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

Improving Customization of Neural Transducers by Mitigating Acoustic Mismatch of Synthesized Audio

Gakuto Kurata[1], *George Saon*[2], *Brian Kingsbury*[2], *David Haws*[2], *Zoltán Tüske*[2]

[1]IBM Research - Tokyo, Japan
[2]IBM T. J. Watson Research Center, USA

gakuto@jp.ibm.com, {gsaon,bedk,dhaws}@us.ibm.com, zoltan.tuske@ibm.com

Abstract

Customization of automatic speech recognition (ASR) models using text data from a target domain is essential to deploying ASR in various domains. End-to-end (E2E) modeling for ASR has made remarkable progress, but the advantage of E2E modeling, where all neural network parameters are jointly optimized, is offset by the challenge of customizing such models. In conventional hybrid models, it is easy to directly modify a language model or a lexicon using text data, but this is not true for E2E models. One popular approach for customizing E2E models uses audio synthesized from the target domain text, but the acoustic mismatch between the synthesized and real audio can be problematic. We propose a method that avoids the negative effect of synthesized audio by (1) adding a mapping network before the encoder network to map the acoustic features of the synthesized audio to those of the source domain, (2) training the added mapping network using text and synthesized audio from the source domain while freezing all layers in the E2E model, (3) training the E2E model with text and synthesized audio from the target domain, and (4) removing the added mapping network when decoding real audio from the target domain. Experiments on customizing RNN Transducer and Conformer Transducer models demonstrate the advantage of the proposed method over encoder freezing, a popular customization method for E2E models.

Index Terms: End-to-end speech recognition, customization, recurrent neural network transducer (RNN-T), conformer transducer (Conformer-T), synthesized audio

1. Introduction

End-to-end (E2E) automatic speech recognition (ASR) with neural networks is making remarkable progress [1–3]. In E2E ASR, all neural network parameters are jointly optimized with paired audio and reference text, which contributes to superior accuracy, efficient decoding, and ease of training. By contrast, the conventional hybrid ASR uses three modules, an Acoustic Model (AM), a Language Model (LM), and a lexicon (pronunciation dictionary or model), and these modules are independently trained, making training and decoding more complex.

When ASR systems are deployed to various domains, flexible customizability is essential [4]. Customization with only text data from the target domain is especially important because it is much easier to prepare text data than paired real audio and text data. The modularity of the conventional hybrid ASR allows for independent customization of the necessary modules: target domain text data can be used to customize the LM and lexicon. Because E2E ASR models are not modular, it is not straightforward to directly modify the language model and the lexicon portions of an E2E model with only text data.

One popular approach to customize E2E models with text data uses audio data synthesized from the target domain text data [5–9]. While speech synthesis technology is also making remarkable progress recently [10–14], there still exists acoustic mismatch between synthesized and real audio, which can have a negative effect in customizing E2E ASR. To mitigate this problem, freezing the acoustic encoder part of the E2E model, such as the encoder network in neural transducers, has been explored [5, 7].

Another approach customizes an external LM that is jointly used with the E2E model during decoding. Some external LM fusion techniques, such as shallow fusion [15–17], density ratio fusion [18], and their enhancements [19,20] have been explored in the context of customization. While external LM fusion is simple, it requires additional computation during decoding and the performance improvement can be smaller than the customization of the E2E model itself using synthesized audio [7].

For E2E ASR, various modeling approaches including Connectionist Temporal Classification (CTC) [21–24], attention-based encoder-decoder [1, 25–27], and neural transducers such as Recurrent Neural Network Transducer (RNN-T) [28, 29], Transformer Transducer [30, 31], and Conformer Transducer (Conformer-T) [2] have been proposed. Among these approaches, neural transducers do not assume conditional independence between predictions at different time steps, unlike CTC, and enable monotonic decoding in time, unlike attention-based encoder-decoder, which are attractive characteristics for deployment of E2E ASR in various industry settings. These advantages motivated us to focus on neural transducers, more specifically, RNN-T and Conformer-T, in this paper.

Hereafter, we focus on directly customizing neural transducers, and exclude customization of external LMs due to the higher inference cost of LM fusion. Our approach improves on the popular approach of using synthesized audio by avoiding the negative impact that training on mismatched synthetic speech has on an E2E model. To this end, we (1) add an additional *mapping network* before the encoder network to map the acoustic features of the synthesized audio to those of the source domain, (2) train the added mapping network using text and synthesized audio from the source domain while freezing all layers in the E2E model, (3) train the E2E model with text and synthesized audio from the target domain, and (4) remove the added mapping network when decoding real audio from the target domain. In this proposed method, the mapping network is a form of feature-space transformation [32] that is trained with the transducer loss given a fixed neural transducer model. Note that this mapping network is used only during customization and is discarded for decoding real audio in the target domain.

Experimental results on customizing RNN-T and Conformer-T models to various domains demonstrate the

http://dx.doi.org/10.21437/Interspeech.2021-1656

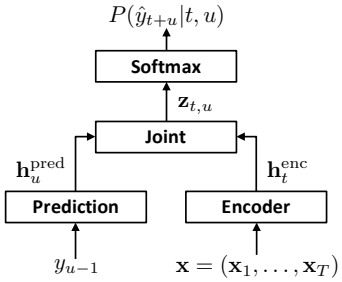

Figure 1: *Schematic diagram of neural transducer.*

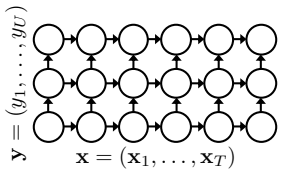

Figure 2: *Posterior lattice for neural transducer.*

advantage of the proposed method over encoder freezing, a popular customization method for E2E models.

2. Neural Transducers

Let $\mathbf{y} = (y_1, \cdots, y_U)$ denote a length-U sequence of target output symbols drawn from target symbol set $\mathcal{Y}$. Elements of $\mathcal{Y}$ may be letters, phonemes, graphemes, wordpieces, and so on [33, 34]. In this work, we use letters. Let $\mathbf{x} = (\mathbf{x}_1, \cdots, \mathbf{x}_T)$ denote a sequence of acoustic feature vectors over T time steps. In neural transducer modeling, an extra blank symbol ϕ is introduced to expand the length-U sequence $\mathbf{y}$ to a set of length-$(T+U)$ sequences $\Phi(\mathbf{y})$. Each sequence $\hat{\mathbf{y}} \in \Phi(\mathbf{y})$ is one of the transducer alignments between $\mathbf{x}$ and $\mathbf{y}$, where the elements of $\hat{\mathbf{y}} = (\hat{y}_1, \cdots, \hat{y}_{T+U})$ are drawn from $\mathcal{Y} \cup \{\phi\}$. The transducer loss is the negative log-posterior of the target symbol sequence, marginalized over all possible transducer alignments: $\mathcal{L}_{\text{T}} = -\log P(\mathbf{y}|\mathbf{x}) = -\log \sum_{\hat{\mathbf{y}} \in \Phi(\mathbf{y})} P(\hat{\mathbf{y}}|\mathbf{x})$.

As shown in Figure 1, the encoder network serves as an acoustic model that converts the input features $\mathbf{x}$ to a sequence of projected embedding vectors $\mathbf{h}_t^{\text{enc}}$ with the same length T. The prediction network works as an LM, producing a sequence of projected embedding vectors $\mathbf{h}_u^{\text{pred}}$ conditioned on the previous non-blank predictions, $(y_1, \cdots, y_{u-1})$. Note that the division of the encoder, prediction, and joint networks is slightly different from the most popular one, and linear projections to match the dimensions of both embedding sequences are applied inside the encoder and prediction networks. The joint network outputs an embedding $\mathbf{z}_{t,u}$ by combining the output from the encoder network $\mathbf{h}_t^{\text{enc}}$ and the output from the prediction network $\mathbf{h}_u^{\text{pred}}$. A common implementation of the joint network sums the projected embeddings followed by a linear transformation as $\mathbf{z}_{t,u} = \mathbf{W}\psi(\mathbf{h}_t^{\text{enc}} + \mathbf{h}_u^{\text{pred}}) + \mathbf{b}$ where ψ is hyperbolic tangent, $\mathbf{W}$ is a weight matrix, and $\mathbf{b}$ is a bias vector. Another promising implementation, which is used in this paper, is multiplicative integration of the two streams followed by a linear transformation as $\mathbf{z}_{t,u} = \mathbf{W}\psi(\mathbf{h}_t^{\text{enc}} \odot \mathbf{h}_u^{\text{pred}}) + \mathbf{b}$ where $\odot$ denotes elementwise multiplication [35, 36]. A softmax operation is applied to $\mathbf{z}_{t,u}$ to calculate a posterior distribution $P(\hat{y}_{t+u}|t,u)$ over the set $\mathcal{Y} \cup \{\phi\}$. Thus, $P(\hat{y}_{t+u}|t,u)$ defines a posterior lattice as shown in Figure 2, where each node represents the posterior distribution. With these definitions, neural transducer training is achieved by minimizing the transducer loss $\mathcal{L}_{\text{T}}$, which is efficiently computed using a forward-backward algorithm [28, 37].

RNN-T models are neural transducer models that use RNN (usually Long Short-Term Memory) encoder networks; Conformer-T models use Conformer encoder networks.

3. Proposed Customization for Neural Transducers

We explore a method to customize neural transducer models using only text data from the target domain, which is a common scenario for ASR deployments. Because the prediction network of a transducer model works as an LM, customizing the prediction network to the target domain is a widely used approach. In neural transducer models, the prediction network is jointly optimized with other networks; thus, it is not easy to directly modify only the prediction network with text data. Synthesizing audio from the target domain text to customize neural transducer models is a popular approach, but the acoustic mismatch between the synthesized and real audio must be addressed. *Encoder freezing* tackles this problem by freezing the encoder network and updating only the prediction network to avoid contaminating the encoder network with the synthesized audio [5, 7].

In this paper, we address this problem by explicitly transforming the features of the synthesized audio to fit the encoder network of the baseline transducer model. As shown in Figure 3, there are four steps in the proposed method. We call the domain of the baseline transducer model's training data the *source* domain and the new domain to which the model needs to be customized the *target* domain. As explained in Section 2, transducer models have three components, the joint, prediction, and encoder networks, denoted in Figure 3 by **J**, **P**, and **E**, respectively. Note that the baseline E2E model consists of **J**, **P**, and **E**, and we customize this model using the text data.

Step 1 We add a mapping network before the encoder network to transform the acoustic features of the synthesized audio. We denote the mapping network as **M** in Figure 3.

Step 2 We synthesize the audio from the reference text in the source domain. We then train the mapping network to minimize the transducer loss on the paired synthesized audio and reference text while keeping the other networks fixed. Please note that the reference text has already been used to train the baseline transducer model. Thus the prediction network has "seen" this reference text and the loss contributed by the prediction network should be negligible. Because all networks other than the mapping network are fixed, the mapping network learns to transform the acoustic features of the synthesized audio such that the encoder network can yield appropriate embeddings. We denote the trained mapping network as **M'**.

Step 3 We synthesize the audio from the reference text in the target domain, and then we update only the prediction network using the paired synthesized audio and reference text while keeping the other networks fixed. The acoustic features of the synthesized audio are transformed by **M'**, which was trained in the previous step, before being fed to the encoder network, mitigating the mismatch of the acoustic features of the synthesized audio to the source domain. We denote the trained prediction network as **P'**.

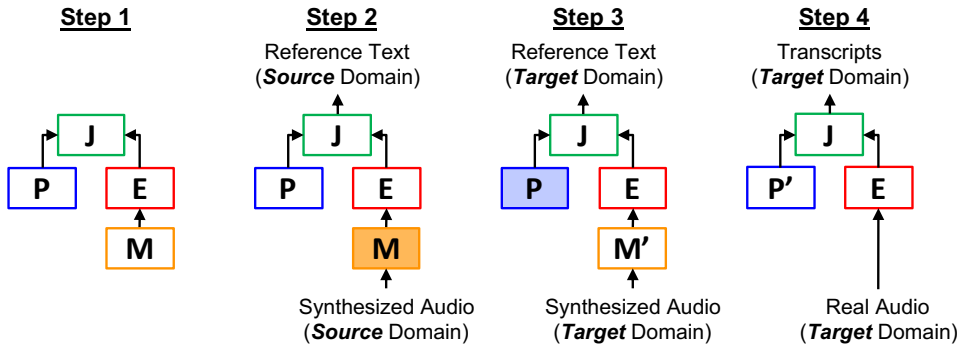

Figure 3: *Proposed customization for neural transducers. J, P, E, and M represent the Joint, Prediction, Encoder, and Mapping networks respectively. In step 1, the mapping network is added before the encoder network . In steps 2 and 3, the network filled in color is trained and others are fixed. In step 2, the mapping network is updated using paired reference text in the source domain and synthesized audio while the other networks are fixed. In step 3, the prediction network is updated using paired reference text in the target domain and synthesized audio transformed by the trained mapping network while the other networks are fixed. In step 4, the mapping network is removed and real audio in the target domain is recognized.*

Step 4 When decoding real audio data in the target domain, we remove the mapping network. The customized transducer model consisting of **J**, **P'**, and **E** in the same topology as the baseline model is used for decoding.

In this proposed method, the mapping network is a feature-space transformation [32] trained using the transducer loss given the fixed transducer model. By using the reference text from the source domain, the mapping network is trained to minimize the loss due to acoustic mismatch. Note that this mapping network is used only during customization and is not used for decoding real audio in the target domain.

Instead of adding the mapping network, updating the encoder network using the reference text from the source domain and their synthesized audio is possible, but the encoder network trained with the synthesized audio is used for decoding real audio in the end. By preparing the mapping network externally, we can keep using the original encoder network with the updated prediction network.

4. Experiments

To evaluate the performance of the proposed method, we conducted experiments to customize strong RNN-T and Conformer-T baseline models. The models are trained from a public data set and customized to three domains including another public data set (Broadcast News: BN) and two internal call-center data sets.

To train the baseline model, we used the *Switchboard 2000* collection of public American English telephone conversations, comprising 262 hours of Switchboard 1, 1698 hours of Fisher, and 15 hours of CallHome audio and their transcripts [38]. One of the target domains is BN, and we used the Defense Advanced Research Projects Agency (DARPA) Effective Affordable Reusable Speech-to-Text (EARS) DEV04F set which contains approximately 2 hours of data as a test set [39,40]. The BN audio data was downsampled to 8kHz in advance. To customize the baseline model to the BN domain, we randomly selected 3285 sentences from several BN related sources [39]. The two other target domains are conversations from two different internal call centers that have different speaking styles. We named them CC-A and CC-B, and both test sets contain approximately 1.5 hours of data. To customize the baseline model to CC-A and CC-B, we used manual transcriptions of approximately 1500 utterances collected previously from CC-A and CC-B.

For acoustic features, we used 40-dimensional log-Mel filterbank energies, their delta, and double-delta coefficients with frame stacking and a skipping rate of 2 [41], which results in 240-dimensional features. Training of the baseline model followed a recipe [36] that includes SpecAugment [42], sequence noise injection [43], DropConnect [44], and speed and tempo perturbation [45]. The architectures of the RNN-T and Conformer-T models are as follows. The RNN-T consists of an encoder network of 6 bidirectional LSTM layers with 640 cells per layer per direction followed by a projection from a 1280-dimensional acoustic embedding to 256 dimensions, a prediction network of a single unidirectional LSTM layer with 768 cells followed by a projection from a 768-dimensional linguistic embedding to 256 dimensions, a joint network using multiplicative integration and hyperbolic tangent followed by a projection, and softmax layer to represent 45 characters and an extra blank symbol. The Conformer-T model replaces the 6 bidirectional LSTM layers with 10 Conformer blocks with a 512-dimensional feed forward module, a 31-kernel convolution block, and 8 64-dimensional attention heads [2]. Total number of parameters of Conformer-T is approximately 30% greater than that of RNN-T. Letter-based decoding uses an efficient beam search algorithm, alignment-length synchronous decoding [46], and Word Error Rate (WER) is used as the evaluation metric.

In addition to the uncustomized baseline model, we also test encoder freezing customization for comparison. For the proposed method, we tested linear and nonlinear mapping networks. Audio is synthesized using one LPCNet male voice from IBM's text-to-speech service [13]. The experimental details are as follows.

Encoder freezing For each target domain, we synthesized audio data from the text in that domain. Following previous literature, we used the paired synthesized audio and text to update the prediction network while freezing the encoder and joint networks. All models were trained for 10 epochs using stochastic gradient descent with Nesterov momentum of 0.9 and a learning rate starting from 0.1 or 0.01, annealing by $\sqrt{0.5}$ per epoch after the 5^{th} epoch. We reserved approximately 10% of each data set as held-out data, selected the learning rate based on the

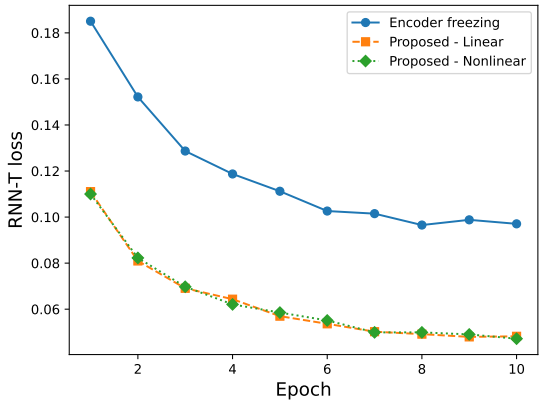

Figure 4: *RNN-T loss on paired synthesized audio and text in the target domain.*

Table 1: *Word error rates for baseline and customized RNN-T models. [%]*

	BN	CC-A	CC-B
Baseline RNN-T	18.9	30.7	37.3
Encoder freezing	18.8	29.8	38.1
Proposed method			
Linear	**18.6**	29.2	36.8
Nonlinear	**18.6**	**29.0**	**36.6**

Table 2: *Word error rates for baseline and customized Conformer-T models. [%]*

	BN	CC-A	CC-B
Baseline Conformer-T	15.5	25.8	33.6
Encoder freezing	15.4	25.8	33.4
Proposed method			
Linear	15.3	25.4	**33.0**
Nonlinear	**15.2**	**25.2**	33.1

held-out WER, and reported the WER for the remaining 90% of the data.

Proposed method We tested *linear* and *nonlinear* versions of the mapping network. In the linear network, input feature $\mathbf{x}_t$ at time step t is converted to $\mathbf{x'}_t$ via $\mathbf{x'}_t = \mathbf{W}\mathbf{x} + \mathbf{b}$ where $\mathbf{W} \in \mathbb{R}^{240\times240}$ and $\mathbf{b} \in \mathbb{R}^{240}$ are learnable parameters. In the nonlinear network, the input feature is converted to $\mathbf{x'}_t$ via $\mathbf{x'}_t = \mathbf{W_2}(\psi(\mathbf{W_1}\mathbf{x} + \mathbf{b_1})) + \mathbf{b_2}$, where $\mathbf{W_1}, \mathbf{W_2} \in \mathbb{R}^{240\times240}$ and $\mathbf{b_1}, \mathbf{b_2} \in \mathbb{R}^{240}$ are learnable parameters, and ψ is hyperbolic tangent. In step 2 in Figure 3, we randomly selected 1742 reference texts from the training data (Switchboard 2000) of the baseline model and synthesized corresponding audio. Using this paired data from the source domain, all weight matrices and bias vectors in the mapping network were updated. In step 3, we updated the prediction network with the synthesized audio for the target domain while freezing other networks. For both step 2 and step 3, we used the same learning rate control as explained in the encoder freezing.

Please note that in our definition of encoder, prediction, and joint networks, learnable parameters in the joint network exist only in the final projection before the softmax operation as explained in Section 2. Thus, while we fixed the joint network, in all cases, the linear projection layers before the multiplicative integration of projected acoustic and linguistic embeddings were updated within the encoder and the prediction networks.

4.1. RNN Transducer

First we conducted customization experiments on the baseline RNN-T model. Using customization to the CC-A domain as an example, we computed the RNN-T loss for the paired synthesized audio and text from the target domain during customization with the encoder freezing and the proposed method (step 3) and show them in Figure 4. Note that exactly the same data were used in the same order in all cases. Because the mapping network trained in step 2 mitigated the acoustic mismatch caused by the synthesized audio in the proposed method, the RNN-T loss of the proposed method is smaller than that of encoder freezing. Comparing the linear and nonlinear transformation in the proposed method, we do not see a significant difference.

We then computed WERs for the three test sets before and after the customization and present the results in Table 1. For all cases, the WER improvement from the proposed method is greater than the improvement from encoder freezing. Comparing the linear and nonlinear transformation in the proposed method, the nonlinear transformation was better than the linear transformation in the CC-A and CC-B cases, but the difference was not significant. It is also noteworthy that in the CC-B case, we saw accuracy degradation with encoder freezing, while we confirmed a significant improvement using the proposed method.

The smaller RNN-T loss during the customization process and the improved WERs on three test cases demonstrate the efficacy of the proposed method.

4.2. Conformer Transducer

Next we customized the baseline Conformer-T model to the three domains. As with the RNN-T, we computed WERs over the three test sets before and after customization and present the results in Table 2. While the baseline WERs of the Conformer-T model are much better than those of the RNN-T model in Table 1, we again find a greater improvement using the proposed method versus encoder freezing, which supports the advantage of the proposed method.

5. Conclusion

In this paper, we proposed an improved customization method for neural transducers. By introducing an additional mapping network before the encoder network only during the customization process, the acoustic mismatch due to the synthesized audio can be mitigated. Experiments on customizing strong baseline RNN-T and Conformer-T models trained with English conversational telephone speech to the domains of public BN and two different internal call-center test cases demonstrated the advantage of the proposed method over encoder freezing, a popular customization alternative.

Another potential advantage of the proposed method is that it theoretically works with single-speaker speech synthesis. In future work, we will compare the proposed method with single-speaker speech synthesis to encoder freezing with multi-speaker speech synthesis.

6. References

[1] Z. Tüske, G. Saon *et al.*, "Single headed attention based sequence-to-sequence model for state-of-the-art results on Switchboard," in *Proc. INTERSPEECH*, 2020, pp. 551–555.

[2] A. Gulati, J. Qin *et al.*, "Conformer: Convolution-augmented transformer for speech recognition," in *Proc. INTERSPEECH*, 2020, pp. 5036–5040.

[3] P. Guo, F. Boyer *et al.*, "Recent developments on ESPnet toolkit boosted by Conformer," *arXiv preprint arXiv:2010.13956*, 2020.

[4] P. Bell, J. Fainberg *et al.*, "Adaptation algorithms for speech recognition: An overview," *arXiv preprint arXiv:2008.06580*, 2020.

[5] M. Mimura, S. Ueno *et al.*, "Leveraging sequence-to-sequence speech synthesis for enhancing acoustic-to-word speech recognition," in *Proc. SLT*, 2018, pp. 477–484.

[6] K. C. Sim, F. Beaufays *et al.*, "Personalization of end-to-end speech recognition on mobile devices for named entities," in *Proc. ICASSP*, 2019, pp. 23–30.

[7] J. Li, R. Zhao *et al.*, "Developing RNN-T models surpassing high-performance hybrid models with customization capability," in *Proc. INTERSPEECH*, 2020, pp. 3590–3594.

[8] E. Sharma, G. Ye *et al.*, "Adaptation of RNN transducer with text-to-speech technology for keyword spotting," in *Proc. ICASSP*, 2020, pp. 7484–7488.

[9] X. Zheng, Y. Liu *et al.*, "Using synthetic audio to improve the recognition of Out-Of-Vocabulary words in end-to-end ASR systems," *arXiv preprint arXiv:2011.11564*, 2020.

[10] A. v. d. Oord, S. Dieleman *et al.*, "WaveNet: A generative model for raw audio," *arXiv preprint arXiv:1609.03499*, 2016.

[11] N. Kalchbrenner, E. Elsen *et al.*, "Efficient neural audio synthesis," in *ICML*, 2018, pp. 2410–2419.

[12] J.-M. Valin and J. Skoglund, "LPCNet: Improving neural speech synthesis through linear prediction," in *Proc. ICASSP*, 2019, pp. 5891–5895.

[13] Z. Kons, S. Shechtman *et al.*, "High quality, lightweight and adaptable TTS using LPCNet," in *Proc. INTERSPEECH*, 2019, pp. 176–180.

[14] N. Chen, Y. Zhang *et al.*, "WaveGrad: Estimating gradients for waveform generation," *arXiv preprint arXiv:2009.00713*, 2020.

[15] C. Gulcehre, O. Firat *et al.*, "On using monolingual corpora in neural machine translation," *arXiv preprint arXiv:1503.03535*, 2015.

[16] A. Kannan, Y. Wu *et al.*, "An analysis of incorporating an external language model into a sequence-to-sequence model," in *Proc. ICASSP*, 2018, pp. 5824–5828.

[17] A. Zeyer, K. Irie *et al.*, "Improved training of end-to-end attention models for speech recognition," in *Proc. INTERSPEECH*, 2018, pp. 7–11.

[18] E. McDermott, H. Sak *et al.*, "A density ratio approach to language model fusion in end-to-end automatic speech recognition," in *Proc. ASRU*, 2019, pp. 434–441.

[19] Z. Meng, S. Parthasarathy *et al.*, "Internal language model estimation for domain-adaptive end-to-end speech recognition," in *Proc. SLT*, 2021, pp. 243–250.

[20] E. Variani, D. Rybach *et al.*, "Hybrid autoregressive transducer (HAT)," in *Proc. ICASSP*, 2020, pp. 6139–6143.

[21] A. Graves, S. Fernández *et al.*, "Connectionist temporal classification: labelling unsegmented sequence data with recurrent neural networks," in *Proc. ICML*, 2006, pp. 369–376.

[22] Y. Miao, M. Gowayyed *et al.*, "EESEN: End-to-end speech recognition using deep RNN models and WFST-based decoding," in *Proc. ASRU*, 2015, pp. 167–174.

[23] H. Sak, A. Senior *et al.*, "Learning acoustic frame labeling for speech recognition with recurrent neural networks," in *Proc. ICASSP*, 2015, pp. 4280–4284.

[24] K. Audhkhasi, B. Kingsbury *et al.*, "Building competitive direct acoustics-to-word models for English conversational speech recognition," in *Proc. ICASSP*, 2018, pp. 4759–4763.

[25] J. K. Chorowski, D. Bahdanau *et al.*, "Attention-based models for speech recognition," in *Proc. NIPS*, 2015, pp. 577–585.

[26] D. Bahdanau, J. Chorowski *et al.*, "End-to-end attention-based large vocabulary speech recognition," in *Proc. ICASSP*, 2016, pp. 4945–4949.

[27] W. Chan, N. Jaitly *et al.*, "Listen, attend and spell: A neural network for large vocabulary conversational speech recognition," in *Proc. ICASSP*, 2016, pp. 4960–4964.

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

[29] A. Graves, A. Mohamed *et al.*, "Speech recognition with deep recurrent neural networks," in *Proc. ICASSP*, 2013, pp. 6645–6649.

[30] C.-F. Yeh, J. Mahadeokar *et al.*, "Transformer-transducer: End-to-end speech recognition with self-attention," *arXiv preprint arXiv:1910.12977*, 2019.

[31] Q. Zhang, H. Lu *et al.*, "Transformer transducer: A streamable speech recognition model with transformer encoders and RNN-T loss," in *Proc. ICASSP*, 2020, pp. 7829–7833.

[32] M. J. Gales, "Maximum likelihood linear transformations for HMM-based speech recognition," *Computer Speech & Language*, vol. 12, no. 2, pp. 75–98, 1998.

[33] Y. Wu, M. Schuster *et al.*, "Google's neural machine translation system: Bridging the gap between human and machine translation," *arXiv preprint arXiv:1609.08144*, 2016.

[34] K. Rao, H. Sak *et al.*, "Exploring architectures, data and units for streaming end-to-end speech recognition with RNN-transducer," in *Proc. ICASSP*, 2017, pp. 193–199.

[35] Y. Wu, S. Zhang *et al.*, "On multiplicative integration with recurrent neural networks," in *Proc. NIPS*, 2016, pp. 2864–2872.

[36] G. Saon, Z. Tüske *et al.*, "Advancing RNN transducer technology for speech recognition," *arXiv preprint arXiv:2103.09935*, 2021.

[37] T. Bagby, K. Rao *et al.*, "Efficient implementation of recurrent neural network transducer in Tensorflow," in *Proc. SLT*, 2018, pp. 506–512.

[38] G. Saon, G. Kurata *et al.*, "English conversational telephone speech recognition by humans and machines," in *Proc. INTERSPEECH*, 2017, pp. 132–136.

[39] S. F. Chen, B. Kingsbury *et al.*, "Advances in speech transcription at IBM under the DARPA EARS program," *IEEE Transactions on Audio, Speech and Language Processing*, vol. 14, pp. 1596–1608, 2006.

[40] S. Thomas, M. Suzuki *et al.*, "English broadcast news speech recognition by humans and machines," in *Proc. ICASSP*, 2019, pp. 6455–6459.

[41] H. Sak, A. Senior *et al.*, "Fast and accurate recurrent neural network acoustic models for speech recognition," *arXiv preprint arXiv:1507.06947*, 2015.

[42] D. S. Park, W. Chan *et al.*, "SpecAugment: A simple data augmentation method for automatic speech recognition," in *Proc. INTERSPEECH*, 2019, pp. 2613–2617.

[43] G. Saon, Z. Tüske *et al.*, "Sequence noise injected training for end-to-end speech recognition," in *Proc. ICASSP*, 2019, pp. 6261–6265.

[44] L. Wan, M. Zeiler *et al.*, "Regularization of neural networks using dropconnect," in *Proc. ICML*, 2013, pp. 1058–1066.

[45] T. Ko, V. Peddinti *et al.*, "Audio augmentation for speech recognition," in *Proc. INTERSPEECH*, 2015, pp. 3586–3589.

[46] G. Saon, Z. Tüske *et al.*, "Alignment-length synchronous decoding for RNN transducer," in *Proc. ICASSP*, 2020, pp. 7804–7808.