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

Librispeech Transducer Model with Internal Language Model Prior Correction

Albert Zeyer[1,2], *André Merboldt*[1], *Wilfried Michel*[1,2], *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, michel, schlueter, ney}@cs.rwth-aachen.de, andre.merboldt@rwth-aachen.de`

Abstract

We present our transducer model on Librispeech. We study variants to include an external language model (LM) with shallow fusion and subtract an estimated internal LM. This is justified by a Bayesian interpretation where the transducer model prior is given by the estimated internal LM. The subtraction of the internal LM gives us over 14% relative improvement over normal shallow fusion. Our transducer has a separate probability distribution for the non-blank labels which allows for easier combination with the external LM, and easier estimation of the internal LM. We additionally take care of including the end-of-sentence (EOS) probability of the external LM in the last blank probability which further improves the performance. All our code and setups are published.

Index Terms: transducer, language model integration, speech recognition

1. Introduction & Related Work

The recurrent neural network transducer (RNN-T) model [1, 2] is an end-to-end model which allows for time-synchronous decoding, which a more natural fit for many applications such as online recognition. Thus RNN-T and many variations has recently gained interest [3–8].

In a Bayesian interpretation, a discriminative acoustic model $p_{\mathrm{AM}}(y \mid x)$ can be combined with an external language model $p_{\mathrm{LM}}(y)$ by

$$p(y \mid x) = \frac{p_{\mathrm{AM}}(y \mid x)}{p_{\mathrm{AM}}(y)} \cdot p_{\mathrm{AM}}(x) \cdot p_{\mathrm{LM}}(y) \cdot \frac{1}{p(x)}.$$

In recognition, when searching for $\arg\max_y p(y \mid x)$, we can omit $p(x)$ and $p_{\mathrm{AM}}(x)$. In shallow fusion, $p_{\mathrm{AM}}(y)$ is omitted as well. In the density ratio approach [9], $p_{\mathrm{AM}}(y)$ is estimated by a separate language model trained on just the acoustic training transcriptions. In the hybrid autoregressive transducer (HAT) [6], $p_{\mathrm{AM}}(y)$ is estimated directly based on the implicit internal LM (ILM) of $p_{\mathrm{AM}}(y \mid x)$. The HAT model has a particular simple architecture which was designed such that there is a simple approximation for this ILM estimation by setting the encoder input to 0. We follow up on the ILM estimation approach and try some variations of the estimation. Using 0 as encoder input also works but we found some other variations to be better.

2. Model

We follow a transducer variant as defined in [7]. The whole model can be seen in Figure 1. Let $x_1^{T'}$ be the acoustic input features (MFCC in our case) of length T', and y_1^S some label sequence of length S over labels Σ (excluding blank ϵ). We use *byte pair encoding (BPE)*-based *subword units* [10, 11] with a vocabulary size of about 1000 labels[1].

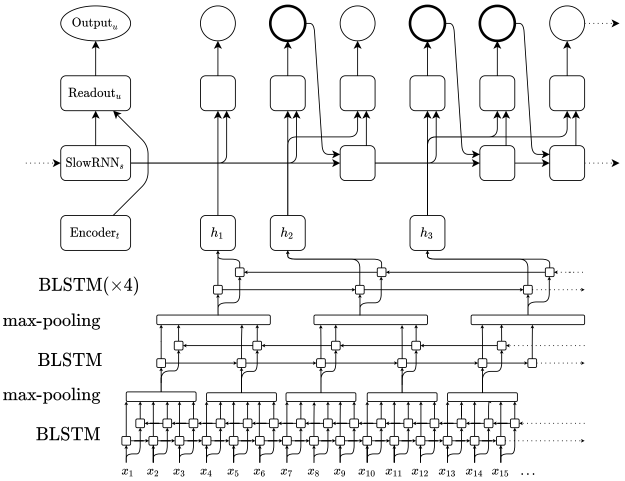
Output$_u$
Readout$_u$
SlowRNN$_s$
Encoder$_t$
h_1
h_2
h_3
BLSTM(×4)
max-pooling
BLSTM
max-pooling
BLSTM
x_1 x_2 x_3 x_4 x_5 x_6 x_7 x_8 x_9 x_{10} x_{11} x_{12} x_{13} x_{14} x_{15} …

Figure 1: *Our transducer model with all dependencies. The decoder is unrolled over the alignment axis* u. *Compare to [7].*

We have a multi-layer bidirectional LSTM [12] *encoder* model with interchanged max-pooling in time to downscale the input to length T with factor 6. This results in

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

We define the probability for the label sequence y_1^S as

$$p(y_1^S \mid x_1^{T'}) := \sum_{\alpha_1^U : y_1^S} p(\alpha_1^U \mid x_1^T),$$

$$p(\alpha_1^U \mid x_1^T) := \prod_{u=1}^{U} p_u(\alpha_u \mid \alpha_1^{u-1}, x_1^T),$$

with alignment label $\alpha_u \in \Sigma' := \Sigma \cup \{\epsilon\}$, and where $\alpha_1^U : y_1^S$ is defined by the label topology $\mathcal{A}$. Specifically, we use alignment length $U = T + S$ and allow all alignment label sequences α_1^U which match the sequence y_1^S after removing all blanks ϵ, also notated as $\mathcal{A}(\alpha_1^U) = y_1^S$. This defines an alignment between h and y as can be seen in Figure 2.

[1] In earlier work on attention-based encoder decoder models, we used 10k BPE labels for Librispeech. However, because of computation time and memory constraints, we reduced it to 1k for the transducer model.

http://dx.doi.org/10.21437/Interspeech.2021-1510

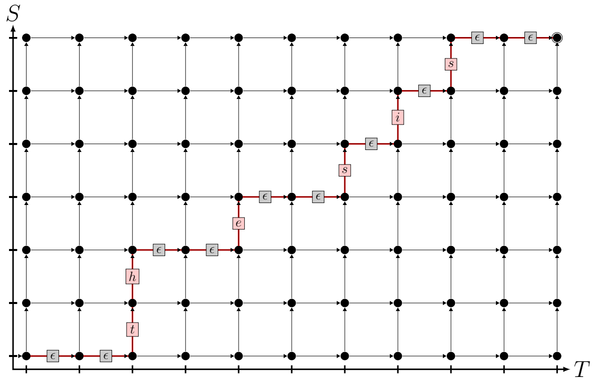

Figure 2: *Unrolled label topology with allowed vertical transitions, with a highlighted path for the sequence* $\mathcal{A}(\text{“}\epsilon\epsilon th\epsilon\epsilon e\epsilon\epsilon s\epsilon i\epsilon s\epsilon\epsilon\text{”}) = \text{“}thesis\text{”}$*. The terminal node is marked in the top-right corner.*

Our *decoder* model defines the probability distribution over labels α_u as

$$
\begin{aligned}
p_u(\alpha_u \mid ...) &:= \begin{cases} p_u(\Delta t_u{=}1 \mid ...), & \alpha_u = \epsilon, \\ p_u(\Delta t_u{=}0 \mid ...) \cdot q_u(\alpha_u \mid ...), & \alpha_u \in \Sigma \end{cases} \\
p_u(\Delta t_u \mid ...) &:= \begin{cases} \sigma(-\operatorname{FF}_{\text{emit}}(z_u^{\text{fast}})), & \Delta t_u = 1 \\ \sigma(\ \ \operatorname{FF}_{\text{emit}}(z_u^{\text{fast}})), & \Delta t_u = 0 \end{cases} \\
q_u(\alpha_u \mid ...) &:= \operatorname{softmax}_\Sigma(\operatorname{FF}_\Sigma(z_u^{\text{fast}})), \quad \alpha_u \in \Sigma \\
z_u^{\text{fast}} &:= \operatorname{Readout}(h_{t_u}, z_{s_u}^{\text{slow}}) \\
z_{s_u}^{\text{slow}} &:= \operatorname{SlowRNN}(y_1^{s_u - 1})
\end{aligned}
$$

where σ is the sigmoid function and $\Delta t \in \{0, 1\}$, where $\Delta t_u = 0$ means that we emit a new non-blank label ($\Delta s_u = 1$) and $\Delta t_u = 1$ means that we proceed forward in the time dimension without emitting a non-blank label ($\Delta s_u = 0$). Thus $\Delta t_u = 1$ can be understand as a reinterpretation of the blank label ϵ. Then we have a separate probability distribution q over the labels Σ (excluding ϵ). $\operatorname{FF}_{\text{emit}}$, FF_Σ are linear transformations, Readout is a linear transformation with maxout activation, and SlowRNN is an LSTM.

3. Training

The loss is defined as

$$
L := -\log p(y_1^S \mid x_1^{T'}) = -\log \sum_{\alpha_1^U : y_1^S} p(\alpha_1^U \mid x_1^T).
$$

As we use the simplified transducer model where Readout does not depend on α_{u-1}, we can efficiently calculate the exact full sum over all alignments $\alpha_1^U : y_1^S$ and do not need the maximum approximation [7].

We use zoneout [13] for the SlowRNN and optionally recurrent weight dropout [14] for the encoder BLSTMs.

We use the Adam optimizer [15] with learning rate scheduling based on cross validation scores. Additionally, we reset the learning rate back to the initial value after a larger number of epochs, after the model already converged, and start over with the learning rate scheduling. We train for 128 epochs. The long amount of training had a huge effect on the overall performance.

3.1. Pretraining

We use a pretraining scheme where we schedule multiple aspects of the training:

- We grow the encoder from 3 layers with 500 dim. up to 6 layers with 1000 dimensions [16].
- We increase the dropout rates.
- We use curriculum learning and start with shorter sequences initially.
- We use linear learning rate warmup from 0.0001 to 0.001.
- We use a higher initial time reduction factor 20 in the encoder and reduce it to the final factor 6.

3.2. Distributed Multi-GPU Training

Our distributed training implementation uses independent trainer (worker) instances per GPU. Each worker independently loads the dataset. To make sure that every worker uses a different part of the dataset, it is common to use striding. Striding has the disadvantage that it is very IO intensive in this setting where every worker loads the dataset independently and often becomes the bottleneck. So we came up with the idea to use a different random seed for the shuffling of the dataset for every worker, to replace the striding. This greatly improved the IO in our case and made the training much faster.

Additionally, every worker independently trains an own copy of the model for multiple update steps, until the models get synchronized by averaging the parameters over all workers. We made the further improvement that we do not synchronize after a fixed number of steps, but instead after a fixed time interval. A fixed number of steps implies that the training is always as slow as the slowest worker, and variations in the runtime often lead to some workers being slower than others even on same hardware. Synchronizing after a fixed time interval does not have this problem, while being more stochastic.

We synchronize only after 100 seconds to reduce the communication between workers. The workers can potentially be on different computing nodes and might need to communicate over network, which can result in 1-2 seconds for the synchronization. We train on either 8 or 16 GPUs.

4. Decoding & Language Model Combination

Our beam search decoding tries to find the sequence $\hat{y}_1^{\hat{S}}$ given $x_1^{T'}$ which maximizes the probability, i.e. specifically

$$
\begin{aligned}
x_1^{T'} \mapsto \hat{S}, \hat{y}_1^{\hat{S}} &:= \operatorname*{arg\,max}_{S, y_1^S} \log p(y_1^S \mid x_1^{T'}) \\
&\approx \mathcal{A} \circ \operatorname*{arg\,max}_{U, \alpha_1^U} \log p(\alpha_1^U \mid x_1^{T'})
\end{aligned}
$$

We perform alignment-synchronous decoding, i.e. all hypotheses are in the same alignment step u when being pruned [7, 17]. We merge hypotheses by summing their scores when they correspond to the same word sequence after BPE-merging.

The training recipe for our BPE-10K LSTM LM [18] has been adapted for the new BPE-1k label set but otherwise no changes have been made. *Shallow fusion* (SF) [19] is a log-linear combination of the log-scores of external LM and ASR model scores during the recognition process, with scale β for the LM and scale λ for the acoustic (non-blank) label probability q, while we do not add an own scale for $p(\Delta t)$. Specifically, we use the score

$$\log p_u^{\text{SF}}(\alpha_u \mid ...) := \begin{cases} \log p_u(\Delta t_u{=}1 \mid ...), & \alpha_u = \epsilon, \\ \log p_u(\Delta t_u{=}0 \mid ...) & \\ \quad + \lambda \cdot \log q_u(\alpha_u \mid ...) & \\ \quad + \beta \cdot \log p_{\text{LM}}(\alpha_u \mid ...), & \alpha_u \in \Sigma \end{cases}.$$

We experimented with fixing the label scale at $\lambda = 1$ or $\lambda = 1 - \beta$.

Inspired by [6, 9, 20] we also tried to *subtract the internal LM* log score. It assumes that we can factorize our model into a language and acoustic model. Although our model is not directly formulated as such, we can approximate the internal language model. For that we used the estimated score $\log p_{\text{ILM}}$ as shown in Section 4.1, where we use the average of encoder features in the time dimension.

$$\log p_u^{\text{SF-ILM}}(\alpha_u \mid ...) := \begin{cases} \log p_u(\Delta t_u{=}1 \mid ...), & \alpha_u = \epsilon, \\ \log p_u(\Delta t_u{=}0 \mid ...) & \\ \quad + \lambda \cdot \log q_u(\alpha_u \mid ...) & \\ \quad + \beta \cdot \log p_{\text{LM}}(\alpha_u \mid ...) & \\ \quad - \gamma \cdot \log p_{\text{ILM}}(\alpha_u \mid ...), & \alpha_u \in \Sigma \end{cases}$$

4.1. Internal LM Estimation

The transducer is trained on audio-text pairs but learns an implicit prior model on the text. This is explicitly given by the context dependency on previous labels. In this transducer case, the SlowRNN is also explicitly modeled such that it models the most important part of this prior as it operates only on the text-only part and runs label-synchronous. This prior is an implicit internal LM in our acoustic model

$$p_{\text{prior}}(y) = \sum_x p_{\text{AM}}(y \mid x) \cdot p(x)$$

which can not be calculated efficiently in general. To approximate the internal LM, we replace the encoder input to the rest of the model (Readout). We either use a 0 vector or the encoder mean (avg). The mean is computed over the time dimension for each sequence separately.[2]

We evaluate the estimated internal LM on text-only data. In Table 1, the BPE-level perplexities (PPL) are shown and compared against the LSTM LM which was trained only on text data, but without any overlap to the audio transcriptions [22].

4.2. EOS Modelling

In contrast to language models or attention models, transducers and models with explicit time modeling do not have to model the end-of-sentence/sequence explicitly with an additional token (denoted as $\langle\text{eos}\rangle$). Instead the search ends when all input frames have been consumed. However, for LM integration, when only considering actual output symbols, the information about when the sequence should end is not considered. This additional information is usually ignored in the literature, however it provides valuable information to the search process.

Table 1: *Perplexity and WER measurements on Librispeech dev-other of a transducer model. Note that the BPE-level (1k units) perplexity is evaluated without the EOS token, since the transducer has no explicit end-of-sequence symbol. Compared are both setting the encoder (h) to 0 and to the mean over the time-dimension (*avg*). The LSTM and Trafo-LM are trained on text-only data without overlap to the audio transcriptions [22].*

Model	Epochs	Perplexity		WER
		$h = 0$	avg	[%]
Transducer BPE-1K	8	82.76	67.47	36.41
	16	49.32	38.89	17.16
	32	45.13	32.86	11.85
	64	46.53	31.94	9.69
	133	47.05	31.37	8.92
LSTM LM	20	15.40		–
Trafo LM	39	14.44		–

Our approach is to combine the LM EOS probability with $p_u(\Delta t{=}1)$ (ϵ) in the last time frame ($t_u = T$) because that determines the EOS in the transducer.

$$\log p_u^{\text{SF-ILM+EOS}}(\alpha_u \mid ...)$$
$$:= \begin{cases} \log p_u(\Delta t{=}1 \mid ...), & \alpha_u = \epsilon, t_{u-1} < T \\ \delta_{\langle\text{eos}\rangle} \log p_u(\Delta t{=}1 \mid ...) & \\ \quad + \beta_{\langle\text{eos}\rangle} \log p_{\text{LM}}(\langle\text{eos}\rangle \mid ...), & \alpha_u = \epsilon, t_{u-1} = T \\ \log p_u(\Delta t{=}0 \mid ...) & \\ \quad + \lambda \cdot \log q_u(\alpha_u \mid ...) & \\ \quad + \beta \cdot \log p_{\text{LM}}(\alpha_u \mid ...) & \\ \quad - \gamma \cdot \log p_{\text{ILM}}(\alpha_u \mid ...), & \alpha_u \in \Sigma \end{cases}$$

Usually $\delta_{\langle\text{eos}\rangle} = \beta_{\langle\text{eos}\rangle} = 0.5$ yielded good performance, although it was not tuned properly.

5. Experiments

We perform experiments on LibriSpeech [22]. Our model training and decoding is implemented in RETURNN [23], based on TensorFlow [24]. The distributed multi-GPU training is implemented with Horovod [25]. We make use of Mingkun Huang's warp-transducer loss implementation[3]. Our decoder uses the builtin RETURNN features for stochastic variables and searches over α_u. This uses GPU-based batched one-pass decoding with the external LM and internal LM subtraction. We publish all the configuration files needed to reproduce the experiments[4].

We have a variety of different exponent scales for our log-linear modeling, as well as additional parameters for EOS-modeling. Label scale λ which is set to either $\lambda = 1$ or $\lambda = 1 - \beta$, the emission model scale δ, and the scales for external and internal LM β, γ, respectively. Additionally for EOS-modeling $\delta_{\langle\text{eos}\rangle}$ and $\beta_{\langle\text{eos}\rangle}$ is used, although it was fixed to $\delta_{\langle\text{eos}\rangle} = \beta_{\langle\text{eos}\rangle} = 0.5$. The scaling factors β and γ have to be tuned jointly on a held-out dataset, as can be seen in Fig. 3, with $\lambda = 1 - \beta$. They were tuned separately for each subset dev-clean and dev-other. Results for LM integration are presented in Table 2 and in Table 3 with additional EOS-modeling. With shallow fusion of just the LM we already see a significant WER improvement by over 22% relative. When additionally subtracting the internal LM, a further significant improvement is observed by over 14% relative over the shallow fusion.

[2] We also tested several other variants but got mixed inconclusive results. In another work [21], we investigate variants on the ILM estimation in more detail for attention-based encoder-decoder models.

[3] `https://github.com/HawkAaron/warp-transducer`

[4] `https://github.com/rwth-i6/returnn-experiments/tree/master/2021-transducer`

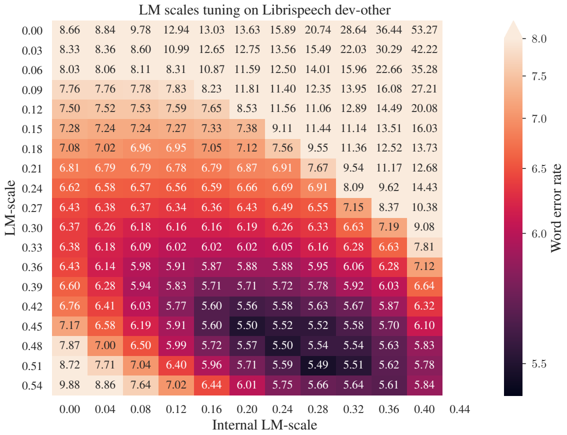

Figure 3: *Tuning of LM scales of a transducer with EOS-modeling. The baseline without an external LM has* 8.66% *WER on dev-other with a beam size of 24.* $\lambda = 1 - \beta$*.*

Table 2: *We investigate the effect of LM integration for the model with either shallow fusion (SF) or additional internal LM (ILM) subtraction. All experiments were conducted with a fixed* ***beam-size 24*** *and* ***without EOS-modeling****, the LSTM-LM has a BPE-1K level perplexity of* 15.4 *on dev-other.*

LM	LM Integration Method	Label scale λ	WER [%]			
			dev		test	
			clean	other	clean	other
—	—	$\lambda = 1$	3.22	8.76	3.30	8.70
LSTM	SF		2.53	6.79	2.66	6.99
		$\lambda = 1 - \beta$	2.47	6.50	2.57	6.70
	SF-ILM(avg)		**2.29**	**5.63**	**2.36**	**6.39**

The avgILM estimation seems to be better than 0except on test-other. The effect of EOS gives us 7% relative improvement. We also test a stronger Transformer LM in Table 3 (perplexities in Table 1) and see further improvement.

5.1. Error Analysis

One of the sources of errors when looking at an entire system are errors the model made when its prediction was wrong. We look at the percentages of substitution, deletion, and insertion errors of the word error rate (WER). Especially interesting is the comparison between different models and their respective LM integration. Also of interest are how long the hypothesized sentences are, relative to the reference transcription. The transducer seems to model the hypothesis length better than hybrid (without rescoring) and attention-based models, although adding an external LM seems to help the attention model. Overall we can see that introducing the external LM helps with substitutions and insertion errors, while the deletions actually increase. In comparison to the attention-based model, the transducer model has significantly less insertion errors, but more deletion errors, relative to the overall WER.

6. Conclusions & Future Work

The subtraction of the ILM helped to improve the model by a lot (over 14% relative) over the already strong shallow fusion. The EOS modeling also helped (7% relative). We noticed that all recognition experiments are very sensitive to the LM/ILM scales. The long training time also had a huge effect on the final performance.

As future work, we plan to study the effect of the label unit and to test simple characters and other subword variations, similar to [28]. The encoder model might get improvements by more recent advancements [5]. The decoder can be extended as well [7]. We also can potentially improve the ILM estimation. Finally, we expect to get improvements by min. WER training.

Table 3: *We investigate the effect of LM integration for the model with either shallow fusion (SF) or additional internal LM (ILM) subtraction. All experiments were conducted with a fixed* ***beam-size 24*** *and* ***EOS-modeling*** *(last blank frame), the LSTM-LM has a BPE-1K level perplexity of* 15.4 *on dev-other. In Fig. 3 the heat map for the joint tuning over* β_{OTHER} *and* γ_{OTHER}*.* $\delta_{\langle\text{eos}\rangle} = \beta_{\langle\text{eos}\rangle} = 0.5$.

LM	LM Integration Method	Label scale λ	WER [%]			
			dev		test	
			clean	other	clean	other
—	—	$\lambda = 1$	3.20	8.66	3.28	8.60
LSTM	SF	$\lambda = 1$	2.52	6.69	2.65	6.85
		$\lambda = 1 - \beta$	2.45	6.35	2.55	6.68
	SF-ILM(avg)		**2.26**	**5.49**	**2.42**	**5.91**
Trafo	SF	$\lambda = 1 - \beta$	2.41	6.29	2.52	6.56
	SF-ILM(avg)		**2.17**	**5.28**	**2.23**	5.74
	SF-ILM(0)		2.22	5.32	2.25	**5.60**

Table 4: *We investigate the different type of word errors on various models. Hybrid [26], Attention [27], and Transducer (ours). With either shallow fusion (SF) or additional internal LM (ILM) subtraction. For log-linear combination in the transducer case,* $\lambda = 1 - \beta$*.*

Model	LM	LM integration	Edit operations [%]			WER
			Sub.	Del.	Ins.	[%]
Attention	None	—	80.40	6.96	12.65	9.93
	LSTM	SF	79.42	5.68	14.90	7.50
Hybrid	4-gr.	SF	75.75	12.98	11.27	9.37
Transd.	None	—	82.52	7.55	9.93	8.76
	LSTM	SF	79.10	10.93	9.97	6.50
		SF-ILM(avg)	80.90	9.06	10.04	5.63
		SF-ILM(avg)	79.81	10.15	10.04	5.49
	Trafo	+EOS	80.45	9.55	10.00	5.28

7. Acknowledgements

We thank Yingbo Gao for providing us with the Transformer LM on our BPE-1k label set. This project has received funding from the European Research Council (ERC) under the European Union's Horizon 2020 research and innovation programme (grant agreement n$^{\text{o}}$ 694537, project "SEQCLAS"). The work reflects only the authors' views and the European Research Council Executive Agency (ERCEA) is not responsible for any use that may be made of the information it contains. This work was partly funded by the Google Focused Award "Pushing the Frontiers of ASR: Training Criteria and Semi-Supervised Learning".

8. References

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

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

[3] 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.

[4] 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.

[5] A. Gulati, J. Qin, C.-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," Preprint arXiv:2005.08100, 2020.

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

[7] A. Zeyer, A. Merboldt, R. Schlüter, and H. Ney, "A new training pipeline for an improved neural transducer," in *Interspeech*, Shanghai, China, Oct. 2020.

[8] W. Zhou, S. Berger, R. Schlüter, and H. Ney, "Phoneme based neural transducer for large vocabulary speech recognition," in *IEEE International Conference on Acoustics, Speech, and Signal Processing*, May 2021, submitted to.

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

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

[11] 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.

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

[13] D. Krueger, T. Maharaj, J. Kramár, M. Pezeshki, N. Ballas, N. R. Ke, A. Goyal, Y. Bengio, A. Courville, and C. Pal, "Zoneout: Regularizing RNNs by randomly preserving hidden activations," in *ICLR*, 2017.

[14] 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

[15] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," in *Proceedings of the International Conference on Learning Representations (ICLR)*, San Diego, CA, 2015.

[16] A. Zeyer, A. Merboldt, R. Schlüter, and H. Ney, "A comprehensive analysis on attention models," in *Interpretability and Robustness in Audio, Speech, and Language (IRASL) Workshop, NeurIPS*, Montreal, Canada, Dec. 2018.

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

[18] K. Irie, "Advancing neural language modeling in automatic speech recognition," Ph.D. dissertation, RWTH Aachen University, Computer Science Department, RWTH Aachen University, Aachen, Germany, May 2020. [Online]. Available: http://publications.rwth-aachen.de/record/789081

[19] Ç. Gülçehre, O. Firat, K. Xu, K. Cho, L. Barrault, H.-C. Lin, F. Bougares, H. Schwenk, and Y. Bengio, "On using monolingual corpora in neural machine translation," *Computer Speech & Language*, vol. 45, pp. 137–148, Sep. 2017.

[20] Z. Meng, S. Parthasarathy, E. Sun, Y. Gaur, N. Kanda, L. Lu, X. Chen, R. Zhao, J. Li, and Y. Gong, "Internal language model estimation for domain-adaptive end-to-end speech recognition," in *2021 IEEE Spoken Language Technology Workshop (SLT)*. IEEE, 2021, pp. 243–250.

[21] M. Zeineldeen, A. Glushko, W. Michel, A. Zeyer, R. Schlüter, and H. Ney, "Investigating methods to improve language model integration for attention-based encoder-decoder ASR models," submitted to Interspeech 2021, 2021.

[22] V. Panayotov, G. Chen, D. Povey, and S. Khudanpur, "LibriSpeech: an ASR corpus based on public domain audio books," in *ICASSP*. IEEE, 2015, pp. 5206–5210.

[23] 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.

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

[25] A. Sergeev and M. Del Balso, "Horovod: fast and easy distributed deep learning in TensorFlow," Preprint arXiv:1802.05799, 2018.

[26] C. Lüscher, E. Beck, K. Irie, M. Kitza, W. Michel, A. Zeyer, R. Schlüter, and H. Ney, "RWTH ASR systems for LibriSpeech: Hybrid vs attention," in *Interspeech*, Graz, Austria, Sep. 2019, pp. 231–235.

[27] 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.

[28] M. Zeineldeen, A. Zeyer, W. Zhou, T. Ng, R. Schlüter, and H. Ney, "A systematic comparison of grapheme-based vs. phoneme-based label units for encoder-decoder-attention models," Preprint arXiv:2005.09336, submitted to ICASSP 2021, Nov. 2020. [Online]. Available: http://arxiv.org/abs/2005.09336