INTERSPEECH 2020
October 25–29, 2020, Shanghai, China

Single headed attention based sequence-to-sequence model for state-of-the-art results on Switchboard

Zoltán Tüske, George Saon, Kartik Audhkhasi, Brian Kingsbury*

IBM Research AI, Yorktown Heights, USA
zoltan.tuske@ibm.com

Abstract

It is generally believed that direct sequence-to-sequence (seq2seq) speech recognition models are competitive with hybrid models only when a large amount of data, at least a thousand hours, is available for training. In this paper, we show that state-of-the-art recognition performance can be achieved on the Switchboard-300 database using a single headed attention, LSTM based model. Using a cross-utterance language model, our single-pass speaker independent system reaches 6.4% and 12.5% word error rate (WER) on the Switchboard and CallHome subsets of Hub5'00, without a pronunciation lexicon. While careful regularization and data augmentation are crucial in achieving this level of performance, experiments on Switchboard-2000 show that nothing is more useful than more data. Overall, the combination of various regularizations and a simple but fairly large model results in a new state of the art, 4.8% and 8.3% WER on the Switchboard and CallHome sets, using SWB-2000 without any external data resources.

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

1. Introduction

Powerful neural networks have enabled the use of "end-to-end" speech recognition models that directly map a sequence of acoustic features to a sequence of words without conditional independence assumptions. Typical examples are attention based encoder-decoder [1] and recurrent neural network transducer models [2]. Due to training on full sequences, an utterance corresponds to a single observation from the view point of these models; thus, data sparsity is a general challenge for such approaches, and it is believed that these models are effective only when sufficient training data is available. Indeed, many end-to-end speech recognition papers focus on LibriSpeech, which has 960 hours of training audio. Nevertheless, the best performing systems follow the traditional hybrid approach [3], outperforming attention based encoder-decoder models [4, 5, 6, 7], and when less training data is used, the gap between "end-to-end" and hybrid models is more prominent [4, 8]. Several methods have been proposed to tackle data sparsity and overfitting problems; a detailed list can be found in Sec. 2. Recently, increasingly complex attention mechanisms have been proposed to improve seq2seq model performance, including stacking self and regular attention layers and using multiple attention heads in the encoder and decoder [5, 9].

We show that consistent application of various regularization techniques brings a simple, single-head LSTM attention based encoder-decoder model to state-of-the-art performance on Switchboard-300 (SWB-300), a task where data sparsity is more severe than LibriSpeech. We also note that remarkable performance has been achieved with single-head LSTM models in a recent study on language modeling [10].

*currently at Google Inc.

2. Methods to improve seq2seq models

In contrast to traditional hybrid models, where even recurrent networks are trained on randomized, aligned chunks of labels and features [11, 12], whole sequence models are more prone to memorizing the training samples. In order to improve generalization, many of the methods we investigate introduce additional noise, either directly or indirectly, to stochastic gradient descent (SGD) training to avoid narrow, local optima. The other techniques we study address the highly non-convex nature of training neural networks, ease the optimization process, and speed up convergence.

Weight decay adds the l_2 norm of the trainable parameters to the loss function, which encourages the weights to stay small unless necessary, and is one of the oldest techniques to improve neural network generalization. As shown in [13], weight decay can improve generalization by suppressing some of the effects of static noise on the targets.

Dropout randomly deactivates neurons with a predefined probability in every training step [14] to reduce co-adaptation of neurons.

DropConnect, which is similar in spirit to dropout, randomly deactivates connections between neurons by temporarily zeroing out weights [15].

Zoneout, which is also inspired by dropout and was especially developed for recurrent models [16], stochastically forces some hidden units to maintain their previous values. In LSTMs, the method is applied on the cell state or on the recurrent feedback of the output.

Label smoothing interpolates the hard label targets with a uniform distribution over targets, and improves generalization in many classification tasks [17].

Batch normalization (BN) accelerates training by standardizing the distribution of each layer's input [18]. In order to reduce the normalization mismatch between training and testing, we modify the original approach by freezing the batch normalization layers in the middle of the training when the magnitude of parameter updates is small. After freezing, the running statistics are not updated, batch statistics are ignored, and BN layers approximately operate as global normalization.

Scheduled sampling stochastically uses the token produced by a sequence model instead of the true previous token during training to mitigate the effects of exposure bias [19].

Residual networks address the problem of vanishing and exploding gradients by including skip connections [20] in the model that force the neural network to learn a residual mapping function using a stack of layers. Optimization of this residual mapping is easier, allowing the use of much deeper structures.

Curriculum learning simplifies deep neural network training by presenting training examples in a meaningful order, usually by increasing order of difficulty [21]. In seq2seq models, the input acoustic sequences are frequently sorted in order of increasing length [22].

http://dx.doi.org/10.21437/Interspeech.2020-1488

Speed and tempo perturbation changes the rate of speech, typically by $\pm 10\%$, with or without altering the pitch and timbre of the speech signal [23, 24]. The goal of these methods is to increase the amount of training data for the model.

Sequence noise injection adds structured sequence level noise generated from speech utterances to training examples to improve the generalization of seq2seq models [25]. As previously shown, input noise during neural network training encourages convergence to a local optimum with lower curvature, which indicates better generalization [26].

Weight noise adds noise directly to the network parameters to improve generalization [27]. This form of noise can be interpreted as a simplified form of Bayesian inference that optimizes a minimum description length loss [28].

SpecAugment masks blocks of frequency channels and blocks of time steps [4] and also warps the spectrogram along the time axis to perform data augmentation. It is closely related to [29].

3. Experimental setup

This study focuses on Switchboard-300, a standard 300-hour English conversational speech recognition task. Our acoustic and text data preparation follows the Kaldi [30] `s5c` recipe, which is based on the transcription release of Mississippi State University [31]. Our attention based seq2seq model is similar to [32, 33] and follows the structure of [34].

We extract 80-dimensional log-Mel filterbank features over 25ms frames every 10ms from the input speech signal. The input audio is speed and/or tempo perturbed with $\frac{5}{6}$ probability. Following [25], sequence noise mixed from up to 4 utterances is injected with 40% probability and 0.3 weight. The filterbank output is mean-and-variance normalized at the speaker level, and first (Δ) and second ($\Delta\Delta$) derivatives are also calculated. The final features presented to the network are also processed through a SpecAugment block that uses the `SM` policy [4] with $p = 0.3$ and no time warping.

The encoder network comprises 8 bidirectional LSTM layers with 1536 nodes per direction per layer [35, 36]. As shown in Fig. 1a, each LSTM block in the encoder includes a residual connection with a linear transformation that bypasses the LSTM, a 1024-dimensional linear reduction layer on the LSTM output, and batch-normalization (BN) of the block output. A pyramidal structure [33] in the first two LSTM layers reduces the frame rate by a factor of 4. The final dimension of the encoder output is 256, enforced by a linear bottleneck. We apply 30% dropout to the LSTM outputs and 30% drop-connect to the hidden-to-hidden matrices [15, 37]. As suggested by [38], the weight dropout is fixed for a batch of sequences.

The attention based decoder model is illustrated in Fig. 1b. The decoder models the sequence of 600 BPE units estimated on characters [39], where the BPE units are embedded in 256 dimensions. We use additive, location aware attention, without key/value transformations, and the attention is smoothed by 256, 5-dimensional kernels [40]. The decoder block consists of 2 unidirectional LSTM layers: one is a dedicated language-model-like component with 512 nodes that operates only on the embedded predicted symbol sequence, and the other is a 768 unit layer processing acoustic and symbol information. The output of both LSTMs is reduced to 256 dimensions by a linear bottleneck [41]. Fixed sequence-level weight dropout of 15% is applied in the decoder LSTMs, a dropout of 5% is applied to the embeddings, and a dropout of 15% is applied to the decoder LSTM outputs. The second LSTM in the decoder also uses zoneout, where the cell state update is deactivated with 15% probability and the recurrent feedback from the output maintains its previous value with 5% probability.

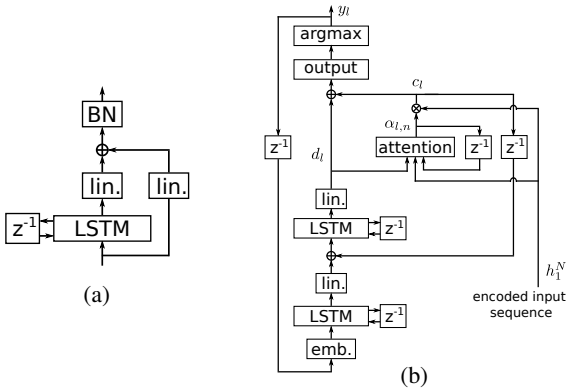

Figure 1: *(a) Building block of the encoder; (b) attention based decoder network used in the experiments.*

Overall, the model has 280M parameters, of which only 5.4M are in the decoder. Aiming at the best word error rate, this design choice is based on our observation that an external language model has significantly larger effect if the decoder is not over-parametrized [34]. The model is trained for 250 epochs on 32 P100 GPUs in less than 4 days using a PyTorch [42] implementation of distributed synchronous SGD with up to 32 sequences per GPU per batch. Training uses a learning rate of 0.03 and Nesterov momentum [43] of 0.9. The weight decay parameter is 4e-6, the label smoothing parameter is 0.35, and teacher forcing is fixed to 0.8 throughout training. In the first 3 epochs the learning rate is warmed up and batch size is gradually increased from 8 to 32 [44]. In the first 35 epochs, the neural network is trained on sequences sorted in ascending order of length of the input. Afterwards, batches are randomized within length buckets, ensuring that a batch always contains sequences with similar length. Weight noise from a normal distribution with mean 0.0 and variance 0.015 is switched on after 70 epochs. After 110 epochs, the updates of sufficient statistics in the batch-normalization layers are turned off, converting them into fixed affine transformations. The learning rate is annealed by 0.9 per epoch after 180 epochs of training, and simultaneously label smoothing is also switched off.

The external language model (LM) is built on the BPE segmentation of 24M words from the Switchboard and Fisher corpora (from which the SWB-300 corpus roughly corresponds to 3M words). It is trained for 40 epochs using label smoothing of 0.15 in the first 20 epochs. The baseline LM has 57M parameters and consists of 2 unidirectional LSTM layers with 2048 nodes [45] trained with drop-connect and dropout probabilities of 15%. The embedding layer has 512 nodes, and the output of the last LSTM is projected to 128 dimensions. When the LM is trained and evaluated across utterances, consecutive segments of a single-channel recording are grouped together up to 40 seconds. Perplexities (PPL) are measured at the word level on the concatenation of ground truth transcripts, while the WER is obtained by retaining the LM state of the single-best hypothesis of the preceding utterance.

Decoding uses simple beam search with a beam width of 60 hypotheses and no lexical prefix tree constraint [46]. The search performs shallow fusion of the encoder-decoder score, the external language model score, a length normalization term, and a coverage term [47, 48, 49]. For more details, please refer to [34]. Hub5'00 is used as a development set to optimize decoding hyperparameters, while Hub5'01 and RT03 are used as final test sets.

Table 1: *Effect of data preparation steps on WER [%] measured on Hub5'00, models are trained on SWB-300. The second row corresponds to the Kaldi* `s5c` *recipe.*

filter			w/o LM		w/ LM	
frag.	noise	dup.	swb	chm	swb	chm
			7.5	**14.3**	6.7	**12.6**
		✓	7.8	14.8	6.5	13.2
	✓		7.6	15.1	6.5	13.1
	✓	✓	**7.5**	15.1	6.6	13.3
✓	✓		7.6	14.6	**6.4**	12.7
✓	✓	✓	7.7	14.7	**6.4**	13.1

4. Experimental results

Our current setup is the result of incremental development. Keeping in mind that several other equally powerful setups probably exist, the focus of the following experiments is to investigate ours around the current optimum.

4.1. Effect of data preparation

We first investigate the importance of different data processing steps. The `s5c` Kaldi recipe includes a duplicate filtering step, in which the maximum number of occurrences of utterances with the same content is limited. We measure the impact of duplicate filtering and also the effect of filtering out word fragments and noise tokens from the training transcripts. Since the LM is trained on identically filtered transcripts from Fisher+Switchboard data, word fragment and noise token filters were applied consistently. The results are summarized in Table 1. Deactivating the duplicate filter is never harmful when an external LM is used, and the gains on CallHome can be substantial. Considering performance on the complete Hub5'00 data, the best systems either explicitly handle both word fragments and noise tokens or filter them all out. When an external LM is used, the best results are obtained when word fragment and noise token filters are activated and the duplicate filter is deactivated. This setting is also appealing in cases where the external LM may be trained on text data that will not contain word fragments or noise; thus, the remaining experiments are carried out with this system setting.

4.2. Ablation study

In a second set of experiments, we characterize the importance of each of the regularization methods described in Sec. 2 for our model performance by switching off one training method at a time without re-optimizing the remaining settings. In these experiments, decoding is performed without an external language model. Curriculum learning is evaluated by either switching to randomized batches after 35 epochs or leaving the sorting on throughout training. We also test the importance of Δ and $\Delta\Delta$ features [50]. Sorting the results by decreasing number of absolute errors on Hub5'00, Table 2 indicates that each regularization method contributes to the improved WER. SpecAugment is by far the most important method, while using Δ and $\Delta\Delta$ features or switching off the curriculum learning in the later stage of training have marginal but positive effects. Other direct input level perturbation steps (speed/tempo perturbation and sequence noise injection) are also key techniques that can be found in the upper half of the table. If we compare the worst and baseline models, we find that the relative performance difference between them is nearly unchanged by including the external LM in decoding. Without the LM, the gap is 18% relative, while with the LM the gap is 17% relative. This clearly underlines the importance of the regularization techniques.

Table 2: *Ablation study on the final training recipe, models are trained on SWB-300. WER is measured without using external LM.*

		WER [%]			#err.
		swb	chm	total	[word]
discarded ingredient	SpecAugment	9.1	17.3	13.2	5665
	speed/tempo pert.	8.2	15.5	11.9	5113
	dropout	8.0	15.6	11.8	5092
	label smoothing	7.9	15.6	11.8	5072
	sequence noise	7.8	15.6	11.7	5027
	weight noise	7.9	15.2	11.6	4973
	weight decay	**7.6**	15.1	11.4	4909
	DropConnect	7.7	15.1	11.4	4897
	BN freezing	7.8	14.9	11.4	4880
	scheduled samp.	7.7	14.8	11.3	4856
	zoneout (in dec.)	7.7	14.9	11.2	4831
	residual (in enc.)	7.7	14.7	11.2	4827
	random. batch	**7.6**	14.7	11.2	4794
	$+\Delta$, $+\Delta\Delta$	7.7	**14.6**	**11.1**	4792
baseline		**7.6**	**14.6**	**11.1**	**4775**

Table 3: *Optimizing dropout (dropo.), DropConnect (dropc.), layer and bottleneck (bn) size for LSTM LM, optionally modeling across utterances (x-utt.). WER is measured in shallow fusion with the best SWB-300 seq2seq ASR model.*

model					PPL_{word}		WER	
dropo.	dropc.	width	bn	x-utt.	CV	Hub5'00	swb	chm
15%	15%	2048	128		56.7	65.7	6.7	13.2
				✓	46.3	52.7	6.5	13.1
			-		52.9	61.2	6.6	13.2
				✓	44.1	50.1	6.4	**12.7**
		3072			53.9	64.0	6.7	13.2
30%	30%				50.3	58.3	6.4	13.1
				✓	**41.4**	**47.0**	**6.3**	12.8

4.3. Optimizing the language model

The following experiments summarize our optimization of the LM. Compared to our previous LM [25], we measure better perplexity and WER if no bottleneck is used before the softmax layer (rows 1 and 3 in Table 3). Increasing the model capacity to 122M parameters results in a significant gain in PPL only after the dropout rates are tuned (rows 3, 5 and 6). Similar to [51, 52], significant PPL gain is observed if the LM was trained across utterances. However, this PPL improvement does not translate into reduced WER with a bigger model when cross utterance modeling is used (rows 4 and 7). Thus, in all other experiments we use the smaller, 57M-parameter model.

4.4. Effect of beam size and number of parameters

A 280M-parameter model may be larger than is practical in many applications. Thus, we also conduct experiments to see if this model size is necessary for reasonable ASR performance. Models are trained without changing the training configuration, except that the size or number of LSTM layers is reduced. As Table 4 shows, although our smallest attention based model achieves reasonable results on this task, a significant loss is indeed observed with decreasing model size, especially on CallHome. Nevertheless, an external language model reduces the performance gap. A small, 57M-parameter model together with a similar size language model is only 5% relative worse than our largest model. We note that this model already outperforms the best published attention based seq2seq model [4], with roughly 66% fewer parameters.

Additional experiments are carried out to characterize the

Table 4: *Effect of model size, models are trained on SWB-300.*

enc.			dec.	#par. [M]	WER			
					w/o LM		w/ LM	
depth	LSTM	lin.	LSTM		swb	chm	swb	chm
6	512	384	512	28.5	9.4	17.0	7.4	14.6
	768	512		57.3	8.3	15.5	6.6	13.4
8				75.1	8.4	15.1	6.8	13.4
	1024	640	768	125.0	7.6	15.2	6.5	13.3
	1280	896		201.6	**7.5**	15.0	**6.4**	12.9
	1536	1024		280.1	7.6	**14.6**	**6.4**	**12.7**

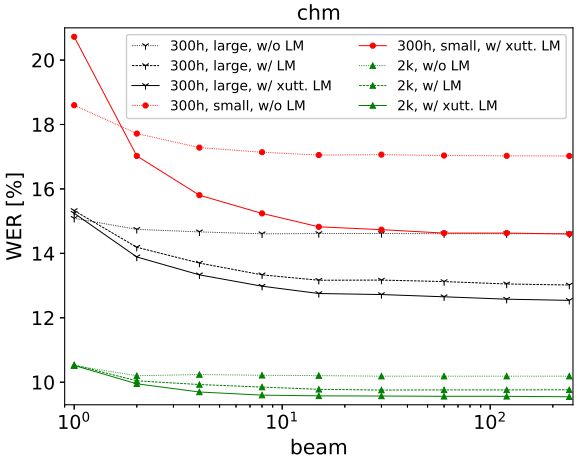

Figure 2: *Effect of beam size on word error rate (WER) measured on the CallHome (chm) part of Hub5'2000. "300h" indicate models trained on SWB-300, whereas "2k" corresponds to the 2000-hour Swichboard+Fisher training setup.*

search and modeling errors in decoding. The results of tuning the beam size and keeping the other search hyperparameters unchanged are shown in Fig. 2. "Small" denotes the 57M model, while "large" denotes the 280M model. When greedy search (beam 1) is used, the external language model increases WER, an effect that might be mitigated with re-optimized hyperparameters. Nevertheless, if a beam of at least 2 hypotheses is used, the positive effect of the language model is clear. We also observe that without the language model the search saturates much earlier, around beam 8, fluctuating within only a few absolute errors afterwards. On the contrary, decoding with the language model, we measure consistent but small gains with larger beams. The minimum number of word errors was measured with a relatively large beam of 240. The figure also shows that the effect of a cross-utterance language model grows with larger beams. Lastly, if the model is trained on 2000 hours of speech data (see next section), the extremely fast greedy decoding gives remarkably good performance. Although the importance of beam search decreases with an increased amount of training data, we still measure 10% relative degradation compared to a system with a cross-utterance LM and wide (240) beam search.

4.5. Experiments on Switchboard-2000

As a contrast to our best results on Switchboard-300, we also train two seq2seq models on the 2000-hour Switchboard+Fisher data. The first model consists of 10 encoder layers, has 360M parameters, and is trained for only 50 epochs. The second model (`lrg`) has 660M parameters, 14 encoder and 4 decoder layers, and training runs for 250 epochs. Our overall results on the Hub5'00 and other evaluation sets are summarized in Table 5. The results in Fig. 2 and Table 5 show that adding more training

Table 5: *Detailed results with the best performing systems on both SWB-300 and SWB-2000.*

SWB	ext. LM	xutt.	hub5'00		hub5'01			rt03	
			swb	chm	swb	swb2 p3	swb2 p4	swb	fsh
300			7.6	14.6	8.1	11.0	15.7	17.8	10.5
	✓		6.5	13.0	7.0	9.3	13.8	15.0	8.8
	✓	✓	**6.4**	**12.5**	**6.8**	**9.1**	**13.4**	**14.8**	**8.4**
2k			5.9	10.2	6.8	8.7	11.7	10.4	7.2
	✓		**5.5**	9.8	6.6	8.3	11.5	**9.8**	6.7
	✓	✓	5.6	**9.5**	**6.5**	**8.2**	**11.4**	**9.8**	**6.6**
2k-lrg			4.8	8.0	5.5	6.7	10.3	8.5	7.0
	✓	✓	**4.7**	**7.8**	**5.2**	**6.3**	**9.7**	**8.2**	**6.4**

data and using a fairly large model greatly improves the system, over 30% relative in some cases. For comparison with others, our best 2000-hour system reaches 7.1% and 6.0% WER on rt02 and rt04. It is also worth to note that human performance is at 6.0%, 4.5%, 4.7% WER on the rt0{2,3,4} sets, measured as in [53]. We observe that the regularization techniques, which are extremely important on the 300h setup, are also beneficial to train much larger models using big data. Considering the recognition speed and applicability of the `2k-lrg` model without using external language model, we measure 0.73 – 0.77 real-time factor and 6.5 – 6.4% total WER on Hub5'00 after varying the beam between 4 and 16, using a single core of an Intel Xeon Platinum 8280 processor and 8-bit integer weight quantization.

5. Comparison with the literature

For comparison with results in the literature we refer to the Switchboard-300 results in [4, 8, 54, 55] and the Switchboard-2000 results in [52, 54, 53, 56, 57, 58, 59]. Our 300-hour model not only outperforms the previous best attention based encoder-decoder model [4] by a large margin, it also surpasses the best hybrid systems with multiple LMs [8]. Our single system result on Switchboard-2000 is also better than the best system combination results reported to date.

6. Conclusions

We presented an attention based encoder-decoder setup which achieves state-of-the-art performance on both Switchboard 300 and 2000. A rather simple model built from LSTM layers and a decoder with a single-headed attention mechanism outperforms the standard hybrid approach. This is particularly remarkable given that in our model neither a pronunciation lexicon nor a speech model with explicit hidden state representations is needed. We also demonstrated that excellent results are possible with smaller models and with practically search-free, greedy decoding. The best results were achieved with a speaker independent model in a single decoding pass, using a minimalistic search algorithm, and without any attention mechanism in the language model. Thus, we believe that further improvements are still possible if we apply a more complicated sequence-level training criterion and speaker adaptation. As a further possible extension of this study, the training of conventional ASR models should also be revisited for fairer comparison of different modeling approaches.

7. References

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

[2] A. Graves, "Sequence transduction with recurrent neural networks," in *Representation Learning Workshop, ICML*, 2012.

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

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

[5] S. Karita *et al.*, "A comparative study on Transformer vs RNN in speech applications," in *ASRU*, 2019.

[6] C. Lüscher *et al.*, "RWTH ASR systems for LibriSpeech: Hybrid vs attention," in *Interspeech*, 2019, pp. 231–235.

[7] Y. Wang *et al.*, "Transformer-based acoustic modeling for hybrid speech recognition," 2019. [Online]. Available: http://arxiv.org/abs/1910.09799

[8] K. Irie *et al.*, "Training language models for long-span cross-sentence evaluation," in *ASRU*, 2019.

[9] A. Vaswani *et al.*, "Attention is all you need," in *NIPS*, 2017, pp. 5998–6008.

[10] S. Merity, "Single headed attention RNN: Stop thinking with your head," 2019. [Online]. Available: http://arxiv.org/abs/1911.11423

[11] G. Saon *et al.*, "Unfolded recurrent neural networks for speech recognition," in *Interspeech*, 2014, pp. 343–347.

[12] A.-r. Mohamed *et al.*, "Deep bi-directional recurrent networks over spectral windows," in *ASRU*, 2015, pp. 78–83.

[13] A. Krogh and J. A. Hertz, "A simple weight decay can improve generalization," in *NIPS*, 1992, pp. 950–957.

[14] G. E. Hinton *et al.*, "Improving neural networks by preventing co-adaptation of feature detectors," 2012. [Online]. Available: http://arxiv.org/abs/1207.0580

[15] L. Wan *et al.*, "Regularization of neural networks using DropConnect," in *ICML*, vol. 28, no. 3, 2013, pp. 1058–1066.

[16] D. Krueger *et al.*, "Zoneout: regularizing RNNs by randomly preserving hidden activations," in *ICLR*, 2017.

[17] C. Szegedy *et al.*, "Rethinking the inception architecture for computer vision," in *CVPR*, 2016, pp. 2818–2826.

[18] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," in *ICML*, 2015, pp. 448–456.

[19] S. Bengio *et al.*, "Scheduled sampling for sequence prediction with recurrent neural networks," in *NIPS*, 2015, pp. 1171–1179.

[20] K. He *et al.*, "Deep residual learning for image recognition," in *CVPR*, 2016, pp. 770–778.

[21] Y. Bengio *et al.*, "Curriculum learning," in *ICML*, 2009, pp. 41–48.

[22] D. Amodei *et al.*, "Deep speech 2: End-to-end speech recognition in English and Mandarin," in *ICML*, 2016, pp. 173–182.

[23] N. Kanda, R. Takeda, and Y. Obuchi, "Elastic spectral distortion for low resource speech recognition with deep neural networks," in *ASRU*, 2013, pp. 309–314.

[24] T. Ko *et al.*, "Audio augmentation for speech recognition," in *Interspeech*, 2015, pp. 3586–3589.

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

[26] C. M. Bishop, "Training with noise is equivalent to Tikhonov regularization," *Neural Computation*, vol. 7, no. 1, pp. 108–116, 1995.

[27] A. F. Murray and P. J. Edwards, "Enhanced MLP performance and fault tolerance resulting from synaptic weight noise during training," *IEEE Trans. on Neural Networks*, vol. 5, no. 5, pp. 792–802, 1994.

[28] A. Graves, "Practical variational inference for neural networks," in *NIPS*, 2011, pp. 545–552.

[29] Z. Zhong *et al.*, "Random erasing data augmentation," 2017. [Online]. Available: http://arxiv.org/abs/1708.04896

[30] D. Povey *et al.*, "The Kaldi speech recognition toolkit," in *ASRU*, 2011.

[31] "https://www.isip.piconepress.com/projects/switchboard."

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

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

[34] Z. Tüske, K. Audhkhasi, and G. Saon, "Advancing sequence-to-sequence based speech recognition," in *Interspeech*, 2019, pp. 3780–3784.

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

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

[37] N. Srivastava *et al.*, "Dropout: A simple way to prevent neural networks from overfitting," *Journal of Machine Learning Research*, vol. 15, pp. 1929–1958, 2014.

[38] Y. Gal and Z. Ghahramani, "A theoretically grounded application of dropout in recurrent neural networks," in *NIPS*, 2016, pp. 1019–1027.

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

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

[41] K. Veselý, M. Karafiát, and F. Grézl, "Convolutive bottleneck network features for LVCSR," in *ASRU*, 2011, pp. 42–47.

[42] A. Paszke *et al.*, "Automatic differentiation in PyTorch," in *NIPS Workshop*, 2017.

[43] 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.

[44] P. Goyal *et al.*, "Accurate, large minibatch SGD: Training imagenet in 1 hour," 2017. [Online]. Available: http://arxiv.org/abs/1706.02677

[45] M. Sundermeyer, R. Schlüter, and H. Ney, "LSTM neural networks for language modeling," in *Interspeech*, 2012, pp. 194–197.

[46] H. Ney *et al.*, "Improvements in beam search for 10000-word continuous speech recognition," in *ICASSP*, vol. 1, 1992, pp. 9–12.

[47] Ç. Gülçehre *et al.*, "On using monolingual corpora in neural machine translation," 2015. [Online]. Available: http://arxiv.org/abs/1503.03535

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

[49] Z. Tu *et al.*, "Modeling coverage for neural machine translation," in *ACL*, 2016, pp. 76–85.

[50] S. Furui, "Comparison of speaker recognition methods using statistical features and dynamic features," *IEEE Trans. on Acoustics, Speech, and Signal Processing*, vol. 29, no. 3, pp. 342–350, Jun. 1981.

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

[52] W. Xiong *et al.*, "The Microsoft 2017 conversational speech recognition system," in *ICASSP*, 2018, pp. 5934–5938.

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

[54] H. Hadian *et al.*, "End-to-end speech recognition using lattice-free MMI," in *Interspeech*, 2018, pp. 12–16.

[55] K. Audhkhasi *et al.*, "Forget a bit to learn better : Soft forgetting for CTC-based automatic speech recognition," in *Interspeech*, 2019, pp. 2618–2622.

[56] E. Battenberg *et al.*, "Exploring neural transducers for end-to-end speech recognition," in *ASRU*, 2017, pp. 206–213.

[57] G. Kurata *et al.*, "Language modeling with highway LSTM," in *ASRU*, 2017, pp. 244–251.

[58] T.-S. Nguyen *et al.*, "Improving sequence-to-sequence speech recognition training with on-the-fly data augmentation," 2019. [Online]. Available: https://arxiv.org/abs/1910.13296

[59] K. J. Han *et al.*, "The CAPIO 2017 conversational speech recognition system," 2018. [Online]. Available: http://arxiv.org/abs/1801.00059