Jasper: An End-to-End Convolutional Neural Acoustic Model

Jason Li[1], Vitaly Lavrukhin[1], Boris Ginsburg[1], Ryan Leary[1], Oleksii Kuchaiev[1],
Jonathan M. Cohen[1], Huyen Nguyen[1], Ravi Teja Gadde[2]

[1]NVIDIA, Santa Clara, USA
[2]New York University, New York, USA

{`jasoli,vlavrukhin,bginsburg,rleary,okuchaiev,jocohen,chipn`}`@nvidia.com`, `rtg267@nyu.edu`

Abstract

In this paper we report state-of-the-art results on LibriSpeech among end-to-end speech recognition models without any external training data. Our model, Jasper, uses only 1D convolutions, batch normalization, ReLU, dropout, and residual connections. To improve training, we further introduce a new layer-wise optimizer called NovoGrad. Through experiments, we demonstrate that the proposed deep architecture performs as well or better than more complex choices. Our deepest Jasper variant uses 54 convolutional layers. With this architecture, we achieve 2.95% WER using a beam-search decoder with an external neural language model and 3.86% WER with a greedy decoder on LibriSpeech test-clean. We also report competitive results on Wall Street Journal and the Hub5'00 conversational evaluation datasets.

Index Terms: speech recognition, convolutional networks, time-delay neural networks

1. Introduction

Conventional automatic speech recognition (ASR) systems typically consist of several independently learned components: an acoustic model to predict context-dependent sub-phoneme states (senones) from audio, a graph structure to map senones to phonemes, and a pronunciation model to map phonemes to words. Hybrid systems combine hidden Markov models to model state dependencies with neural networks to predict states [1, 2, 3, 4]. Newer approaches such as end-to-end (E2E) systems reduce the overall complexity of the final system.

Our research builds on prior work that has explored using time-delay neural networks (TDNN), other forms of convolutional neural networks, and Connectionist Temporal Classification (CTC) loss [5, 6, 7]. We took inspiration from wav2letter [7], which uses 1D-convolution layers. Liptchinsky et al. [8] improved wav2letter by increasing the model depth to 19 convolutional layers and adding Gated Linear Units (GLU) [9], weight normalization [10] and dropout.

By building a deeper and larger capacity network, we aim to demonstrate that we can match or outperform non end-to-end models on the LibriSpeech and 2000hr Fisher+Switchboard tasks. Like wav2letter, our architecture, Jasper, uses a stack of 1D-convolution layers, but with ReLU and batch normalization [11]. We find that ReLU and batch normalization outperform other activation and normalization schemes that we tested for convolutional ASR. As a result, Jasper's architecture contains only 1D convolution, batch normalization, ReLU, and dropout layers – operators highly optimized for training and inference on GPUs.

It is possible to increase the capacity of the Jasper model by stacking these operations. Our largest version uses 54 convolutional layers (333M parameters), while our smaller model uses 34 (201M parameters). We use residual connections to enable this level of depth. We investigate a number of residual options and propose a new residual connection topology we call *Dense Residual (DR)*.

Integrating our best acoustic model with a Transformer-XL [12] language model allows us to obtain new state-of-the-art (SOTA) results on LibriSpeech [13] test-clean of 2.95% WER and SOTA results among end-to-end models[1] on LibriSpeech test-other. We show competitive results on Wall Street Journal (WSJ), and 2000hr Fisher+Switchboard (F+S). Using only greedy decoding without a language model we achieve 3.86% WER on LibriSpeech test-clean.

This paper makes the following contributions:

1. We present a computationally efficient end-to-end convolutional neural network acoustic model.

2. We show ReLU and batch norm outperform other combinations for regularization and normalization, and residual connections are necessary for training to converge.

3. We introduce *NovoGrad*, a variant of the Adam optimizer [15] with a smaller memory footprint.

4. We improve the SOTA WER on LibriSpeech test-clean.

2. Jasper Architecture

Jasper is a family of end-to-end ASR models that replace acoustic and pronunciation models with a convolutional neural network. Jasper uses mel-filterbank features calculated from 20ms windows with a 10ms overlap, and outputs a probability distribution over characters per frame[2]. Jasper has a block architecture: a Jasper BxR model has B blocks, each with R sub-blocks. Each sub-block applies the following operations: a 1D-convolution, batch norm, ReLU, and dropout. All sub-blocks in a block have the same number of output channels.

Each block input is connected directly into the last sub-block via a residual connection. The residual connection is first projected through a 1x1 convolution to account for different numbers of input and output channels, then through a batch norm layer. The output of this batch norm layer is added to the output of the batch norm layer in the last sub-block. The result of this sum is passed through the activation function and dropout to produce the output of the current block.

---

[2] Work was conducted while the author was at NVIDIA

[1] We follow Hadian et. al's definition of end-to-end [14]: "flat-start training of a single DNN in one stage without using any previously trained models, forced alignments, or building state-tying decision trees."

[2] We use 40 features for WSJ and 64 for LibriSpeech and F+S.
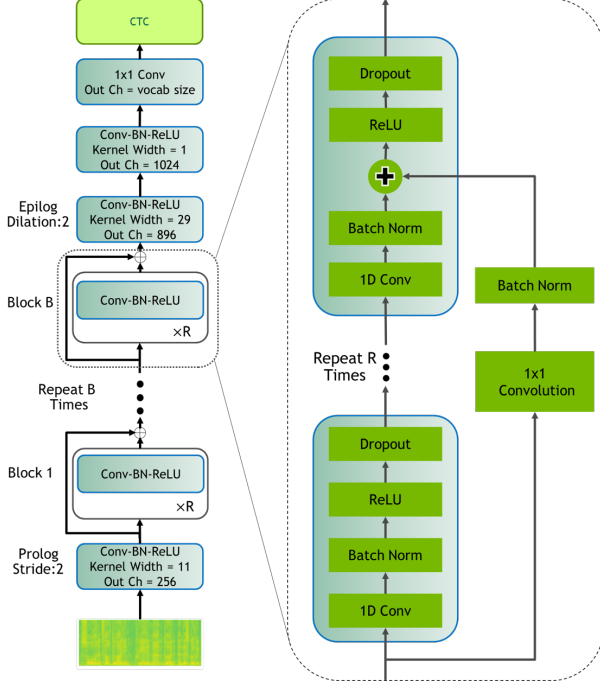
Figure 1: *Jasper BxR model: B - number of blocks, R - number of sub-blocks.*

Figure 2: *Jasper Dense Residual*

The sub-block architecture of Jasper was designed to facilitate fast GPU inference. Each sub-block can be fused into a single GPU kernel: dropout is not used at inference-time and is eliminated, batch norm can be fused with the preceding convolution, ReLU clamps the result, and residual summation can be treated as a modified bias term in this fused operation.

All Jasper models have four additional convolutional blocks: one pre-processing and three post-processing. See Figure 1 and Table 1 for details.

Table 1: *Jasper 10x5: 10 blocks, each consisting of 5 1D-convolutional sub-blocks, plus 4 additional blocks.*

# Blocks	Block	Kernel	# Output Channels	Dropout	# Sub Blocks
1	Conv1	11 *stride=2*	256	0.2	1
2	B1	11	256	0.2	5
2	B2	13	384	0.2	5
2	B3	17	512	0.2	5
2	B4	21	640	0.3	5
2	B5	25	768	0.3	5
1	Conv2	29 *dilation=2*	896	0.4	1
1	Conv3	1	1024	0.4	1
1	Conv4	1	# graphemes	0	1

We also build a variant of Jasper, *Jasper Dense Residual* (DR). Jasper DR follows DenseNet [16] and DenseRNet [17], but instead of having dense connections within a block, the output of a convolution block is added to the inputs of all the following blocks. While DenseNet and DenseRNet concatenates the outputs of different layers, Jasper DR adds them in the same way that residuals are added in ResNet. As explained below, we find addition to be as effective as concatenation.
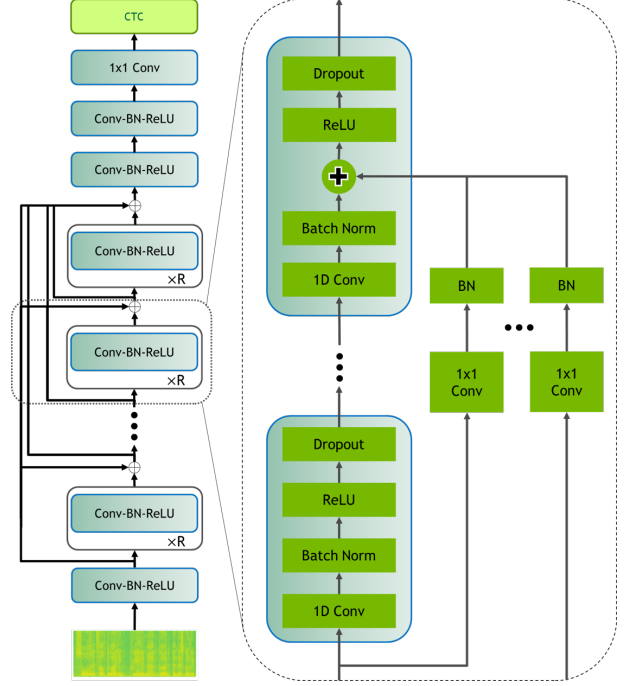
2.1. Normalization and Activation

In our study, we evaluate performance of models with:

- 3 types of normalization: batch norm [11], weight norm [10], and layer norm [18]
- 3 types of rectified linear units: ReLU, clipped ReLU (cReLU), and leaky ReLU (lReLU)
- 2 types of gated units: gated linear units (GLU) [9], and gated activation units (GAU) [19]

All experiment results are shown in Table 2. We first experimented with a smaller Jasper5x3[3] model to pick the top 3 settings before training on larger Jasper models. We found that layer norm with GAU performed the best on the smaller model. Layer norm with ReLU and batch norm with ReLU came second and third in our tests. Using these 3, we conducted further experiments on a larger Jasper10x4. For larger models, we noticed that batch norm with ReLU outperformed other choices. Thus, leading us to decide on batch normalization and ReLU for our architecture.

During batching, all sequences are padded to match the longest sequence. These padded values caused issues when using layer norm. We applied a sequence mask to exclude padding values from the mean and variance calculation. Further, we computed mean and variance over both the time dimension and channels similar to the sequence-wise normalization proposed by Laurent et al. [20]. In addition to masking layer norm, we additionally applied masking prior to the convolution operation, and masking the mean and variance calculations in batch norm. These results are shown in Table 3. Interestingly, we found that while masking before convolution gives a lower WER, using masks for both convolutions and batch norm results in worse performance.

---

[3] Jasper 5x3 models contain one block of each B1 to B5.

As a final note, we found that training with weight norm was very unstable leading to exploding activations.

Table 2: *Normalization and Activation: Greedy WER, LibriSpeech after 50 epochs*

Model	Normalization	Activation	Dev	
			Clean	Other
Jasper 5x3	Batch Norm	ReLU	8.82	23.26
		cReLU	8.89	23.02
		lReLU	11.31	26.90
		GLU	9.46	24.30
		GAU	9.41	24.65
	Layer Norm (masked)	ReLU	8.82	**22.83**
		cReLU	9.14	23.26
		lReLU	11.29	26.35
		GLU	12.62	29.22
		GAU	**8.35**	23.07
	Weight Norm	ReLU	9.98	24.87
		cReLU	11.25	26.87
		lReLU	11.87	27.54
		GLU	11.05	27.10
		GAU	11.25	27.70
Jasper 10x4	Batch Norm	ReLU	**6.15**	**17.58**
	Layer Norm (Masked)	ReLU	6.56	18.48
		GAU	7.14	19.19

Table 3: *Sequence Masking: Greedy WER, LibriSpeech for Jasper 10x4 after 50 epochs*

Model	Masking	Dev	
		Clean	Other
Jasper DR 10x4	None	5.88	17.62
Jasper DR 10x4	BN Mask	5.92	17.63
Jasper DR 10x4	Conv Mask	**5.66**	**16.77**
Jasper DR 10x4	Conv+BN Mask	5.80	16.97

2.2. Residual Connections

For models deeper than Jasper 5x3, we observe consistently that residual connections are necessary for training to converge. In addition to the simple residual and dense residual model described above, we investigated DenseNet [16] and DenseRNet [17] variants of Jasper. Both connect the outputs of each sub-block to the inputs of following sub-blocks within a block. DenseRNet, similar to Dense Residual, connects the output of each block to the input of all following blocks. DenseNet and DenseRNet combine residual connections using concatenation whereas Residual and Dense Residual use addition. We found that Dense Residual and DenseRNet perform similarly with each performing better on specific subsets of LibriSpeech. We decided to use Dense Residual for subsequent experiments. The main reason is that due to concatenation, the growth factor for DenseNet and DenseRNet requires tuning for deeper models whereas Dense Residual does not have a growth factor.

2.3. Language Model

A language model (LM) is a probability distribution over arbitrary symbol sequences $P(w_1, ..., w_n)$ such that more likely sequences are assigned higher probabilities. LMs are frequently used to condition beam search. During decoding, candidates are evaluated using both acoustic scores and LM scores. Traditional N-gram LMs have been augmented with neural LMs in recent work [21, 22, 23].

Table 4: *Residual Connections: Greedy WER, LibriSpeech for Jasper 10x3 after 400 epochs. All models sized to have roughly the same parameter count.*

Model	#params, M	Dev	
		Clean	Other
Residual	201	4.65	14.36
Dense Residual	211	4.51	**14.15**
DenseNet	205	4.77	14.55
DenseRNet	211	**4.32**	14.21

We experiment with statistical N-gram language models [24] and neural Transformer-XL [12] models. Our best results use acoustic and word-level N-gram language models to generate a candidate list using beam search with a width of 2048. Next, an external Transformer-XL LM rescores the final list. All LMs were trained on datasets independently from acoustic models. We show results with the neural LM in our Results section. We observed a strong correlation between the quality of the neural LM (measured by perplexity) and WER as shown in Figure 3.
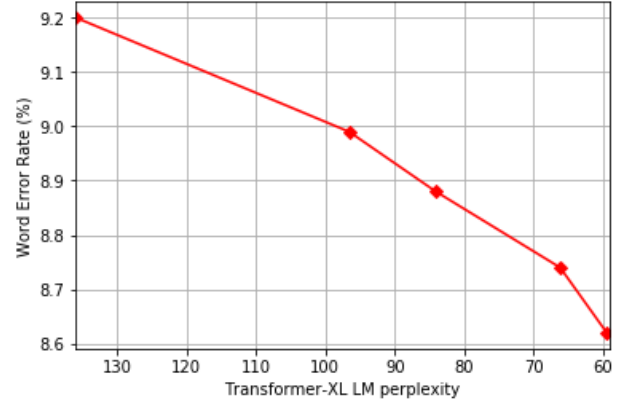

Figure 3: *LM perplexity vs WER. LibriSpeech dev-other. Varying perplexity is achieved by taking earlier or later snapshots during training.*

2.4. NovoGrad

For training, we use either Stochastic Gradient Descent (SGD) with momentum or our own *NovoGrad*, an optimizer similar to Adam [15], except that its second moments are computed per layer instead of per weight. Compared to Adam, it reduces memory consumption and we find it to be more numerically stable.

At each step t, NovoGrad computes the stochastic gradient g_t^l following the regular forward-backward pass. Then the second-order moment v_t^l is computed for each layer l similar to ND-Adam [29]:

$$v_t^l = \beta_2 \cdot v_{t-1}^l + (1 - \beta_2) \cdot ||g_t^l||^2 \qquad (1)$$

The second-order moment v_t^l is used to re-scale gradients g_t^l before calculating the first-order moment m_t^l:

$$m_t^l = \beta_1 \cdot m_{t-1}^l + \frac{g_t^l}{\sqrt{v_t^l + \epsilon}} \qquad (2)$$

Table 5: *LibriSpeech, WER (%)*

Model	E2E	LM	dev-clean	dev-other	test-clean	test-other
CAPIO (single) [23]	N	RNN	3.02	8.28	3.56	8.58
pFSMN-Chain [25]	N	RNN	2.56	7.47	2.97	**7.5**
DeepSpeech2 [26]	Y	5-gram	-	-	5.33	13.25
Deep bLSTM w/ attention [21]	Y	LSTM	3.54	11.52	3.82	12.76
wav2letter++ [27]	Y	ConvLM	3.16	10.05	3.44	11.24
LAS + SpecAugment [4] [28]	Y	RNN	-	-	*2.5*	*5.8*
Jasper DR 10x5	Y	-	3.64	11.89	3.86	11.95
Jasper DR 10x5	Y	6-gram	2.89	9.53	3.34	9.62
Jasper DR 10x5	Y	Transformer-XL	2.68	8.62	**2.95**	8.79
Jasper DR 10x5 + Time/Freq Masks [4]	Y	Transformer-XL	2.62	7.61	2.84	7.84

If L2-regularization is used, a weight decay $d \cdot w_t$ is added to the re-scaled gradient (as in AdamW [30]):

$$m_t^l = \beta_1 \cdot m_{t-1}^l + \frac{g_t^l}{\sqrt{v_t^l + \epsilon}} + d \cdot w_t \tag{3}$$

Finally, new weights are computed using the learning rate α_t:

$$w_{t+1} = w_t - \alpha_t \cdot m_t \tag{4}$$

Using NovoGrad instead of SGD with momentum, we decreased the WER on dev-clean LibriSpeech from 4.00% to 3.64%, a relative improvement of 9% for Jasper DR 10x5. For more details and experiment results with NovoGrad, see [31].

3. Results

We evaluate Jasper across a number of datasets in various domains. In all experiments, we use dropout and weight decay as regularization. At training time, we use 3-fold speed perturbation with fixed +/-10% [32] for LibriSpeech. For WSJ and Hub5'00, we use a random speed perturbation factor between [-10%, 10%] as each utterance is fed into the model. All models have been trained on NVIDIA DGX-1 in mixed precision [33] using OpenSeq2Seq [34]. Pretrained models and training configurations are available from "https://nvidia.github.io/OpenSeq2Seq/html/speech-recognition.html".

3.1. Read Speech

We evaluated the performance of Jasper on two read speech datasets: LibriSpeech and Wall Street Journal (WSJ). For LibriSpeech, we trained Jasper DR 10x5 using our NovoGrad optimizer for 400 epochs. We achieve SOTA performance on the test-clean subset and SOTA among end-to-end speech recognition models on test-other.

We trained a smaller Jasper 10x3 model using the SGD with momentum optimizer for 400 epochs on a combined WSJ dataset (80 hours): LDC93S6A (WSJ0) and LDC94S13A (WSJ1). The results are provided in Table 6.

3.2. Conversational Speech

We also evaluate the Jasper model's performance on a conversational English corpus. The Hub5 Year 2000 (Hub5'00) evaluation (LDC2002S09, LDC2002T43) is widely used in academia.

Table 6: *WSJ End-to-End Models, WER (%)*

Model	LM	nov93	nov92
seq2seq + deep conv [35]	-	-	10.5
wav2letter++ [27]	4-gram	9.5	5.6
wav2letter++ [27]	ConvLM	7.5	4.1
E2E LF-MMI [14]	3-gram	-	4.1
Jasper 10x3	-	16.1	13.3
Jasper 10x3	4-gram	9.9	7.1
Jasper 10x3	Transformer-XL	9.3	6.9

It is divided into two subsets: Switchboard (SWB) and Callhome (CHM). The training data for both the acoustic and language models consisted of the 2000hr Fisher+Switchboard training data (LDC2004S13, LDC2005S13, LDC97S62). Jasper DR 10x5 was trained using SGD with momentum for 50 epochs. We compare to other models trained using the same data and report Hub5'00 results in Table 7.

Table 7: *Hub5'00, WER (%)*

Model	E2E	LM	SWB	CHM
LF-MMI [14]	N	RNN	7.3	14.2
Attention Seq2Seq [36]	Y	-	8.3	15.5
RNN-T [37]	Y	4-gram	8.1	17.5
Char E2E LF-MMI [14]	Y	RNN	8.0	17.6
Phone E2E LF-MMI [14]	Y	RNN	7.5	14.6
CTC + Gram-CTC	Y	N-gram	7.3	14.7
Jasper DR 10x5	Y	4-gram	8.3	19.3
Jasper DR 10x5	Y	Transformer-XL	7.8	16.2

We obtain good results for SWB. However, there is work to be done to improve WER on harder tasks such as CHM.

4. Conclusions

We have presented a new family of neural architectures for end-to-end speech recognition. Inspired by wav2letter's convolutional approach, we build a deep and scalable model, which requires a well-designed residual topology, effective regularization, and a strong optimizer. As our architecture studies demonstrated, a combination of standard components leads to SOTA results on LibriSpeech and competitive results on other benchmarks. Our Jasper architecture is highly efficient for training and inference, and serves as a good baseline approach on top of which to explore more sophisticated regularization, data augmentation, loss functions, language models, and optimization strategies. We are interested to see if our approach can continue to scale to deeper models and larger datasets.

---

[4] We include the latest SOTA which was achieved by Park et al. [28] after our initial submission. We add results for Jasper with time and frequency masks similar to SpecAugment. We use 1 continuous time mask of size $T \sim U(0, 99)$ time steps, and 1 continuous frequency mask of size $F \sim U(0, 26)$ frequency bands.

5. References

[1] A. Waibel, T. Hanazawa, G. Hinton, K. Shirano, and K. Lang, "A time-delay neural network architecture for isolated word recognition," *IEEE Trans. on Acoustics, Speech and Signal Processing*, 1989.

[2] Y. Bengio, R. De Mori, G. Flammia, and R. Kompe, "Global optimization of a neural network-hidden markov model hybrid," *IEEE Transactions on Neural Networks, 3(2), 252259*, 1992.

[3] A. Graves and J. Schmidhuber, "Framewise phoneme classification with bidirectional lstm and other neural network architectures," *Neural Networks, vol. 18*, pp. 602–610, 2005.

[4] G. Hinton *et al.*, "Deep neural networks for acoustic modeling in speech recognition," *IEEE Signal Processing Magazine*, 2012.

[5] A. Graves, S. Fernández, F. Gomez, and J. Schmidhuber, "Connectionist temporal classification: labelling unsegmented sequence data with recurrent neural networks," in *Proceedings of the 23rd international conference on Machine learning*. ACM, 2006, pp. 369–376.

[6] Y. Zhang *et al.*, "Towards end-to-end speech recognition with deep convolutional neural networks," in *Interspeech 2016*, 2016, pp. 410–414.

[7] R. Collobert, C. Puhrsch, and G. Synnaeve, "Wav2letter: an end-to-end convnet-based speech recognition system," *arXiv preprint arXiv:1609.03193*, 2016.

[8] V. Liptchinsky, G. Synnaeve, and R. Collobert, "Letter-based speech recognition with gated convnets," *arXiv preprint arXiv:1712.09444*, 2017.

[9] Y. N. Dauphin, A. Fan, M. Auli, and D. Grangier, "Language modeling with gated convolutional networks," in *Proceedings of the 34th International Conference on Machine Learning - Volume 70*, ser. ICML'17. JMLR.org, 2017, pp. 933–941.

[10] T. Salimans and D. P. Kingma, "Weight normalization: A simple reparameterization to accelerate training of deep neural networks," in *Advances in Neural Information Processing Systems 29*, D. D. Lee, M. Sugiyama, U. V. Luxburg, I. Guyon, and R. Garnett, Eds. Curran Associates, Inc., 2016, pp. 901–909.

[11] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," *CoRR*, vol. abs/1502.03167, 2015.

[12] Z. Dai *et al.*, "Transformer-xl: Language modeling with longer-term dependency," *CoRR*, vol. abs/1901.02860, 2018.

[13] V. Panayotov, G. Chen, D. Povey, and S. Khudanpur, "Librispeech: an asr corpus based on public domain audio books," in *Acoustics, Speech and Signal Processing (ICASSP), 2015 IEEE International Conference on*. IEEE, 2015, pp. 5206–5210.

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

[15] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *CoRR*, vol. abs/1412.6980, 2014.

[16] G. Huang, Z. Liu, L. van der Maaten, and K. Q. Weinberger, "Densely connected convolutional networks," *arXiv preprint arXiv:1608.06993*, 2016.

[17] J. Tang, Y. Song, L. Dai, and I. McLoughlin, "Acoustic modeling with densely connected residual network for multichannel speech recognition," in *Proc. Interspeech 2018*, 2018, pp. 1783–1787.

[18] J. L. Ba, J. R. Kiros, and G. E. Hinton, "Layer normalization," *CoRR*, vol. abs/1607.06450, 2016.

[19] A. van den Oord *et al.*, "Conditional image generation with pixelcnn decoders," in *Advances in Neural Information Processing Systems 29*, D. D. Lee, M. Sugiyama, U. V. Luxburg, I. Guyon, and R. Garnett, Eds. Curran Associates, Inc., 2016, pp. 4790–4798.

[20] C. Laurent, G. Pereyra, P. Brakel, Y. Zhang, and Y. Bengio, "Batch normalized recurrent neural networks," in *2016 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, March 2016, pp. 2657–2661.

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

[22] D. Povey *et al.*, "Semi-orthogonal low-rank matrix factorization for deep neural networks," in *Interspeech*, 2018.

[23] K. J. Han, A. Chandrashekaran, J. Kim, and I. R. Lane, "The CAPIO 2017 conversational speech recognition system," *CoRR*, vol. abs/1801.00059, 2018.

[24] K. Heafield, "Kenlm: Faster and smaller language model queries," in *Proceedings of the sixth workshop on statistical machine translation*. Association for Computational Linguistics, 2011, pp. 187–197.

[25] X. Yang, J. Li, and X. Zhou, "A novel pyramidal-fsmn architecture with lattice-free MMI for speech recognition," *CoRR*, vol. abs/1810.11352, 2018.

[26] D. Amodei *et al.*, "Deep speech 2: End-to-end speech recognition in english and mandarin," in *Proceedings of the 33rd International Conference on International Conference on Machine Learning - Volume 48*, ser. ICML'16. JMLR.org, 2016, pp. 173–182.

[27] N. Zeghidour *et al.*, "Fully convolutional speech recognition," *CoRR*, vol. abs/1812.06864, 2018.

[28] D. S. Park *et al.*, "SpecAugment: A Simple Data Augmentation Method for Automatic Speech Recognition," *arXiv e-prints*, 2019.

[29] Z. Zhang, L. Ma, Z. Li, and C. Wu, "Normalized direction-preserving adam," *arXiv e-prints arXiv:1709.04546*, 2017.

[30] I. Loshchilov and F. Hutter, "Decoupled weight decay regularization," in *International Conference on Learning Representations*, 2019.

[31] B. Ginsburg *et al.*, "Stochastic Gradient Methods with Layer-wise Adaptive Moments for Training of Deep Networks," *arXiv e-prints*, 2019.

[32] K. Tom, P. Vijayaditya, P. Daniel, and K. Sanjeev, "Audio augmentation for speech recognition," *Interspeech 2015*, 2015.

[33] P. Micikevicius *et al.*, "Mixed precision training," *arXiv preprint arXiv:1710.03740*, 2017.

[34] O. Kuchaiev *et al.*, "Openseq2seq: extensible toolkit for distributed and mixed precision training of sequence-to-sequence models," , 2018.

[35] Y. Zhang, W. Chan, and N. Jaitly, "Very deep convolutional networks for end-to-end speech recognition," in *Acoustics, Speech and Signal Processing (ICASSP), 2017 IEEE International Conference on*. IEEE, 2017.

[36] C. Weng *et al.*, "Improving attention based sequence-to-sequence models for end-to-end english conversational speech recognition," in *Proc. Interspeech 2018*, 2018, pp. 761–765.

[37] E. Battenberg *et al.*, "Exploring neural transducers for end-to-end speech recognition," in *2017 IEEE Automatic Speech Recognition and Understanding Workshop (ASRU)*, Dec 2017, pp. 206–213.