INTERSPEECH 2019
September 15–19, 2019, Graz, Austria

Advancing sequence-to-sequence based speech recognition

Zoltán Tüske, Kartik Audhkhasi, George Saon

IBM Research AI, Yorktown Heights, USA

zoltan.tuske@ibm.com

Abstract

The paper presents our endeavor to improve state-of-the-art speech recognition results using attention based neural network approaches. Our test focus was LibriSpeech, a well-known, publicly available, large, speech corpus, but the methodologies are clearly applicable to other tasks. After systematic application of standard techniques – sophisticated data augmentation, various dropout schemes, scheduled sampling, warm-restart –, and optimizing search configurations, our model achieves 4.0% and 11.7% word error rate (WER) on the test-clean and test-other sets, *without* any external language model. A powerful recurrent language model drops the error rate further to 2.7% and 8.2%. Thus, we not only report the lowest sequence-to-sequence model based numbers on this task to date, but our single system even challenges the best result known in the literature, namely a hybrid model together with recurrent language model rescoring. A simple ROVER combination of several of our attention based systems achieved 2.5% and 7.3% WER on the clean and other test sets.

Index Terms: encoder-decoder, attention, speech recognition, LibriSpeech

1. Introduction

With the widespread use of deep neural networks (NN), end-to-end approaches have rapidly gained popularity in speech recognition community as a result of dramatic simplification of the training and decoding pipelines. Such models can operate on characters, sub-words, or even full-words instead of phonetic sub-states, hence making pronunciation lexicon unnecessary. The model can directly map audio feature sequences (x_1^T) to word sequences (y_1^L). Expressed in a statistical framework, the aim is to learn the conditional posterior distribution as precisely as possible.

$$p(y_1^L|x_1^T) = \prod_{l=1}^{L} p(y_l|y_1^{l-1}, x_1^T) \tag{1}$$

Depending on the end-to-end approach, the alignment problem, which arises from the different length of the input and target sequences (T vs. L), is handled differently. In the CTC [1], RNN-transducer [2], or standard hybrid based approaches [3, 4], alignment is handled explicitly in the training criterion by introducing hidden variables. In order to keep the input and output streams synchronous, skip and/or repetition is allowed, or a special blank symbol is introduced to equalize the sequence lengths. The other frequently used approach, on which this paper will focus, is attention based encoder-decoder modeling [5, 6, 7, 8]. This model handles the alignment problem internally by squashing the input stream dynamically by a trainable attention mechanism in synchrony with the output sequence. The model is capable to handle problems with non-monotonic alignment exceptionally well, and has became the state-of-the-art approach in e.g. machine translation [9].

Despite the simple left-to-right nature of speech, several attempts have been made to reach the state-of-the-art performance of a hybrid system with attention based speech recognition models [10, 11, 12]. As has been shown, a sufficiently large database might be a key component to achieve this goal [13]. Nevertheless, the usual comparisons consider hybrid results after decoding with count based language model, and often discard the fact that encoder-decoder models learn neural-network-like word-sequence representation internally, which is much more powerful than count based models [14, 15]. Thus, it has still remained an open question whether attention based models can match the performance of a standard hybrid model combined with RNN-based rescoring/decoding.

Looking for the answer of the above question, we carried out several experiments with our attention based speech models on a relatively large, publicly available dataset, LibriSpeech. We present the detailed description of our successful attempt to achieve a new state of the art on this task by encoder-decoder based sequence-to-sequence model.

2. Experimental setup

2.1. Data description

LibriSpeech is a publicly available, read speech corpus [16]. The overall audio data for training sum up to roughly 1000 hours. The development and test sets contain approximately 10 hours of speech. Each set is split in two, the easier part is referred to as "clean" whereas the more challenging half is called "other". A 200k lexicon with manually and automatically generated pronunciations is also provided. With this lexicon the out-of-vocabulary (OOV) rate is well below 1% on the development and test sets. A text corpus of 803M million running words is also available for language modeling (LM) purposes.

2.2. Attention model

Our attention based encoder-decoder model is summarized in Eq. 2 and Fig. 1.

$$\begin{aligned} p(y_l|y_1^{l-1}, x_1^T) &= \text{softmax}\left(W\left(c_l + d_l\right) + b_W\right) \\ c_l &= \sum_{n=1}^{N} \alpha_{l,n} h_n(x_1^T) \\ \alpha_{l,n} &= \text{softmax}_n(v^\intercal \text{ReLU}(d_l + h_n + \bar{\alpha}_{l-1,n}) + b_v) \\ \bar{\alpha}_{l,n} &= A * \alpha_{l,n} + b_a \end{aligned} \tag{2}$$

After encoding the input sequence x_1^T into h_1^N with a deep, pyramidal, bidirectional long-short term memory (LSTM) network [17, 18, 10], h_1^N is processed by the decoder network (Fig. 1). At the lth step, the posterior vector for the output symbols is generated from the transformed decoder state d_l and context-vector c_l. The context vector corresponds to the attention (α) based linear combination of the encoded input vector sequence h_1^N. Our decoder network uses the location aware attention mechanism of [19], and the previous attention is fed back after convolving it with a set of 1-dimensional kernel functions (A) along the time axis. The energy function of the attention calculation is performed by a single-layer rectified-linear-unit (ReLU) feed-forward neural network [20]. W, A (matri-

http://dx.doi.org/10.21437/Interspeech.2019-3018

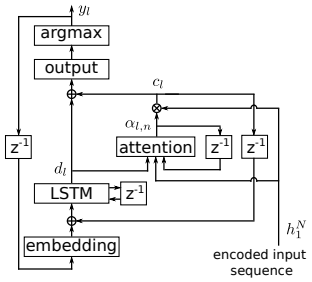

Figure 1: *The attention based decoder network used in the experiments.*

ces), v, b_W, b_a (vectors), and b_v (scalar) are trainable parameters. We use a single attention mechanism, and d_l is calculated by a unidirectional LSTM NN.

At recognition time, the following decision is made to find the optimal sequence using an external LM:

$$\hat{y}_1^L = \arg\max_{y_1^L, L} \left(\sum_{l=1}^{L} log\, p(y_l|y_1^{l-1}, x_1^T) + \lambda_{LM} \sum_{l=1}^{L} log\, p(y_l|y_1^{l-1}) + \lambda_{len} L + \lambda_{ct} \sum_{n=1}^{N} I_{\{\alpha>\tau\}}(\max_l \alpha_{l,n}) \right) \quad (3)$$

A simple model combination of the sequence-to-sequence and LM in the first and second term is often referred to as shallow-fusion [21]. The third term denotes the length normalization [22], which helps promoting longer hypotheses. The last component corresponds to a modified version of the coverage term of [23, 24], where I is the indicator function, and τ is a tunable parameter. Together with the max operation it forces the search to prefer paths with sharp and non-overlapping attention distributions. We found that a cumulative term might fire unnecessarily and boost weak hypothesis with unfocused attention. Eq. 3 was evaluated by beam search. The coverage term was updated at each step keeping track of the necessary statistics. Length normalization was applied when the hypothesis reached sentence end, before expanding the n-best list. The input and output of the encoder-decoder models are not handled synchronously, therefore special care should be taken to avoid running the model indefinitely. We used $L < L_{max} = \lceil \gamma T \rceil$ as a threshold, and enforced that all hypotheses reach the sentence-end state by the L_{max} step. In addition, the posterior $p(y_l|y_1^{l-1}, x_1^T)$ and the prior $p(y_l|y_1^{l-1})$ probability estimates were scaled by $\beta_{seq} < 1$ and $\beta_{LM} < 1$ before the softmax calculation (smoothing). By default, the search process was constrained by the 200k vocabulary (Sec. 2.1), using a search graph to control the hypothesis expansion. In the case of sub-word units, the segmented form of the vocabulary was transformed to a prefix-tree.

Table 1: *LSTM language models for LibriSpeech, perplexity (PPL) measured on the combined set of dev-clean and -other.*

unit		vocab.	#param.	PPL	PPL_{word}
char		29	54.8M	2.44	90.9
BPE	100	156	54.9M	3.56	89.9
			122.1M	3.47	82.6
	1k	1056	55.4M	6.08	92.1
	10k	10056	61.3M	8.91	85.1
	30k	29998	74.0M	9.43	78.2
word		89115	111.9M	71.4*	

*A2W, 0.7% of words were deleted

2.3. Training setup

Unless otherwise noted, our encoder-decoder model uses the following default settings resulted from a series of initial experiments. The input audio files are speed, tempo, and/or volume perturbed with 75% probability [25]. The encoder consists of 6 bidirectional LSTM layers with 512 nodes per layer and per direction. Its input is 50-dimensional speaker independent, utterance-level mean-normalized Gammatone (GT) features [26]. We also tested 40-dimensional MFCC features, which were speaker level mean-and-variance normalized. Following [27], with 40% probability a randomly chosen utterance was also injected as a sequence noise, and it was mixed up with 0.3 weight with the original feature stream at spectral level. In the first layers of the encoder, the frame rate was reduced by a factor of two per layer until the desired ratio was achieved [10, 28]. As a default setting $T = 4N$, and the LSTM output is sub-sampled by max-pooling. Furthermore, the forward and backward directions of the LSTM outputs are summed up after every layer. We also apply 20% dropout rate to the LSTM outputs, and 20% drop-connect to the hidden-to-hidden matrices [29, 30]. The final dimension of the encoder output is 256, enforced by a linear bottleneck. In the decoder, the dropout probability is 5% for the embedding and 10% for the decoder LSTM layer. The output and the embedding layers model characters, byte-pair encoding (BPE) based sub-word units, or full-words [31]. According to Eq. 2, d_l, h_n, c_l, and $\bar{\alpha}_{l,n}$ have the same dimension, thus the LSTM and embedding layers in the decoder also have 256 nodes. Similarly, the attention output is filtered by 256 kernels (A) with kernel size 5. The models were trained to minimize the cross-entropy loss. We used scheduled sampling, forcing the teacher labels in 80% of the time [32]. Recordings of 24 randomly chosen speakers (9.6 hours) were selected for cross-validation. The randomly initialized models were trained from scratch for 50 epochs with variable batch size of up to 256 utterances. Depending on maximum length, the number of sequences in a batch was limited by GPU memory. In the first 10 epochs, sequences were presented in increasing length order to the network. Furthermore, the cross-entropy objective was complemented by CTC loss for 10 epochs, then linearly decayed in 5 epochs [33]. After 30 epochs, the learning rate was annealed by a factor of $1/\sqrt{2}$ in 20 steps. The final models were obtained after an additional 50-epoch training using warm-restart (SGDR) [34], resetting merely the learning rate to its initial value of 0.01. We always used Nesterov momentum set to 0.9 [35].

The LSTM LM and the BPE rules were trained on the merged set of the transcription of the acoustic data and the stand-alone text database (822M running words). Segmenting the LM data to characters resulted in 4.3B running tokens. All sub-word unit (including character) based attention systems and LMs used a special token to denote word end. The NNLMs have an embedding size of 512, two LSTM layers with 2048 nodes, and a 128-dimensional bottleneck layer before the softmax output. Similar to [36], drop-connect with 10% ratio was applied to the embedding and also to the hidden-to-hidden transformation matrices of the LSTM layers, and 10% of the outputs of these layers were also dropped out. The initial learning rate and Nesterov momentum were the same as above. After 10 epochs of training, the learning rate was annealed by a factor of $1/\sqrt{2}$ in 5 steps. It should be noted that the full-word system's vocabulary was limited to 89k, the total number of unique words in the acoustic data. Therefore, during training the OOV words were simply deleted (1.2% of the data). The direct acoustic-to-word (A2W) model also imposed this limited vocabulary constraint in recognition time. Table 1 shows the model perplexities (PPL)

Table 2: *Effect of input features, data perturbation, sequence-noise. The model contains 24M parameters, and was trained on 100 BPE units. Results are without external language model.*

fea.	pert.	seq. noise	SGDR	dev		test	
				clean	others	clean	other
MFCC				4.49	14.74	4.96	14.82
		×		4.47	13.97	4.85	13.96
	×			4.31	13.19	4.61	13.31
	×	×		*4.28*	*12.96*	*4.59*	*13.12*
GT				4.65	14.35	5.03	14.90
		×		4.46	13.95	4.87	14.56
	×			4.21	13.56	4.59	14.00
	×	×		*4.18*	*12.94*	*4.58*	*13.20*
GT	×	×	×	**4.01**	**12.58**	**4.30**	**12.75**

measured on the joint dev-clean and dev-other set. In order to compare the various models, we also present perplexities normalized by the number of running words. As can be seen, character and BPE-100 (indicating 100 replacement rules) models are competitive with BPE-10k models, despite modeling much longer sequences. Training a 4-layer LSTM for BPE-100 led to 9% relative improvement in PPL.

All models were trained with PyTorch [37]. The decoding hyper-parameters β_{seq}, β_{LM}, λ_{LM}, λ_{len}, λ_{ct}, τ, γ were optimized on dev-clean using 1-dimensional grid search iteratively. During recognition, the beam size was 30 without LM, and it was set to 80 with LM.

3. Experimental results

3.1. Optimizing dropout, effect of warm-restart, sequence-noise and perturbation using MFCC or GT features

In the first set of experiments, we trained the models only for 50 epochs without warm-restart, and focused on optimizing our model at the input-level. As Table 2 shows, data perturbation turned out to be more important than sequence-noise injection, nevertheless sequence-noise consistently improved the results further. Regardless of the choice of features, we ended up roughly at the same performance. We also conducted several experiments to find the optimal dropout settings. The results of tuning the encoder are summarized in Fig. 2. Despite the significant noise in the data points, results indicate optimal dropout and drop-connect values around 0.2. As can also be seen in Table 2, additional 50 epochs of training with warm-restart (SGDR) improved the model significantly. We note that this system already outperforms the lowest stand-alone sequence-to-sequence numbers published on this task [38].

3.2. Interaction of scheduled sampling and search configurations

In these experiments we gradually switched off the scheduled sampling in training, the coverage term, and the prefix-tree search constraint in Eq. 3. According to [32, 38], serious word error rate (WER) degradation can be observed without scheduled sampling. Indeed, we obtained the worst results with 100% teacher forcing. However, vocabulary constraints and coverage stabilized the search, and the effect of scheduled sampling faded (Table 3). Nevertheless, the best results on the development set were obtained when all three techniques were applied. We also note, irrespective of scheduled sampling, without the coverage term we observed large, over 10% relative WER variation depending on whether the search hyper-parameters were optimized on clean or other set. The variation reduced to below 0.5% with coverage term and vocabulary constraint, which clearly indicates a much more stable search configuration.

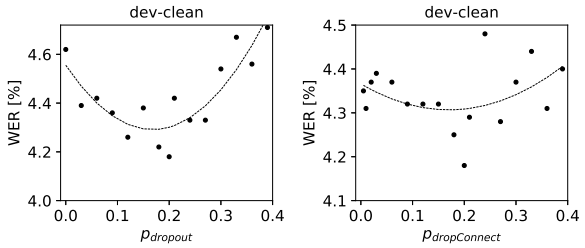

Figure 2: *Tuning dropout and drop-connect in the encoder.*

Table 3: *Interaction of scheduled-sampling, coverage term, and prefix-tree. Measuring WER with BPE-100 model w/o LM.*

scheduled sampling	prefix tree	cover-age	dev		test	
			clean	others	clean	other
×			4.42	14.06	4.65	14.18
×		×	4.34	13.37	**4.53**	13.67
×	×		4.44	15.77	5.35	15.39
×	×	×	**4.18**	**12.94**	4.58	13.20
			4.43	15.24	4.77	15.22
		×	4.43	13.38	4.70	13.57
	×		4.46	16.90	5.48	16.45
	×	×	4.28	13.01	4.72	**13.13**

3.3. Effect of modeling units, model size, sub-sampling rate, and language model

Table 4 and 5 show the results of varying modeling units, the sub-sampling rate in the encoder, and the effect of an external language model. As can be seen, the use of an LM resulted in a 30% relative WER improvement. Optimizing the sub-sampling rate for each representation unit, we found that, in contrast to other papers, larger BPE units are not beneficial if external language model is also used [11, 39]. Without an external language model (best results of each BPE system are indicated by italic font), however, large units (BPE-10k) achieved the best result on the more difficult test-other set. We explain these mixed results by the following. Clean sets can reasonably be recognized with small or even with zero language model weight, thus the easier language modeling offered by larger units is not beneficial. In contrast, "other" sets are acoustically more challenging, and larger modeling units lead to better language modeling by the decoder, which helps reducing the confusion. Similarly, an external LM helps a great deal reducing the WER. As Table 1 shows, dedicated language models are roughly in the same ballpark; and word constraints further reduce the perplexity gap between those models. Thus, finer grade modeling of speech (less sub-sampling, and smaller units) is a crucial factor recognizing speech robustly, especially if only 1000 hours of training data is available. The best results were obtained by the BPE-100 system, it achieved 2.96% and 8.48% WER on test-clean and test-other. We point out that these results are even better than the best hybrid system with lattice rescoring [40].

Aiming at improving the sequence-to-sequence model results further, we also carried out experiments adding more and wider layers to the encoder and/or decoder. In Table 5, we present the best numbers for each modeling unit. We were also curious if more parameters could compensate for the stronger sub-sampling. Comparing the best WERs without the external LM in Table 5 (indicated by the italic font) with the corresponding number in Table 4 significant improvement was measured (e.g. 12.31→11.72% on test other). However, the increased number of parameters could not outperform the best number in Table 5 when external LM was also used.

Table 4: *Effect of sub-sampling and modeling unit, with and without external language model.*

unit		sub.	without LM			with LM	
			#par.	test		test	
				clean	other	clean	other
char		4	24.0M	4.47	13.47	3.22	9.71
BPE	100	2	24.1M	***4.29***	*12.57*	***2.96***	***8.48***
		4		4.30	12.75	3.01	8.59
		8		4.52	13.62	3.45	9.46
	1k	4	24.5M	*4.34*	13.07	*3.33*	*9.52*
		8		4.37	*12.73*	3.55	10.80
	10k	4	29.1M	4.47	12.33	3.59	*9.04*
		8		*4.41*	***12.31***	*3.50*	9.47
		16		4.88	12.94	3.61	9.99
word		8	69.7M	5.02	13.72	4.01	10.84

Table 5: *Results with larger BPE sequence-to-sequence models.* layers *indicate num. of layers in the encoder and decoder.*

BPE	sub.	#param. w/o LM	#layers	LM	dev		test	
					clean	other	clean	other
100	4	41.9M	10+3	none	3.76	12.41	4.22	12.65
				LSTM	**2.61**	**7.92**	**3.00**	**8.51**
1k	4	42.9M	10+4	none	*3.61*	11.50	*3.97*	*11.72*
				LSTM	2.86	8.62	3.26	9.00
10k	4	47.5M	10+4	none	3.89	11.48	4.37	11.79
	8	46.5M	10+2		3.81	*11.45*	4.19	11.80
				LSTM	3.10	8.43	3.25	9.02
	16	37.5M	8+1	none	4.02	11.75	4.44	12.30

3.4. Vocabulary-free search with large LM, effect of beam size and discriminative training

Besides the 55M-parameter LM, we also tested a larger, 4-layer LSTM LM with 122M parameters (Table 1). This larger LM improved our BPE-100 results by 3-5% relative (Table 6, row 2-3). Surprisingly, running this system without constraining the search to valid words, we observed slight gain over the 200k vocabulary baseline. We hypothesized that the improvement could be related to the fact that the word fragment language models were trained on the fragments of all (up to 970k unique) words, thus they learned to model more than those 200k words of the base vocabulary. Indeed, extension of the 200k base lexicon also improved the vocabulary constrained results. Nonetheless, vocabulary free recognition is equally powerful, yet could end up in simpler search code. Analysis of the vocabulary-free recognition output revealed correct recognition of new words which were never seen during training, not even in fragmented form. Fig. 3 shows the analysis of the beam search. As can be seen, the optimal beam for recognition without language model or in clean condition is already reached at 30-50. However, results on the more difficult other set indicate an optimal beam size over hundred, a much higher value than what is usually used in the literature. Without any sophisticated approximation and pruning strategy, a search with beam over 100 together with a 120M-parameter NNLM is more than an order of magnitude slower than real-time using a single CPU core.

Lastly, following [41], we applied minimum Levenshtein-distance training to the model for 1 epoch. The loss was approximated by a 4-best list, and was measured at BPE-100 level. During training, we set the beam size to 10, and vocabulary constraint was enforced. The discriminative training improved our results without LM by 3% relative. The minimum error rate training together with large beam search resulted in our current best single-system (last row of Table 6).

Table 6: *BPE-100 experiments with larger LM, large or vocabulary free search. Effect of discriminative training (DT) and large beam.*

LM #param.	vocab.	beam-200 +DT	dev		test	
			clean	other	clean	other
0M	200k		4.01	12.58	4.30	12.75
60M			2.64	7.88	3.01	8.59
120M			2.53	7.44	2.92	8.25
	300k		2.42	7.37	2.73	8.15
	500k		2.41	7.36	2.73	8.15
	none		2.41	7.37	2.69	8.18
		×	**2.34**	**7.32**	**2.54**	**7.98**

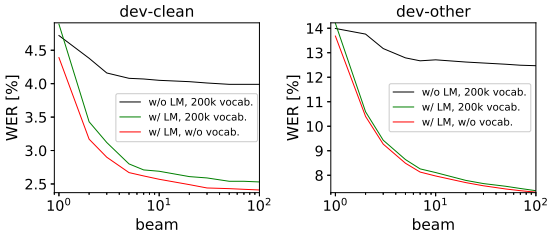

Figure 3: *Effect of beam size on word error rate (WER).*

3.5. Comparison with others

Table 7 compares our best systems to other state-of-the-art results published in the literature. As can be seen, our single system outperforms the previous best by a large margin, 29% relative improvement on test-clean and 7% relative gain on test-other. We also ROVER-combined several of our attention based systems ([42]), which differ only in modeling unit, input features, and model size. Our combination results are significantly better than the combined output of multiple hybrid systems [40].

4. Conclusions

By choosing training configurations and search parameters optimally for sequence-to-sequence models, we demonstrated that attention based encoder-decoder models are truly able to challenge the current state of the art, the hybrid model with neural network based rescoring. With a set of simple techniques, we could set up a new baseline on LibriSpeech (2.54%, 7.98% on test-clean and test-other), which significantly outperforms all previous single system results, including hybrid ones. To reach this level of performance, we found smaller modeling units and moderate application of frame-rate reduction crucial. Combination results also indicated the true potential of encoder-decoder models: a straightforward variation in model size, representation unit, and input features led to further improvement. Thus, we believe, in the future even a single system should be able to reach that performance level.

Table 7: *Comparison with others.*

model	ref.	LM	dev		test	
			clean	other	clean	other
hybrid	[43]	4gram	3.3	8.7	3.8	8.8
	[40]	RNN	3.12	8.28	3.51	8.58
wav2letter	[44]	conv	3.54	11.52	3.82	12.76
attention	[11]	LSTM	3.54	11.52	3.82	12.76
	[39]	LSTM	3.3	10.3	3.6	10.3
	[38]	none	-	-	4.5	13.3
	ours	none	3.61	11.50	3.97	11.72
		LSTM	**2.34**	**7.32**	**2.54**	**7.98**
8 sys. comb.	[40]		2.68	7.56	3.19	7.64
	ours		**2.23**	**6.63**	**2.54**	**7.28**

5. References

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

[2] A. Graves, "Sequence transduction with recurrent neural networks," 2012. [Online]. Available: http://arxiv.org/abs/1211.3711

[3] H. A. Bourlard and N. Morgan, *Connectionist Speech Recognition: A Hybrid Approach*. Norwell, MA, USA: Kluwer Academic Publishers, 1993.

[4] H. Hadian, H. Sameti, D. Povey, and S. Khudanpur, "End-to-end speech recognition using lattice-free MMI," in *Interspeech*, 2018, pp. 12–16.

[5] N. Kalchbrenner and P. Blunsom, "Recurrent continuous translation models," in *EMNLP*, 2013, pp. 1700–1709.

[6] A. Graves, "Generating sequences with recurrent neural networks," 2013. [Online]. Available: http://arxiv.org/abs/1308.0850

[7] D. Bahdanau, K. Cho, and Y. Bengio, "Neural machine translation by jointly learning to align and translate," in *ICLR*, 2015.

[8] I. Sutskever, O. Vinyals, and Q. V. Le, "Sequence to sequence learning with neural networks," pp. 3104–3112, 2014.

[9] Y. Wu *et al.*, "Google's neural machine translation system: Bridging the gap between human and machine translation," 2016. [Online]. Available: http://arxiv.org/abs/1609.08144

[10] W. Chan, N. Jaitly, Q. Le, and O. Vinyals, "Listen, attend and spell: a neural network for large vocabulary conversational speech recognition," in *ICASSP*, 2016, pp. 4960–4964.

[11] A. Zeyer, K. Irie, R. Schlüter, and H. Ney, "Improved training of end-to-end attention models for speech recognition," in *Interspeech*, 2018, pp. 7–11.

[12] C. Weng, J. Cui, G. Wang, J. Wang, C. Yu, D. Su, and D. Yu, "Improving attention based sequence-to-sequence models for end-to-end english conversational speech recognition," in *Interspeech*, 2018, pp. 761–765.

[13] C.-C. Chiu *et al.*, "State-of-the-art speech recognition with sequence-to-sequence models," in *ICASSP*, 2018, pp. 4774–4778.

[14] C. Chelba, M. Norouzi, and S. Bengio, "N-gram language modeling using recurrent neural network estimation," 2017. [Online]. Available: http://arxiv.org/abs/1703.10724

[15] Z. Tüske, R. Schlüter, and H. Ney, "Investigation on LSTM recurrent N-gram language models for speech recognition," in *Interspeech*, 2018, pp. 3358–3362.

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

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

[18] M. Schuster and K. K. Paliwal, "Bidirectional recurrent neural networks," *IEEE Trans. on Signal Processing*, vol. 45, no. 11, pp. 2673–2681, Nov 1997.

[19] J. K. Chorowski, D. Bahdanau, D. Serdyuk, K. Cho, and Y. Bengio, "Attention-based models for speech recognition," in *NIPS*, 2015, pp. 577–585.

[20] V. Nair and G. E. Hinton, "Rectified linear units improve restricted Boltzmann machines," in *ICML*, 2010, pp. 807–814.

[21] Ç. Gülçehre, O. Firat, K. Xu, K. Cho, L. Barrault, H. Lin, F. Bougares, H. Schwenk, and Y. Bengio, "On using monolingual corpora in neural machine translation," 2015. [Online]. Available: http://arxiv.org/abs/1503.03535

[22] A. Hannun *et al.*, "Deep Speech: Scaling up end-to-end speech recognition," 2014. [Online]. Available: http://arxiv.org/abs/1412.5567

[23] J. Chorowski and N. Jaitly, "Towards better decoding and language model integration in sequence to sequence models," 2016. [Online]. Available: http://arxiv.org/abs/1612.02695

[24] Z. Tu, Z. Lu, Y. Liu, X. Liu, and H. Li, "Modeling coverage for neural machine translation," in *ACL*, 2016, pp. 76–85.

[25] T. Ko, V. Peddinti, D. Povey, and S. Khudanpur, "Audio augmentation for speech recognition," in *Interspeech*, 2015, pp. 3586–3589.

[26] R. Schlüter, I. Bezrukov, H. Wagner, and H. Ney, "Gammatone features and feature combination for large vocabulary speech recognition," in *ICASSP*, 2007, pp. 649–652.

[27] G. Saon, Z. Tüske, K. Audhkhasi, and B. Kingsbury, "Sequence noise injected training for end-to-end speech recognition," in *ICASSP*, 2019.

[28] D. Bahdanau, J. Chorowski, D. Serdyuk, P. Brakel, and Y. Bengio, "End-to-end attention-based large vocabulary speech recognition," in *ICASSP*, 2016, pp. 4945–4949.

[29] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: A simple way to prevent neural networks from overfitting," *Journal of Machine Learning Research*, vol. 15, pp. 1929–1958, 2014.

[30] L. Wan, M. Zeiler, S. Zhang, Y. L. Cun, and R. Fergus, "Regularization of neural networks using DropConnect," in *ICML*, vol. 28, no. 3, 2013, pp. 1058–1066.

[31] "https://github.com/rsennrich/subword-nmt."

[32] S. Bengio, O. Vinyals, N. Jaitly, and N. Shazeer, "Scheduled sampling for sequence prediction with recurrent neural networks," in *NIPS*, 2015, pp. 1171–1179.

[33] S. Kim, T. Hori, and S. Watanabe, "Joint CTC-attention based end-to-end speech recognition using multi-task learning," in *ICASSP*, 2017, pp. 4835–4839.

[34] I. Loshchilov and F. Hutter, "SGDR: stochastic gradient descent with warm restarts," in *ICLR*, 2017.

[35] Y. Nesterov, "A method of solving a convex programming problem with convergence rate $O(1/k^2)$," in *Soviet Mathematics Doklady*, vol. 27, no. 2, 1983, pp. 372–376.

[36] S. Merity, N. S. Keskar, and R. Socher, "Regularizing and optimizing LSTM language models," in *ICLR*, 2018.

[37] A. Paszke, S. Gross, S. Chintala, G. Chanan, E. Yang, Z. DeVito, Z. Lin, A. Desmaison, L. Antiga, and A. Lerer, "Automatic differentiation in PyTorch," in *NIPS Workshop*, 2017.

[38] S. Sabour, W. Chan, and M. Norouzi, "Optimal completion distillation for sequence learning," in *ICLR*, 2019.

[39] K. Irie, R. Prabhavalkar, A. Kannan, A. Bruguier, D. Rybach, and P. Nguyen, "Model unit exploration for sequence-to-sequence speech recognition," 2019. [Online]. Available: http://arxiv.org/abs/1902.01955

[40] K. J. Han, A. Chandrashekaran, J. Kim, and I. R. Lane, "The CAPIO 2017 conversational speech recognition system," 2018. [Online]. Available: http://arxiv.org/abs/1801.00059

[41] R. Prabhavalkar, T. N. Sainath, Y. Wu, P. Nguyen, Z. Chen, C.-C. Chiu, and A. Kannan, "Minimum word error rate training for attention-based sequence-to-sequence models," in *ICASSP*, 2018, pp. 4839–4843.

[42] J. G. Fiscus, "A post-processing system to yield reduced word error rates: Recognizer output voting error reduction (ROVER)," in *ASRU*, 1997, pp. 347–354.

[43] H. Hadian, D. Povey, H. Sameti, J. Trmal, and S. Khudanpur, "Improving LF-MMI using unconstrained supervisions for ASR," in *SLT*, 2018, pp. 43–47.

[44] N. Zeghidour, Q. Xu, V. Liptchinsky, N. Usunier, G. Synnaeve, and R. Collobert, "Fully convolutional speech recognition," 2018. [Online]. Available: http://arxiv.org/abs/1812.06864