "A better and faster end-to-end model for streaming ASR," Proc. *ICASSP*, 2020.

[26] N. Houlsby, A. Giurgiu, S. Jastrzebski, B. Morrone, Q. de Laroussilhe, A. Gesmundo, M. Attariyan, and S. Gelly, "Parameter-efficient transfer learning for NLP," Proc. *ICML*, 2019.

[27] J. Pfeiffer, A. Kamath, A. Rücklé, K. Cho, and I. Gurevych, "AdapterFusion: Non-destructive task composition for transfer learning," Proc. *EACL*, 2021.

[28] J. He, C. Zhou, X. Ma, T. Berg-Kirkpatrick, and G. Neubig, "Towards a unified view of parameter-efficient transfer learning," Proc. *ICLR*, 2022.

[29] B. Li, D. Hwang, Z. Huo, J. Bai, G. Prakash, T. N. Sainath, K. C. Sim, Y. Zhang, W. Han, T. Strohman, and F. Beaufays, "Efficient domain adaptation for speech foundation models," Proc. *ICASSP*, 2023.

[30] Y. Zhang, D. S. Park, W. Han, J. Qin, A. Gulati, J. Shor, A. Jansen, Y. Xu, Y. Huang, S. Wang *et al.*, "BigSSL: Exploring the frontier of large-scale semi-supervised learning for automatic speech recognition," *IEEE Journal of Selected Topics in Signal Processing*, vol. 16, 2021.

[31] C. Kim, A. Misra, K. K. Chin, T. Hughes, A. Narayanan, T. N. 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," Proc. *Interspeech*, 2017.

[32] J. Li, D. Yu, J. T. Huang, and Y. Gong, "Improving wideband speech recognition using mixed-bandwidth training data in CD-DNN-HMM," Proc. *SLT*, 2012.

[33] D. S. Park, W. Chan, Y. Zhang, C. Chiu, B. Zoph, E. D. Cubuk, and Q. V. Le, "SpecAugment: A simple data augmentation method for automatic speech recognition," Proc. *Interspeech*, 2019.

[34] A. Narayanan, T. N. Sainath, R. Pang, J. Yu, C. Chiu, R. Prabhavalkar, E. Variani, and T. Strohman, "Cascaded encoders for unifying streaming and non-streaming ASR," Proc. *ICASSP*, 2020.

[35] T. N. Sainath, Y. He, A. Narayanan, R. Botros, W. Wang, D. Qiu, C. Chiu, R. Prabhavalkar, A. Gruenstein, A. Gulati *et al.*, "Improving the latency and quality of cascaded encoders," Proc. *ICASSP*, 2022.

[36] E. Variani, D. Rybach, C. Allauzen, and M. Riley, "Hybrid autoregressive transducer (HAT)," Proc. *ICASSP*, 2020.

[37] M. Schuster and K. Nakajima, "Japanese and Korean voice search," Proc. *ICASSP*, 2012.

[38] R. Botros, T. N. Sainath, R. David, E. Guzman, W. Li, and Y. He, "Tied & reduced RNN-T decoder," Proc. *Interspeech*, 2021.

[39] J. Yu, C. Chiu, B. Li, S. Chang, T. N. Sainath, Y. He, A. Narayanan, W. Han, A. Gulati, Y. Wu, and R. Pang, "FastEmit: Low-latency streaming ASR with sequence-level emission regularization," Proc. *ICASSP*, 2020.

[40] J. Shen, P. Nguyen, Y. Wu, Z. Chen, M. X. Chen, Y. Jia, A. Kannan, T. N. Sainath, Y. Cao, C. Chiu *et al.*, "Lingvo: A modular and scalable framework for sequence-to-sequence modeling," *arXiv:1902.08295*, 2019.

[41] B. Li, S. Chang, T. N. Sainath, R. Pang, Y. He, T. Strohman, and Y. Wu, "Towards fast and accurate streaming end-to-end ASR," Proc. *ICASSP*, 2020.