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

Forget a Bit to Learn Better: Soft Forgetting for CTC-based Automatic Speech Recognition

Kartik Audhkhasi, George Saon, Zoltán Tüske, Brian Kingsbury, Michael Picheny

IBM Research AI, IBM T. J. Watson Research Center, Yorktown Heights, NY 10598
{kaudhkha,gsaon,zoltan.tuske,bedk,picheny}@us.ibm.com

Abstract

Prior work has shown that connectionist temporal classification (CTC)-based automatic speech recognition systems perform well when using bidirectional long short-term memory (BLSTM) networks unrolled over the whole speech utterance. This is because whole-utterance BLSTMs better capture long-term context. We hypothesize that this also leads to overfitting and propose soft forgetting as a solution. During training, we unroll the BLSTM network only over small non-overlapping chunks of the input utterance. We randomly pick a chunk size for each batch instead of a fixed global chunk size. In order to retain some utterance-level information, we encourage the hidden states of the BLSTM network to approximate those of a pre-trained whole-utterance BLSTM. Our experiments on the 300-hour English Switchboard dataset show that soft forgetting improves the word error rate (WER) above a competitive whole-utterance phone CTC BLSTM by an average of 7-9% relative. We obtain WERs of 9.1%/17.4% using speaker-independent and 8.7%/16.8% using speaker-adapted models respectively on the Hub5-2000 Switchboard/CallHome test sets. We also show that soft forgetting improves the WER when the model is used with limited temporal context for streaming recognition. Finally, we present some empirical insights into the regularization and data augmentation effects of soft forgetting.

Index Terms: automatic speech recognition, connectionist temporal classification, regularization

1. Introduction

End-to-end (E2E) automatic speech recognition (ASR) systems [1–20] have been the subject of significant recent research. Such systems aim to simplify the complex training and inference pipelines of conventional hybrid ASR systems [21, 22]. Hybrid systems combine Gaussian mixture models, hidden Markov models (HMMs) and various neural networks, and involve multiple stages of model building and alignment between the sequence of speech features and HMM context-dependent states. In contrast, end-to-end systems use recurrent neural networks (RNNs) and train the acoustic model in one-shot by either summing over all alignments through the connectionist temporal classification (CTC) loss [1] or learning the optimal alignment through an attention mechanism [14, 16]. The word error rate (WER) gap between E2E and hybrid ASR systems has reduced over time as shown by several works.

RNNs with long short-term memory (LSTM) [23] hidden units are the neural networks of choice for ASR systems. Bidirectional LSTM (BLSTM) networks are especially popular and consist of two LSTM networks at each layer that are unrolled forward and backward in time. For E2E ASR systems, the BLSTM network is unrolled over the entire length of the speech utterance. Whole-utterance unrolling enables a BLSTM network to better capture long-term context, which is especially useful given the lack of alignments. The control of remembering long-term context is left to the four trainable gates (input, forget, cell, and output) of the LSTM cell. Prior work has explored variants of the LSTM to control this behavior [24–26].

We hypothesize that whole-utterance unrolling of the BLSTM network leads to overfitting, even in the presence of well-known regularization techniques such as dropout [27]. This is especially detrimental to the WER of E2E ASR systems given limited training data, e.g. a few hundred hours of speech. We propose *soft forgetting* to combat this overfitting. First, we unroll the BLSTM network only over small non-overlapping chunks of the input acoustic utterance instead of the whole utterance. The hidden and cell states of the forward and backward LSTM networks are reset to zero at chunk boundaries. In order to prevent memorization of the fixed-sized chunk, we randomly perturb the chunk size across batches during training. Finally, we use twin regularization [29, 30] in order to retain some utterance-level context. Twin regularization adds the mean-squared error between the hidden states of the chunk-based BLSTM network and a pre-trained whole-utterance BLSTM network to the CTC loss. Since twin regularization promotes some remembering of context across chunks, we call our approach soft forgetting.

To the best of our knowledge, prior works have considered chunked-training of CTC ASR models primarily for streaming inference. For example, [31] gives an overview of temporal chunking, latency-controlled bidirectional RNNs [32], and lookahead convolutions [33]. However, soft forgetting additionally incorporates chunk jitter and twin regularization, and significantly improves both the offline/non-streaming and streaming WER of the CTC ASR system. We conduct experiments on the 300-hour English Switchboard data set and show that soft forgetting significantly improves the WER by 7-9% relative over a competitive phone CTC baseline across several test sets. We also present empirical evidence for the regularization and data augmentation effects of soft forgetting.

2. Soft Forgetting

Before discussing soft forgetting, we first give a brief overview of twin regularization [29,30] for its original application of closing the WER gap between uni-directional LSTM (ULSTM) and BLSTM networks. ULSTM networks lag BLSTM networks in terms of ASR WER because ULSTM networks only incorporate forward-in-time context whereas BLSTM networks additionally incorporate backward-in-time context. In its original formulation, twin regularization jointly trains two ULSTM networks operating forward and backward-in-time independently. The overall training loss is

$$\mathcal{L}_{\text{tot}}(\mathbf{y}|\mathbf{x},\Theta_{\text{f}},\Theta_{\text{b}}) = \mathcal{L}_{\text{CTC-f}}(\mathbf{y}|\mathbf{x},\Theta_{\text{f}}) + \mathcal{L}_{\text{CTC-b}}(\mathbf{y}|\mathbf{x},\Theta_{\text{b}}) + \lambda\mathcal{L}_{\text{twin}}(\mathbf{h}_{\text{f}},\mathbf{h}_{\text{b}}|\Theta_{\text{f}},\Theta_{\text{b}}) \quad (1)$$

http://dx.doi.org/10.21437/Interspeech.2019-2841

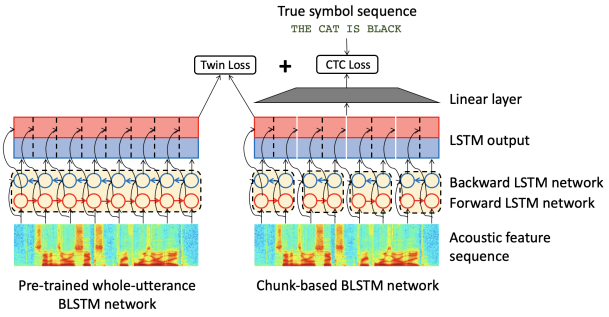

Figure 1: *Soft forgetting unrolls the BLSTM network over non-overlapping chunks and additionally uses a pre-trained whole utterance BLSTM network (left) to regularize the LSTM hidden states of the chunk-based BLSTM network (right).*

where the twin regularization loss is the mean-squared error between the LSTM hidden representations $\mathbf{h}_\text{f}$ and $\mathbf{h}_\text{b}$:

$$\mathcal{L}_\text{twin}(\mathbf{h}_\text{f}, \mathbf{h}_\text{b}|\Theta_\text{f}, \Theta_\text{b}) = ||\mathbf{h}_\text{f} - \mathbf{h}_\text{b}||_2^2 \ . \quad (2)$$

Θ_f and Θ_b denote the parameters of the forward and backward ULSTM networks respectively. $\mathbf{x}$ and $\mathbf{y}$ are the input acoustic and output label sequences, and $\lambda > 0$ is a scaling factor.

Figure 1 shows a block diagram of soft forgetting containing the following elements:

- **Chunk-based BLSTM network**: Soft forgetting trains a BLSTM network unrolled only over non-overlapping windows of C contiguous time steps of the input acoustic sequence. The intuition behind this choice is that whole-utterance BLSTMs tend to overfit the training data, especially when it is limited. Setting the hidden and cell states to 0 after every C time steps mitigates this overfitting, as we show via learning curves in Section 4. We pick C empirically based on heldout CTC loss.
- **Chunk size jitter**: We found that perturbing C from one batch to the next improves the generalization performance of the model. As we show later, adding this jitter has a data augmentation effect. Hence we jitter the chunk size C_batch of each batch as follows: $C_\text{batch} = C + u$, where $u \sim U(-A, A)$ is a uniformly-distributed discrete random variable over $[-A, A]$. We pick A empirically based on heldout CTC loss.
- **Twin regularization**: To incorporate some utterance-level context, we add a twin regularization loss in addition to the CTC loss. This loss is the mean-squared error between the hidden states of a pre-trained whole-utterance BLSTM network and the chunk-based BLSTM network being currently trained. We keep the weights of the whole-utterance BLSTM network fixed during training. This is in contrast to the original formulation of twin regularization where both the models are jointly trained.

Hence, the overall loss function for soft forgetting is:

$$\mathcal{L}_\text{tot}(\mathbf{y}|\mathbf{x}, \Theta) = \mathcal{L}_\text{CTC}(\mathbf{y}|\mathbf{x}, \Theta) + \lambda \mathcal{L}_\text{twin}(\mathbf{h}, \mathbf{h}_\text{whole}|\Theta) \ . \quad (3)$$

Setting $\lambda = 0$ results in *hard forgetting* where hidden or cell state information is not transferred across chunks. Our experiments show that a non-zero value of λ yields a significantly better WER compared to $\lambda = 0$, indicating the value of retaining some whole-utterance context through twin regularization.

We would like to point out that chunked training of the BLSTM network does not require any chunk-wise alignments and only affects the forward pass through the BLSTM network. We consider a batch of B utterances with N chunks each as a batch of BN small utterances. We reshape the resulting chunk-wise posterior vectors to construct the posterior sequence for the original whole utterances and use it to compute the CTC loss. Once the chunk-based BLSTM network is trained using soft forgetting, we discard the whole-utterance BLSTM network and perform inference without any modification.

3. Experiments and Results

We conducted all our experiments on the 300-hour English Switchboard task and trained both speaker independent (SI) and speaker adapted (SA) models. The SI models used 40-dimensional logMel features with a per-speaker cepstral mean subtraction (CMS) and without any vocal tract length normalization (VTLN). The SA models used 40-dimensional logMel features with per-speaker CMS and VTLN, 40-dimensional feature-space maximum likelihood linear regression (FMLLR) features, and 100-dimensional i-vectors, similar to the *feature fusion* system in our previous work [21].

3.1. Baseline CTC Models

We trained 6-layer BLSTM networks in PyTorch [34] with 512 hidden neurons each in the forward and backward directions. The output 1024-dimensional hidden vector was mapped to a 45-dimensional posterior probability vector representing the 44 phones and the blank symbol through a linear layer and softmax. All our models used deltas and double-deltas on the logMel features, and frame stacking and skipping with a rate of 2 for the logMel and FMLLR features. This resulted in 240-dimensional input acoustic features for the SI models and 420-dimensional input acoustic features for the SA models. A 4-gram LM was trained on training text from the SWB+Fisher corpus with a vocabulary of 30k words.

We trained all models using synchronous stochastic gradient descent (SGD) with a learning rate of 0.04, Nesterov momentum of 0.9 and a batch size of 128 over 4 V100 GPUs. The learning rate was multiplied by $\sqrt{0.5}$ per epoch after the first 10 epochs. We also used sequence noise injection [20] on the logMel features for both the SI and SA models. Sequence noise injection performs log-sum-exp on the logMel features sequences of the current utterance and a randomly-sampled utterance from the training set. It regularizes the model and can be also thought of as on-the-fly data augmentation since it creates new input feature sequences. There are two hyper-parameters – weight on the features of the randomly-sampled utterance and the probability of injecting sequence noise. We used values of (0.4,0.4) for these hyper-parameters for all the SI models and (0.2,0.4) for all the SA models based on heldout CTC loss.

Table 1 shows the WERs obtained by the baseline whole-utterance CTC models on the Hub5-2000 Switchboard (SWB) and CallHome (CH) test sets. We first conducted experiments with soft forgetting on the SI model and then on the SA model.

3.2. CTC Models with Soft Forgetting

We introduced various elements of soft forgetting in steps to gain an understanding of their impact on the WER. First, we trained the chunk-based BLSTM network with a fixed chunk size. We varied the chunk size from 5 to 50 stacked frames (100 msec to 1 sec) in steps of 5 stacked frames. We observed

that based on the heldout CTC loss, the optimal chunk size was 40 stacked frames. This model gave WERs of 12.7%/22.5% on the Hub5-2000 SWB/CH test sets as shown in Table 1.

Next, we introduced a chunk size jitter of $U(-2, 2)$ across training batches. This resulted in a further absolute reduction in WER of 0.6% on SWB and 1.0% on CH. At this point, the chunk-based BLSTM network with *hard forgetting* had comparable WER to the whole-utterance BLSTM network without sequence noise injection. We then introduced twin regularization over the last three BLSTM layers and varied λ over $\{0.001, 0.01, 0.1, 1.0\}$. We obtained the best heldout CTC loss at $\lambda = 0.01$. This gave a WER of 11.1% on SWB and 19.7% on CH, which we note is only 0.2% worse than the sequence noise injection baseline on SWB and 1.6% absolute better on CH. Finally, introducing sequence noise injection yielded a WER of 10.6% on SWB and 19.5% on CH. The benefit of sequence noise injection is diminished in the presence of soft forgetting because both techniques have the same effect of regularizing the model and performing on-the-fly data augmentation.

Table 1: *WERs for SI phone CTC models on the Hub5-2000 Switchboard and CallHome test sets.*

Model	SWB	CH
Baseline		
Whole-utterance BLSTM	12.0	21.8
+ seq noise	10.9	21.3
Soft forgetting		
Chunk-based BLSTM (C = 40)	12.7	22.5
+ chunk size jitter (A = 2)	12.1	21.5
+ twin reg (λ = 0.01)	11.1	19.7
+ seq noise	**10.6**	**19.5**

We also performed a grid search by jointly varying chunk size over $\{5, 10, \ldots, 50\}$ as before and chunk jitter over {1,2,...,10} on the best model including twin regularization and sequence noise. It turns out that most values of chunk size ≥ 20 and chunk jitter ≥ 6 yield almost equally-low heldout losses. Upon picking the optimal chunk size of 40 and chunk jitter of 10, we obtained only a minor improvement of 0.2% in the CH WER and no improvement on SWB.

Next, we applied the soft forgetting recipe to the SA model. Table 2 shows the result of the baseline CTC model and the model with soft forgetting across five test sets [21]. We observe that soft forgetting obtains 0.9% and 1.6% absolute improvements over the baseline SA model for the SWB and CH test sets. For reference, we also show the WERs of the SI models from Table 1. Soft forgetting improves the average WER for the SI model by 1.3% absolute and the SA model by 1.4% absolute.

We then performed state-level minimum Bayes risk (sMBR) [40] sequence training of these models, and performed

Table 2: *WERs for SI and SA phone CTC models on the Hub5-2000 Switchboard, CallHome, RT02, RT03, and RT04 test sets.*

Model	SWB	CH	RT02	RT03	RT04	Avg
SI						
Baseline	10.9	21.3	19.3	19.2	17.1	17.6
Soft forgetting	**10.6**	**19.5**	**18.0**	**17.2**	**16.2**	**16.3**
SA						
Baseline	10.6	19.7	18.1	17.1	15.9	16.3
Soft forgetting	**9.7**	**18.1**	**16.4**	**15.6**	**14.6**	**14.9**

Table 3: *WERs for some SI/SA models from literature and our phone CTC models with soft forgetting on the Hub5-2000 SWB and CH test sets. * models used speed perturbation. †BPE-X refers to X units learned by "Byte Pair Encoding" [39].*

Model	LM	Label	SWB	CH
Hybrid				
LF MMI* [35]	4-gram	CD states	9.6	19.3
SA BLSTM (ours)	4-gram	CD states	9.6	16.8
+ sMBR	4-gram	CD states	8.8	16.5
+ LSTM LM	LSTM	CD states	7.8	15.5
End-to-End				
EE LF MMI* [36]	4-gram	Biphones	10.9	20.6
+ RNN LM	RNN	Biphones	9.3	18.9
Attention* [37]	None	Characters	12.2	23.3
Attention [38]	LSTM	BPE-1K†	11.8	25.7
Attention [20]	LSTM	BPE-600†	10.0	21.7
CTC [20]	4-gram	Phones	11.0	20.5
Soft forgetting				
CTC SI	4-gram	Phones	10.6	19.5
+ sMBR	4-gram	Phones	10.2	18.5
+ LSTM LM	LSTM	Phones	**9.6**	**17.7**
+ Speed Perturb*	4-gram	Phones	9.9	18.7
+ sMBR	4-gram	Phones	9.7	18.1
+ LSTM LM	LSTM	Phones	**9.1**	**17.4**
CTC SA	4-gram	Phones	9.7	18.1
+ sMBR	4-gram	Phones	9.4	17.6
+ LSTM LM	LSTM	Phones	**8.7**	**16.8**

rescoring of the resulting lattices with a LSTM language model (NNLM) trained on the Fisher+SWB corpus. The embedding layer of the NNLM has 512 nodes, followed by 2 LSTM layers, each having 2048 nodes. Before the softmax-based estimation of the 30k-dimensional word posterior vector, the feature space was reduced to 128 by a linear bottleneck layer. We used a combination of DropConnect [41] and Dropout [27] to regularize the model. Training used SGD with an initial learning rate of 0.01 and Nesterov momentum of 0.9. After 20 epochs of training, the learning rate was annealed by a factor of $\sqrt{0.5}$ in 15 steps. We obtain final WERs of 9.6%/17.7% for the SI and 8.7%/16.8% for the SA phone CTC models with soft forgetting.

In order to gauge the performance of our phone CTC models with soft forgetting, we show the WERs of various models from the literature on the Hub5-2000 SWB and CH test sets in Table 3. We also performed data augmentation using speed perturbation (0.9x, 1x, and 1.1x) for our SI system to be able to compare it to other systems using data augmentation [35–37]. Speed perturbation further improves our SI system to 9.1%/17.4%. Our models compare favorably to all the end-to-end systems. The WER improvement is especially large on the more challenging CH test set. Soft forgetting also significantly helps to reduces the WER gap between phone CTC and our state-of-the-art SA hybrid BLSTM network with feature fusion that uses a complex multi-stage training pipeline and 32k CD states as output. Specifically, we reduce the WER gap between our SA hybrid BLSTM and phone CTC by 71% for SWB and 74% for CH compared to our prior work [20] that used only sequence noise injection.

Next, we analyze the case of decoding with limited latency. We used our SI models before sMBR and NNLM rescoring for this purpose and believe similar conclusions can be drawn using the SA models. For these experiments, we unrolled the BLSTM

network over non-overlapping chunks during inference. The hidden and cell states of the forward LSTM network are copied over from one chunk to the next as it improved the WER. The backward LSTM hidden and cell states are reset to zero.

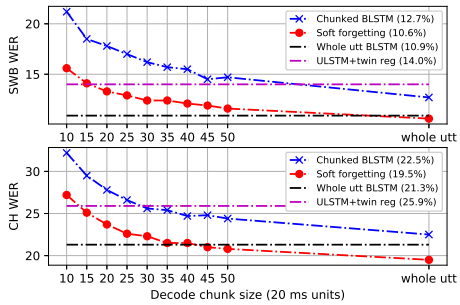

Figure 2: *This figure shows the variation of SWB and CH WERs with decoding chunk size for various SI models.*

Figure 2 shows the WERs of the chunk-based BLSTM network with/without soft forgetting versus the decoding chunk size. For each choice of decoding chunk size, we report the best WER across all training chunk sizes. For reference, we have also shown the WERs of a whole-utterance BLSTM network and a competitive whole-utterance ULSTM network trained with twin regularization using the whole-utterance BLSTM network. We observe that the chunk-based BLSTM network trained with soft forgetting significantly improves upon the chunk-based BLSTM network across all decoding chunk sizes.

4. Analysis

4.1. Regularization Effect

Our starting hypothesis was that whole-utterance BLSTM networks tend to overfit the data and soft forgetting is a way to mitigate this. Figure 3 shows the training and heldout CTC losses as training proceeds for the baseline whole-utterance SI BLSTM network and the chunk-based SI BLSTM network with soft forgetting. We observe that the converged training and heldout losses for the chunk-based BLSTM network with soft forgetting are significantly closer when compared with the whole-utterance BLSTM network. This indicates that soft forgetting does indeed regularize the model.

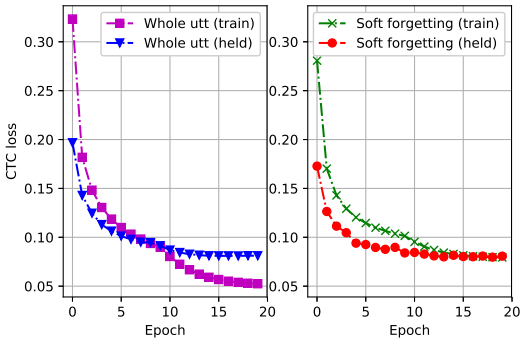

Figure 3: *This figure shows CTC losses for SI models.*

4.2. Data Augmentation Effect

Another intuition behind soft forgetting is that resetting the hidden and cell states of the BLSTM network after a random chunk size effectively creates data augmentation. This random forgetting leads to different hidden representations output by the BLSTM network for the same input acoustic feature sequence and creates different training samples. We used our best SA BLSTM network with soft forgetting and forward-passed the Hub5-2000 SWB test set. For each utterance, we unrolled the BLSTM network over the entire utterance and also over chunks of sizes $\{30, 31, \ldots, 50\}$ representing the chunk sizes used in training. We reset the hidden and cell states to zero after each chunk. We then computed a 1024-dimensional representation by time-averaging the output of the final BLSTM layer and projected it down to 2 dimensions using t-distributed Stochastic Neighbor Embedding (t-SNE) [42].

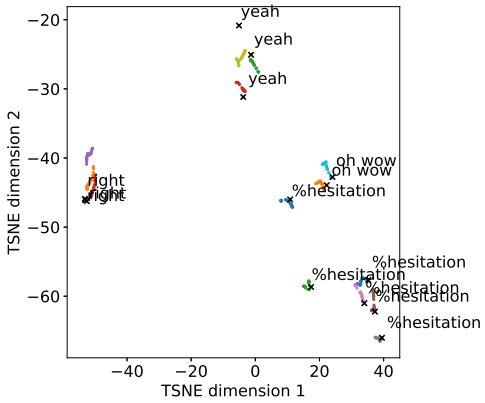

Figure 4: *This figure shows the t-SNE embeddings of our SA BLSTM network representations for some utterances from the SWB test set after hard forgetting with varying chunk sizes (colored dots) and whole utterance unrolling (black crosses).*

Figure 4 shows t-SNE embeddings for several short utterances and their reference transcripts. As expected, the embeddings show acoustic clustering across utterances, e.g. all utterances with hesitations cluster in one region of the space. More importantly, we observe that forgetting hidden and cell states after chunks of different sizes perturbs the embeddings in a local neighborhood for each utterance, attributed to the twin regularization loss used during training. The nature of this perturbation is different for each utterance and is not simply an independent and identically-distributed noise, attributed to the highly non-linear nature of the underlying representation. We believe this perturbation is an effective form of data augmentation.

5. Conclusion and Future Work

We introduced a novel algorithm, soft forgetting, to train better CTC ASR models. It consists of three elements. First, the BLSTM network is unrolled only over non-overlapping chunks of input acoustic frames instead of the whole sequence. The hidden and cell states are hence forgotten from one chunk to the next. Second, the chunk duration is perturbed across training batches. Finally, the CTC loss is regularized by promoting the BLSTM's hidden representations to be close to those from a pre-trained whole-utterance BLSTM network. Our experiments on SI and SA phone CTC models on the English Switchboard data set show that soft forgetting improves the WER by 7-9% relative compared to a competitive phone CTC baseline, and also helps close the WER gap with a state-of-the-art hybrid BLSTM network by around 70%. Future work will focus on exploring other methods to achieve soft forgetting and analyzing its impact on the behavior of the BLSTM network.

6. References

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

[2] H. Sak, A. Senior, K. Rao, and F. Beaufays, "Fast and accurate recurrent neural network acoustic models for speech recognition," in *Proc. Interspeech*, 2015.

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

[4] Y. Miao, M. Gowayyed, X. Na, T. Ko, F. Metze, and A. Waibel, "An empirical exploration of CTC acoustic models," in *Proc. ICASSP*, 2016, pp. 2623–2627.

[5] A. Graves and N. Jaitly, "Towards end-to-end speech recognition with recurrent neural networks," in *Proc. ICML*, vol. 14, 2014, pp. 1764–1772.

[6] G. Zweig, C. Yu, J. Droppo, and A. Stolcke, "Advances in all-neural speech recognition," in *Proc. ICASSP*, 2017, pp. 4805–4809.

[7] H. Soltau, H. Liao, and H. Sak, "Neural Speech Recognizer: Acoustic-to-Word LSTM Model for Large Vocabulary Speech Recognition," in *Proc. Interspeech*, 2017.

[8] K. Audhkhasi, B. Kingsbury, B. Ramabhadran, G. Saon, and M. Picheny, "Building competitive direct acoustics-to-word models for English conversational speech recognition," in *Proc. ICASSP*, 2018, pp. 959–963.

[9] C. Yu, C. Zhang, C. Weng, J. Cui, and D. Yu, "A multistage training framework for acoustic-to-word model," in *Proc. Interspeech*, 2018, pp. 786–790.

[10] R. Sanabria and F. Metze, "Hierarchical multi task learning with CTC," in *Proc. SLT*, 2018.

[11] K. Krishna, T. Toshniwal, and K. Livescu, "Hierarchical multitask leaning for CTC-based speech recognition," *arXiv preprint arXiv:1807.06234*, 2018.

[12] A. Hannun, C. Case, J. Casper, B. Catanzaro, G. Diamos, E. Elsen, R. Prenger, S. Satheesh, S. Sengupta, A. Coates, and A. Y. Ng, "Deep speech: Scaling up end-to-end speech recognition," *arXiv preprint arXiv:1412.5567*, 2014.

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

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

[15] A. L. Maas, Z. Xie, D. Jurafsky, and A. Y. Ng, "Lexicon-free conversational speech recognition with neural networks," in *Proc. HLT-NAACL*, 2015, pp. 345–354.

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

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

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

[19] J. Li, G. Ye, A. Das, R. Zhao, and Y. Gong, "Advancing Acoustic-to-Word CTC Model," *arXiv preprint arXiv:1803.05566*, 2018.

[20] G. Saon, Z. Tuske, K. Audhkhasi, and B. Kingsbury, "Sequence noise injected training for end-to-end speech recognition," in *(To appear) Proc. ICASSP*, 2019.

[21] G. Saon, G. Kurata, T. Sercu, K. Audhkhasi, S. Thomas, D. Dimitriadis, X. Cui, B. Ramabhadran, M. Picheny, L. Lim, B. Roomi, and P. Hall, "English conversational telephone speech recognition by humans and machines," in *Proc. Interspeech*, 2017.

[22] W. Xiong, J. Droppo, X. Huang, F. Seide, M. Seltzer, A. Stolcke, D. Yu, and G. Zweig, "Achieving human parity in conversational speech recognition," *arXiv preprint arXiv:1610.05256*, 2016.

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

[24] J. Chung, C. Gulcehre, K. Cho, and Y. Bengio, "Empirical evaluation of gated recurrent neural networks on sequence modeling," *arXiv preprint arXiv:1412.3555*, 2014.

[25] R. Jozefowicz, W. Zaremba, and I. Sutskever, "An empirical exploration of recurrent network architectures," in *Proc. ICML*, 2015, pp. 2342–2350.

[26] G. Zhou, J. Wu, C. Zhang, and Z. Zhou, "Minimal gated unit for recurrent neural networks," *International Journal of Automation and Computing*, vol. 13, no. 3, pp. 226–234, 2016.

[27] 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.

[28] H. Liu, S. Jin, and C. Zhang, "Connectionist temporal classification with maximum entropy regularization," in *Advances in Neural Information Processing Systems*, 2018, pp. 831–841.

[29] D. Serdyuk, N. R. Ke, A. Sordoni, A. Trischler, C. Pal, and Y. Bengio, "Twin networks: Matching the future for sequence generation," *arXiv preprint arXiv:1708.06742*, 2017.

[30] M. Ravanelli, D. Serdyuk, and Y. Bengio, "Twin regularization for online speech recognition," *arXiv preprint arXiv:1804.05374*, 2018.

[31] E. Battenberg *et al.*, "Reducing bias in production speech models," *arXiv preprint arXiv:1705.04400*, 2017.

[32] K. Chen and Q. Huo, "Training deep bidirectional LSTM acoustic model for LVCSR by a context-sensitive-chunk BPTT approach," *IEEE/ACM Transactions on Audio, Speech and Language Processing (TASLP)*, vol. 24, no. 7, pp. 1185–1193, 2016.

[33] C. Wang, D. Yogatama, A. Coates, T. Han, A. Hannun, and B. Xiao, "Lookahead convolution layer for unidirectional recurrent neural networks," in *Proc. ICLR Workshop*, 2016.

[34] 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.

[35] D. Povey, V. Peddinti, D. Galvez, P. Ghahremani, V. Manohar, X. Na, Y. Wang, and S. Khudanpur, "Purely Sequence-Trained Neural Networks for ASR Based on Lattice-Free MMI," in *Proc. Interspeech*, 2016, pp. 2751–2755.

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

[37] 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," *Proc. Interspeech*, pp. 761–765, 2018.

[38] A. Zeyer, K. Irie, R. Schlüter, and H. Ney, "Improved training of end-to-end attention models for speech recognition," *arXiv preprint arXiv:1805.03294*, 2018.

[39] P. Gage, "A new algorithm for data compression," *The C Users Journal*, vol. 12, no. 2, pp. 23–38, 1994.

[40] B. Kingsbury, "Lattice-based optimization of sequence classification criteria for neural-network acoustic modeling," in *Proc. ICASSP*, 2009, pp. 3761–3764.

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

[42] L. Maaten and G. Hinton, "Visualizing data using t-SNE," *Journal of machine learning research*, vol. 9, no. Nov, pp. 2579–2605, 2008.