A New Training Pipeline for an Improved Neural Transducer

Albert Zeyer[1,2], André Merboldt[1], Ralf Schlüter[1,2], Hermann Ney[1,2]

[1]Human Language Technology and Pattern Recognition, Computer Science Department,
RWTH Aachen University, 52062 Aachen, Germany,
[2]AppTek GmbH, 52062 Aachen, Germany

{zeyer, schlueter, ney}@cs.rwth-aachen.de, andre.merboldt@rwth-aachen.de

Abstract

The *RNN transducer* is a promising end-to-end model candidate. We compare the original training criterion with the full marginalization over all alignments, to the commonly used maximum approximation, which simplifies, improves and speeds up our training. We also generalize from the original neural network model and study more powerful models, made possible due to the maximum approximation. We further generalize the output label topology to cover RNN-T, RNA and CTC. We perform several studies among all these aspects, including a study on the effect of external alignments. We find that the transducer model generalizes much better on longer sequences than the attention model. Our final transducer model outperforms our attention model on Switchboard 300h by over 6% relative WER.

Index Terms: RNN-T, RNA, CTC, max. approx.

1. Introduction & Related work

End-to-end models in speech recognition are models with a very simple decoding procedure, and often a simple training pipeline. Usually the model directly outputs characters, subwords or words. One of the earlier end-to-end approaches was *connectionist temporal classification (CTC)* [1]. Most prominent is the *encoder-decoder-attention model* which has shown very competitive performance [2–5]. Once the streaming aspect becomes more relevant, or having a monotonicity constraint on the (implicit or explicit) alignment, the global attention model needs to be modified. Several ad-hoc solutions exists with certain shortcomings [6, 7]. The *recurrent neural network transducer (RNN-T) model* [8, 9] (or just *transducer model*) is an alternative model where the outputs can be produced in a time-synchronous way, and thus it is implicitly monotonic. Because of this property, RNN-T has recently gained interest [10–22]. RNN-T can be seen as a strictly more powerful generalization of CTC.

Several variations of RNN-T exists, such as the recurrent neural aligner (RNA) [23], monotonic RNN-T [24] or hybrid autoregressive transducer (HAT) [25]. The explicit time-synchronous modeling also makes the alignment of labels explicit, and requires a blank or silence label. The alignment becomes a latent variable. Most existing work keeps the model simple enough such that the marginalization over all possible alignments can be calculated efficiently via the forward-backward algorithm [8]. In case of RNA, an approximation is introduced.

Initializing the encoder parameters from another model (such as a CTC model) has often been done [9, 11, 19, 26]. Initializing some of the decoder parameters from a language model is common as well [11, 19, 27]. Using an external alignment has been studied in [19].

Differences in time-synchronous models (such as hybrid hidden Markov model (HMM) - neural network (NN) [28, 29]) vs. label synchronous models (such as encoder-decoder-attention and segmental RNN) w.r.t. the alignment behavior are studied in [30]. Time-synchronous decoding is also possible on joint CTC-attention models [31].

2. Model

Let $x_1^{T'}$ be the input sequence, which is encoded by a bidirectional LSTM [32] with time downsampling via max-pooling [2] and optional local windowed self-attention [33]

$$h_1^T = \text{Encoder}(x_1^{T'}).$$

Let y_1^N be the target sequence, where $y_n \in \Sigma$, for some discrete target vocabulary Σ, which are byte-pair encoded (BPE) labels [2, 34] in our work. We define a discriminative model

$$p(y_1^N \mid x_1^{T'}) = \sum_{\alpha_1^U : (T, y_1^N)} \prod_{u=1}^{U} p(\alpha_u \mid \alpha_1^{u-1}, h_1^T),$$

where $\alpha_u \in \Sigma' := \{\langle \text{b} \rangle\} \cup \Sigma$, where $\langle \text{b} \rangle$ is the *blank label*. The *output label topology* $\mathcal{T}$ over Σ' defines the mapping on t, and generates the sequence y_1^N. More specifically, the topology $\mathcal{T}$ defines $\Delta t_{\mathcal{T}}(\alpha) \geq 0$ such that $t_{u+1} = t_u + \Delta t_{\mathcal{T}}(\alpha_u)$, and $t_1 = 1$, $t_U = T$, and $\Delta n_{\mathcal{T}}(\alpha) \geq 0$ such that $n_{u+1} = n_u + \Delta n_{\mathcal{T}}(\alpha_u)$, and $n_1 = 1$, $n_U = N$. We study multiple variants of the label topology for α. Emitting a $\langle \text{b} \rangle$ label will always consume a time frame, and $\langle \text{b} \rangle$ will be removed from the final output. We study three variants:

- *CTC topology* [1]: Label emits time frame, repeated label will be collapsed. In this case, $U = T$, and $\Delta t \equiv 1$, $t_u = u$, and $\Delta n(\alpha_u) = \mathbf{1}_{\alpha_u \neq \langle \text{b} \rangle \wedge \alpha_u \neq \alpha_{u-1}}$.
- *RNA topology* [23] or *monotonic RNN-T* [24]: Label emits time frame, repeats will not be collapsed. $U = T$, and $\Delta t \equiv 1$, $t_u = u$, and $\Delta n(\alpha) = \mathbf{1}_{\alpha \neq \langle \text{b} \rangle}$.
- *RNN-T topology* [8]: Label does not emit time frame, repeats will not be collapsed. $U = N + T$, and $\Delta t(\alpha) = \mathbf{1}_{\alpha = \langle \text{b} \rangle}$, and $\Delta n(\alpha) = \mathbf{1}_{\alpha \neq \langle \text{b} \rangle}$.

We generalize from the common RNN-T and RNA model and describe our decoder network in terms of a fast and slow RNN [35–39]. The fast RNN iterates over $u \in \{1, \ldots, U\}$, while the slow RNN is calculated in certain sub frames $n_u \in \{1, \ldots, N\}$, whenever there was a new generated y. We visualize the unrolled decoder in Figure 1. The decoder for given frame u is defined by

$$s_u^{\text{fast}} := \text{FastRNN}\left(s_{u-1}^{\text{fast}}, s_{n_u}^{\text{slow}}, \alpha_{u-1}, h_{t_u}\right),$$

$$s_{n_u}^{\text{slow}} := \text{SlowRNN}\left(s_{n_u-1}^{\text{slow}}, \alpha_{u'-1}, h_{t_{u'}}\right),$$

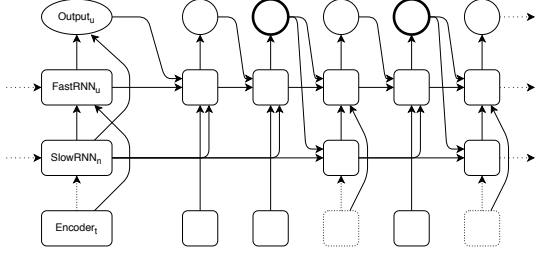$$u' := \min\{k \mid k \leq u, n_k = n_u\}. \qquad \text{(last emit)}$$

Figure 1: *Unrolled decoder. The output labels are in* $\{\langle b\rangle\}\cup\Sigma$. *A bold output depicts the emission of a new label in* Σ *(depending on the output label topology). SlowRNN is only updated when there is a new label. In case of RNN-T label topology, the encoder is only updated when there is no new label. In case of the original RNN-T model, SlowRNN has no dependency on the encoder.*

Both the $\mathrm{SlowRNN}$ and the $\mathrm{FastRNN}$ are LSTMs in our baseline. Note that we have $y_{n_u-1} = \alpha_{u'-1}$, and if we remove the dependency on h in SlowRNN, and if we remove FastRNN and just set $s_u^{\text{fast}} = h_{t_u}$, we get the original RNN-T model. To get the probability distribution for α over Σ', we could use a single softmax, as it was done for the original RNN-T and also RNA. Our baseline splits Σ and $\langle b\rangle$ explicitly into two separate probability distributions:

$$p(\alpha_u = \langle b\rangle \mid ...) := \sigma(\mathrm{Readout}^b(s_u^{\text{fast}}, s_{n_u}^{\text{slow}})),$$

$$p(\alpha_u \neq \langle b\rangle \mid ...) = \sigma(-\mathrm{Readout}^b(s_u^{\text{fast}}, s_{n_u}^{\text{slow}})),$$

$$q(\alpha_u \mid ...) := \mathrm{softmax}_\Sigma(\mathrm{Readout}^y(s_u^{\text{fast}}, s_{n_u}^{\text{slow}})), \; \alpha_u \in \Sigma$$

$$p(\alpha_u \mid ...) := p(\alpha_u \neq \langle b\rangle \mid ...) \cdot q(\alpha_u \mid ...), \; \alpha_u \in \Sigma$$

where $\mathrm{Readout}$ is some feed-forward NN. This can also be interpreted as a hierarchical shallow softmax. HAT [25] uses a similar definition. Note that the expressive power is equivalent to the single distribution:

$$q(\alpha_u \mid ...) = \frac{p(\alpha_u \mid ...)}{\sum_{\alpha'_u \in \Sigma} p(\alpha'_u \mid ...)}, \; \alpha_u \in \Sigma$$

$$p(\alpha_u \neq \langle b\rangle \mid ...) = 1 - p(\alpha_u = \langle b\rangle \mid ...).$$

2.1. Training

The model will be trained by minimizing the loss

$$L := -\log p(y_1^N \mid x_1^T) = -\log \sum_{\alpha_1^U : (T, y_1^N)} p(\alpha_1^U \mid x_1^T).$$

The sum $\sum_{\alpha_1^U}$ is usually solved via dynamic programming [40] by iterating over $u \in \{1, \ldots, U\}$. When we have a dependence on individual α_u in the decoder, it is not possible to calculate the sum $\sum_{\alpha_1^U}$ efficiently. It is possible though to do an approximation in the recombination [23]. The simplest approximation is to only use a single item of the sum, specifically $\arg\max \alpha_1^U$, which is the well known maximum approximation, which is done via the Viterbi alignment, or some other external alignment, as it is the standard approach for hybrid HMM-NN models [28, 29]. We study two variants here:

- Exact calculation of the full sum (when possible).
- Maximum approximation, together with fixed external alignment α_1^U. This is equivalent to frame-wise cross entropy training.

Using a fixed external alignment for the max. approximation has the disadvantage that we depend on a good external alignment, which complicates the training pipeline. But it comes with a number of advantages:

- We can train *strictly more expressive models*, which can potentially be more powerful, where the full sum cannot be calculated efficiently anymore.
- The training itself is *simpler* and reduces to simple frame-wise cross entropy training. This requires *less computation* and should be faster.
- It is *more flexible*, and we can use methods like chunking [29], focal loss [41], label smoothing [42].
- In addition, maybe it is *more stable*? Or we get *faster convergence* rate?

We also study the relevance of the type of external alignment. This is a forced alignment on some other unrelated model with the same output label topology. This other model can be a weaker model, only trained with the goal to generate the alignments. We study several variants of models for this task.

2.2. Decoding

We use beam search decoding with a fixed small beam size (12 hypotheses). The hypotheses in the beam are partially finished sequences α_1^u of the same length u. The pruning is based on the scores $p(\alpha_1^u | x_1^T)$. I.e. the decoding is time-synchronous (in case $U = T$), or synchronous over the axis $\{1, \ldots, U\}$ [22]. This is the same beam search algorithm and implementation as for our attention-based encoder-decoder model [2]. The only difference is that it runs over the axis $\{1, \ldots, U\}$.

As an optimization of the beam search space, we combine multiple hypotheses in the beam when they correspond to the same partial word sequence (after BPE merging), and we take the sum of their scores (which is another approximation, based on the model).

3. Experiments

We use RETURNN [43] as the training framework, which builds upon TensorFlow [44]. For the full-sum experiments on the RNN-T label topology, we use warp-transducer[1]. Our current full-sum implementation on the other label topologies is a pure TensorFlow implementation of the dynamic-programming forward computation. Via auto-diff, this results in the usual forward-backward algorithm. This is reasonably fast, but still slower than a handcrafted pure CUDA implementation, and also slower than the simple CE training. Both training and decoding is done on GPU. We present the training speeds in Table 1. We publish all our code, configs and full training pipeline to reproduce our results[2].

All the individual studies are performed on the Switchboard 300h English telephone speech corpus [45]. We use SpecAugment [3] as a simple on-the-fly data augmentation. We later compare to our attention-based encoder-decoder model [4].

3.1. Full-sum vs. frame-wise cross entropy

We compare full-sum (FS) vs. frame-wise cross entropy (CE) in Table 2. We observe that the full-sum training is more unstable, esp. in the beginning of the training, and leads to worse performance within the same amount of training time. We do not count the time to get the external alignment, so the comparison might not be completely fair. Chunking also has a positive effect on the CE training, as we will show later in Table 4.

We study the influence of the external alignment for the frame-wise CE training in Table 3. We see that a standard CTC model can be used to generate an alignment, but we also see that other models produce better alignments for our purpose. Specifically, using a transducer model (trained from scratch with full-

---

[1]https://github.com/HawkAaron/warp-transducer
[2]https://github.com/rwth-i6/returnn-experiments
/tree/master/2020-rnn-transducer

Table 1: *On Switchboard 300h. For each model, label topology, loss (full-sum (FS) or frame-wise cross entropy (CE)), and loss implementation (pure TensorFlow (TF), or CUDA), we compare the **training time** on a single GTX 1080 Ti GPU. This measures the whole training, not just the loss calculation. CE training is without chunking.*

Model	Label Topology	Loss	Loss Impl.	# params [M]	time / epoch [min]
Transd.	RNA	FS	TF	147	306
	CTC				326
	RNN-T				333
			CUDA		219
	CTC	CE	TF		160
Attention	–	CE	TF	162	138

Table 2: *On Switchboard 300h, transducer model, without external LM. Comparison of **full-sum (FS)** training and **frame-wise cross entropy (CE)** training via a fixed external alignment. All models are trained for 25 epochs and share the same network topology, which has no label feedback to allow FS training. CE training uses CTC alignments, where label repetition is enabled for CTC-Vit and disabled for RNA-Vit. CE training uses chunking.*

Label Topology	Training Criterion	WER[%]			
		Hub5'00			Hub5'01
		SWB	CH	Σ	Σ
RNA	FS	11.5	23.4	17.5	16.5
	CE	10.1	20.4	15.2	14.8
CTC	FS	15.0	24.6	19.8	20.1
	CE	10.5	20.6	15.6	15.3
RNN-T	FS	11.6	22.3	17.0	16.4

sum) to generate the alignment seems to work best. This is as expected, as this is the most consistent setup.

3.2. Ablations and variations

Along our research on training transducer models, we came up with many variants, until we eventually ended up with the baselines B1 and B2. Both transducer baselines use the CTC label topology with a separate sigmoid for the blank label (similar as in [25]). We use CE training using the fixed alignment CTC-align 6l (as in Table 3). Sequences of the training data are cut into chunks [29]. We use focal loss [41], an additional auxiliary CTC loss on the encoder (for regularization [2, 46]), dropout [47], dropconnect [48] (weight dropout) for the FastRNN, and switchout [49] (randomly switch labels for label feedback). B1 uses local windowed self-attention, while B2 does not, which is the only difference between B1 and B2. Some of these tricks were copied from our hybrid HMM-NN model [29]. Based on these baselines, we want to see the effect of individual aspects of the model or training. We summarized the variations and ablations in Table 4. We see that chunked training greatly helped for CE training, which is consistent with the literature [50]. We find that the results about label feedback are not conclusive. Having the separate sigmoid for blank seems to help.

3.3. Output label topology

We compare the output label topology in Table 2. We find that the RNN-T topology seems to perform best, followed by RNA, and CTC is worse. We note that this result is inconsistent to earlier results, where CTC looked better than RNA. However, when we repeat the RNA vs. CTC comparison on the B2 model with CE training, we also see that RNA performs better (14.2% vs 14.5% WER on Hub5'00). For simplicity, we did not follow the RNN-T topology further in this work. Also, because of our

Table 3: *On Switchboard 300h, WER on Hub 5'00. CE-trained transducer B1 and B2 models (Section 3.2), always with CTC label topology, without external LM, with randomly initialized parameters, trained for 25 epochs. Comparing **alignments** (specifically the models used to get the alignments). The alignment model was also always trained for 25 epochs.*

Alignment model	WER[%]	
	B1	B2
CTC-align 4l	14.7	14.3
CTC-align 6l	14.7	14.5
CTC-align 6l with prior (non-peaky)	15.4	14.9
CTC-align 6l, less training	14.6	14.6
Att.-based enc.-dec. + CTC-align	14.4	14.2
Transducer-align	14.2	14.1

Table 4: *On Switchboard 300h, WER on Hub5'00. **Ablations and variations**. Using transducer baselines B1 and B2 (see Section 3.2 for details), without external LM. B2 is exactly the B1 baseline without local windowed self-attention.*

Variant	WER[%]	
	B1	B2
Baseline	14.7	14.5
No chunked training	16.3	15.7
No switchout	15.0	14.5
SlowRNN always updated (not slow)	14.8	14.8
No SlowRNN	14.8	14.7
No attention	14.5	*
FastRNN dim 128 → 512	14.3	14.5
No encoder feedback to SlowRNN	14.9	14.7
+ No FastRNN label feedback (like RNN-T)	14.9	14.5
+ No FastRNN (exactly RNN-T)	15.2	15.1
No separate blank sigmoid	14.9	14.9

earlier results, we focused more on the CTC topology.

3.4. Importing existing parameters

For faster and easier convergence, it can be helpful to import existing model parameters into our RNA model. If we do not use the full-sum training, we anyway make use of an external alignment, which comes from some other model, so it might make sense to reuse these parameters. We collect our results in Table 5. We see that the CTC model parameters (of the same model which was used to create the alignment) seem to be suboptimal, and training from scratch performs better. The encoder of an attention-based encoder-decoder model seems to be very helpful. Importing the model itself, i.e. effectively training twice as long, helps just as much. However, we can also use the attention-based encoder-decoder model with an additional CTC layer on-top of the encoder to generate the alignments as shown in Table 3. We also tried to initialize the SlowRNN with the parameters of a LM but this had no effect.

3.5. Beam search decoding

We study different beam sizes, and compare the attention model and our RNA model. For RNA, we also implemented a variation of the beam search where hypotheses corresponding to the same word sequence (i.e. after collapsing label repetitions, removing blank, and BPE merging) were recombined together by taking their sum (in log space) and only the best hypothesis survives. The results are in Table 6. In all cases, the WER seems to saturate for beam size ≥ 8.

3.6. Generalization on longer sequences

It is known that global attention models do not generalize well to longer sequences than seen during training [51–53]. Esp. the

Table 5: *On Switchboard 300h, WER on Hub5'00. Varying the **imported model params**, for transducer baseline models B1 and B2, with CTC topology, trained with CE using a fixed external alignment (CTC-align 6l), without external LM. Trained for 25 epochs, with randomly initialized parameters, except of the imported ones. The imported models themselves are also trained for 25 epochs.*

Imported model params	WER[%]	
	B1	B2
None	14.7	14.5
CTC as encoder	15.4	15.5
Att. encoder	14.2	13.9
Transducer (itself)	13.7	13.6

Table 6: *On Switchboard 300h, WER on RT03S. The transducer uses the CTC-label topology. Without external LM. Comparison of performance on **different beam sizes**. We optionally recombine hypotheses in the beam corresponding to the same word sequence (after collapsing repetitions, removing blank, and BPE merging).*

Model	Merge	WER[%]							
		Beam size							
		1	2	4	8	12	24	32	64
Att.	no	17.9	17.0	16.7	16.6	16.6	16.5	16.6	16.5
Transd.	no	16.8	16.4	16.2	16.2	16.2	16.2	16.2	16.2
	yes	16.8	16.3	16.0	15.9	15.9	15.9	15.9	16.0

attention process has problems with this. The alignment process in the transducer is explicit, and this aspect should have no problems in generalizing to any sequence length. To analyze, during recognition, we concatenate every C consecutive seqs. within a recording and thus increase the avg. seq. lengths. We show the results in Table 7. We report the WER on RT03S to minimize overfitting effects. Both models degrade with longer sequences, but the attention model performs much worse, and the transducer model has generalized much better. The small degradation of the transducer could also be explained by unusual sentence boundaries. We note that this small degradation looks better than reported previously (relatively) [52–54].

3.7. Overall performance

Our final transducer model is based on B1 but with RNA label topology and better transducer-based alignment. We compare to our attention model and to other results from the literature in Table 8. Our final transducer model performs better than our attention model, although it needs the preprocessing step to get an alignment. We observe that many other works train for much longer, and there seems to be a correlation between training time and WER.

4. Conclusions

We found that the frame-wise CE training for transducer models greatly simplifies, speeds up and improves our transducer training by methods like chunking. It also allows us to train a novel transducer model. We gain interesting insights regarding model behaviour in decoding. Finally we achieve good results compared to the literature within much less training time. Our final transducer model is better than our attention model, and also generalizes much better on longer sequences.

5. Acknowledgements

Table 7: *On Switchboard 300h, WER on RT03S. The transducer uses the CTC output topology. Without external LM, beam size 12. Comparison of performance on **varying sequence lengths**, by concatenating every C consecutive seqs. (only in recog.).*

C	Seq. length [secs]		WER[%]	
	mean±std	min-max	Att.	Transd.
1	2.71± 2.38	0.17- 34.59	16.5	16.0
2	7.83± 4.96	0.42- 63.70	16.8	16.1
4	17.74± 9.25	0.42- 91.62	17.9	16.3
10	45.57±19.93	0.74-126.52	29.3	16.7
20	86.37±39.58	0.74-194.10	51.9	17.1
30	122.10±57.28	1.17-297.15	65.1	18.1
100	290.58±35.50	8.54-309.14	94.8	18.2

Table 8: *On Switchboard 300h, comparing final results of our **transducer model** with RNA label topology to our **attention model**, and to other attention models from the **literature**. One big difference in varying results is the different amount of training time, which we state as number of epochs.*

Work	Label Type	#	#Ep	LM	WER[%]				
					Hub5^{00}			Hub5^{01}	RT^{03}
					SWB	CH	Σ	Σ	Σ
[55]	Phone	4.5k	13	yes	9.6	18.5	14.0	14.1	
[4]	BPE	1k	33	no	10.1	20.6	15.4	14.7	
[56]	BPE	4k	50	no	8.8	17.2	13.0		
[57]		2k	100	yes	9.0	18.1	13.6		
[58]	BPE^{Ph}	500	150	yes	7.9	16.1			14.5
[5]	BPE	600	250	no	7.6	14.6			
[3]	WPM	1k	760	no	7.2	14.6			
Ours									
Att.	BPE	1k	25	no	9.2	21.1	15.2	14.2	17.6
			50		8.7	19.3	14.0	13.3	16.6
Transd.			25		9.4	18.7	14.1	14.1	16.7
			50		8.7	18.3	13.5	13.3	15.6

6. References

[1] A. Graves, S. Fernández, F. Gomez, and J. Schmidhuber, "Connectionist temporal classification: labelling unsegmented sequence data with recurrent neural networks," in *ICML*. ACM, 2006, pp. 369–376.

[2] A. Zeyer, K. Irie, R. Schlüter, and H. Ney, "Improved training of end-to-end attention models for speech recognition," in *Interspeech*, Hyderabad, India, Sep. 2018.

[3] 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," in *Proc. Interspeech 2019*, 2019, pp. 2613–2617.

[4] A. Zeyer, P. Bahar, K. Irie, R. Schlüter, and H. Ney, "A comparison of Transformer and LSTM encoder decoder models for ASR," in *ASRU*, Sentosa, Singapore, Dec. 2019, pp. 8–15.

[5] Z. Tüske, G. Saon, K. Audhkhasi, and B. Kingsbury, "Single headed attention based sequence-to-sequence model for state-of-the-art results on Switchboard-300," Preprint arXiv:2001.07263, 2020.

[6] C.-C. Chiu and C. Raffel, "Monotonic chunkwise attention," *arXiv preprint arXiv:1712.05382*, 2017.

[7] A. Merboldt, A. Zeyer, R. Schlüter, and H. Ney, "An analysis of local monotonic attention variants," in *Interspeech*, Graz, Austria, Sep. 2019, pp. 1398–1402.

[8] A. Graves, "Sequence transduction with recurrent neural networks," Preprint arXiv:1211.3711, 2012.

[9] A. Graves, A.-r. Mohamed, and G. Hinton, "Speech recognition with deep recurrent neural networks," in *ICASSP*, 2013.

[10] R. Prabhavalkar, K. Rao, T. N. Sainath, B. Li, L. Johnson, and N. Jaitly, "A comparison of sequence-to-sequence models for speech recognition," in *Interspeech*, 2017, pp. 939–943.

[11] K. Rao, H. Sak, and R. Prabhavalkar, "Exploring architectures, data and units for streaming end-to-end speech recognition with RNN-transducer," in *ASRU*, 2017.

[12] Y. He, T. N. 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. Y. Chang, K. Rao, and A. Gruenstein, "Streaming end-to-end speech recognition for mobile devices," in *ICASSP*, 2019, pp. 6381–6385.

[13] J. Li, R. Zhao, H. Hu, and Y. Gong, "Improving RNN transducer modeling for end-to-end speech recognition," Preprint arXiv:1909.12415, 2019.

[14] M. Jain, K. Schubert, J. Mahadeokar, C.-F. Yeh, K. Kalgaonkar, A. Sriram, C. Fuegen, and M. L. Seltzer, "RNN-T for latency controlled ASR with improved beam search," Preprint arXiv:1911.01629, 2019.

[15] 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," Preprint arXiv:1910.12977, 2019.

[16] A. Andrusenko, A. Laptev, and I. Medennikov, "Towards a competitive end-to-end speech recognition for CHiME-6 dinner party transcription," Preprint arXiv:2004.10799, 2020.

[17] B. Li, S.-y. Chang, T. N. Sainath, R. Pang, Y. He, T. Strohman, and Y. Wu, "Towards fast and accurate streaming end-to-end ASR," in *ICASSP*, 2020, pp. 6069–6073.

[18] 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*. IEEE, 2020, pp. 7829–7833.

[19] H. Hu, R. Zhao, J. Li, L. Lu, and Y. Gong, "Exploring pre-training with alignments for RNN transducer based end-to-end speech recognition," in *ICASSP*. IEEE, 2020, pp. 7079–7083.

[20] M. Ghodsi, X. Liu, J. Apfel, R. Cabrera, and E. Weinstein, "RNN-transducer with stateless prediction network," in *ICASSP*, 2020, pp. 7049–7053.

[21] W. Han, Z. Zhang, Y. Zhang, J. Yu, C.-C. Chiu, J. Qin, A. Gulati, R. Pang, and Y. Wu, "ContextNet: Improving convolutional neural networks for automatic speech recognition with global context," Preprint arXiv:2005.03191, 2020.

[22] G. Saon, Z. Tüske, and K. Audhkhasi, "Alignment-length synchronous decoding for RNN transducer," in *ICASSP*, 2020, pp. 7804–7808.

[23] H. Sak, M. Shannon, K. Rao, and F. Beaufays, "Recurrent neural aligner: An encoder-decoder neural network model for sequence to sequence mapping," in *Proc. of Interspeech*, 2017.

[24] A. Tripathi, H. Lu, H. Sak, and H. Soltau, "Monotonic recurrent neural network transducer and decoding strategies," in *ASRU*, 2019, pp. 944–948.

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

[26] S. Wang, P. Zhou, W. Chen, J. Jia, and L. Xie, "Exploring RNN-transducer for Chinese speech recognition," in *APSIPA*, 2019.

[27] L. Dong, S. Zhou, W. Chen, and B. Xu, "Extending recurrent neural aligner for streaming end-to-end speech recognition in Mandarin," in *INTERSPEECH*, 2018.

[28] H. Bourlard and N. Morgan, *Connectionist speech recognition: a hybrid approach*. Springer, 1994, vol. 247.

[29] A. Zeyer, P. Doetsch, P. Voigtlaender, R. Schlüter, and H. Ney, "A comprehensive study of deep bidirectional LSTM RNNs for acoustic modeling in speech recognition," in *ICASSP*, New Orleans, LA, USA, Mar. 2017, pp. 2462–2466.

[30] E. Beck, A. Zeyer, P. Doetsch, A. Merboldt, R. Schlüter, and H. Ney, "Sequence modeling and alignment for LVCSR-systems," in *ITG Conference on Speech Communication*, Oct. 2018.

[31] N. Moritz, T. Hori, and J. Le Roux, "Streaming end-to-end speech recognition with joint ctc-attention based models," in *ASRU*, Dec. 2019, pp. 936–943.

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

[33] Z. Lin, M. Feng, C. N. d. Santos, M. Yu, B. Xiang, B. Zhou, and Y. Bengio, "A structured self-attentive sentence embedding," *arXiv preprint arXiv:1703.03130*, 2017.

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

[35] J. Schmidhuber, "Learning complex, extended sequences using the principle of history compression," *Neural Computation*, vol. 4, no. 2, pp. 234–242, 1992.

[36] P. Liu, X. Qiu, X. Chen, S. Wu, and X. Huang, "Multi-timescale long short-term memory neural network for modelling sentences and documents," in *EMNLP*. Lisbon, Portugal: Association for Computational Linguistics, Sep. 2015, pp. 2326–2335. [Online]. Available: https://www.aclweb.org/anthology/D15-1280

[37] J. Chung, S. Ahn, and Y. Bengio, "Hierarchical multiscale recurrent neural networks," Preprint arXiv:1609.01704, 2016.

[38] A. Mujika, F. Meier, and A. Steger, "Fast-slow recurrent neural networks," in *NIPS*, 2017, pp. 5915–5924.

[39] I. Song, J. Chung, T. Kim, and Y. Bengio, "Dynamic frame skipping for fast speech recognition in recurrent neural network based acoustic models," in *ICASSP*. IEEE, 2018, pp. 4984–4988.

[40] R. E. Bellman, "Dynamic programming," Princeton University Press, Princeton, NJ, USA, 1957.

[41] T.-Y. Lin, P. Goyal, R. Girshick, K. He, and P. Dollár, "Focal loss for dense object detection," Preprint arXiv:1708.02002, 2017.

[42] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna, "Rethinking the inception architecture for computer vision," in *IC-CVPR*, 2016, pp. 2818–2826.

[43] A. Zeyer, T. Alkhouli, and H. Ney, "Returnn as a generic flexible neural toolkit with application to translation and speech recognition," in *Annual Meeting of the Assoc. for Computational Linguistics*, Melbourne, Australia, Jul. 2018.

[44] TensorFlow Development Team, "TensorFlow: Large-scale machine learning on heterogeneous systems," 2015, software available from tensorflow.org. [Online]. Available: https://www.tensorflow.org/

[45] J. J. Godfrey, E. C. Holliman, and J. McDaniel, "Switchboard: Telephone speech corpus for research and development," in *ICASSP*, 1992, pp. 517–520.

[46] T. Hori, S. Watanabe, Y. Zhang, and W. Chan, "Advances in joint CTC-attention based end-to-end speech recognition with a deep CNN encoder and RNN-LM," in *Interspeech*, 2017.

[47] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: A simple way to prevent neural networks from overfitting," *The Journal of Machine Learning Research*, vol. 15, no. 1, pp. 1929–1958, 2014.

[48] L. Wan, M. Zeiler, S. Zhang, Y. L. Cun, and R. Fergus, "Regularization of neural networks using dropconnect," in *Proceedings of the 30th International Conference on Machine Learning*, ser. Proceedings of Machine Learning Research, vol. 28, no. 3. Atlanta, Georgia, USA: PMLR, 17–19 Jun 2013, pp. 1058–1066. [Online]. Available: http://proceedings.mlr.press/v28/wan13.html

[49] X. Wang, H. Pham, Z. Dai, and G. Neubig, "SwitchOut: an efficient data augmentation algorithm for neural machine translation," in *EMNLP*. Brussels, Belgium: Association for Computational Linguistics, Oct. 2018, pp. 856–861. [Online]. Available: https://www.aclweb.org/anthology/D18-1100

[50] A. Zeyer, E. Beck, R. Schlüter, and H. Ney, "Ctc in the context of generalized full-sum hmm training," in *Interspeech*, Stockholm, Sweden, Aug. 2017, pp. 944–948.

[51] J. Rosendahl, V. A. K. Tran, W. Wang, and H. Ney, "Analysis of positional encodings for neural machine translation," in *IWSLT*, Hong Kong, China, Nov. 2019.

[52] A. Narayanan, R. Prabhavalkar, C.-C. Chiu, D. Rybach, T. N. Sainath, and T. Strohman, "Recognizing long-form speech using streaming end-to-end models," in *ASRU*, 2019.

[53] C.-C. Chiu, W. Han, Y. Zhang, R. Pang, S. Kishchenko, P. Nguyen, A. Narayanan, H. Liao, S. Zhang, A. Kannan, R. Prabhavalkar, Z. Chen, T. N. Sainath, and Y. Wu, "A comparison of end-to-end models for long-form speech recognition," in *ASRU*, 2019, pp. 889–896.

[54] C.-C. Chiu, A. Narayanan, W. Han, R. Prabhavalkar, Y. Zhang, N. Jaitly, R. Pang, T. N. Sainath, P. Nguyen, L. Cao, and Y. Wu, "RNN-T models fail to generalize to out-of-domain audio: Causes and solutions," Preprint arXiv:2005.03271, 2020.

[55] M. Zeineldeen, A. Zeyer, R. Schlüter, and H. Ney, "Layer-normalized LSTM for hybrid-HMM and end-to-end ASR," in *ICASSP*, Barcelona, Spain, May 2020.

[56] T.-S. Nguyen, S. Stueker, J. Niehues, and A. Waibel, "Improving sequence-to-sequence speech recognition training with on-the-fly data augmentation," Preprint arXiv:1910.13296, 2019.

[57] 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," in *ASRU*, 2019.

[58] W. Wang, Y. Zhou, C. Xiong, and R. Socher, "An investigation of phone-based subword units for end-to-end speech recognition," Preprint arXiv:2004.04290, 2020.