LISTEN, ATTEND AND SPELL: A NEURAL NETWORK FOR LARGE VOCABULARY CONVERSATIONAL SPEECH RECOGNITION

William Chan

Carnegie Mellon University

Navdeep Jaitly, Quoc Le, Oriol Vinyals

Google Brain

ABSTRACT

We present Listen, Attend and Spell (LAS), a neural speech recognizer that transcribes speech utterances directly to characters without pronunciation models, HMMs or other components of traditional speech recognizers. In LAS, the neural network architecture subsumes the acoustic, pronunciation and language models making it not only an *end-to-end trained* system but an *end-to-end model*. In contrast to DNN-HMM, CTC and most other models, LAS makes no independence assumptions about the probability distribution of the output character sequences given the acoustic sequence. Our system has two components: a listener and a speller. The listener is a pyramidal recurrent network encoder that accepts filter bank spectra as inputs. The speller is an attention-based recurrent network decoder that emits each character conditioned on all previous characters, and the entire acoustic sequence. On a Google voice search task, LAS achieves a WER of 14.1% without a dictionary or an external language model and 10.3% with language model rescoring over the top 32 beams. In comparison, the state-of-the-art CLDNN-HMM model achieves a WER of 8.0% on the same set.

Index Terms— Recurrent neural network, neural attention, end-to-end speech recognition

1. INTRODUCTION

State-of-the-art speech recognizers of today are complicated systems comprising of various components - acoustic models, language models, pronunciation models and text normalization. Each of these components make assumptions about the underlying probability distributions they model. For example n-gram language models and Hidden Markov Models (HMMs) make strong Markovian independence assumptions between words/symbols in a sequence. Connectionist Temporal Classification (CTC) and DNN-HMM systems assume that neural networks make independent predictions at different times and use HMMs or language models (which make their own independence assumptions) to introduce dependencies between these predictions over time [1, 2, 3]. *End-to-end* training of such models attempts to mitigate these problems by training the components jointly [4, 5, 6]. In these models, acoustic models are updated based on a WER proxy, while the pronunciation and language models are rarely updated [7], if at all.

In this paper we introduce Listen, Attend and Spell (LAS), a neural network that learns to transcribe an audio sequence signal to a word sequence, one character at a time, without using explicit language models, pronunciation models, HMMs, etc. LAS does not make any independence assumptions about the nature of the probability distribution of the output character sequence, given the input acoustic sequence. This method is based on the sequence-to-sequence learning framework with attention [8, 9, 10, 11, 12, 13]. It consists of an encoder Recurrent Neural Network (RNN), which is named the *listener*, and a decoder RNN, which is named the *speller*. The listener is a pyramidal RNN that converts speech signals into high level features. The speller is an RNN that transduces these higher level features into output utterances by specifying a probability distribution over the next character, given all of the acoustics and the previous characters. At each step the RNN uses its internal state to guide an attention mechanism [10, 11, 12] to compute a "context" vector from the high level features of the listener. It uses this context vector, and its internal state to both update its internal state and to predict the next character in the sequence. The entire model is trained jointly, from scratch, by optimizing the probability of the output sequence using a chain rule decomposition. We call this an *end-to-end model* because all the components of a traditional speech recognizer are integrated into its parameters, and optimized together during training, unlike *end-to-end training* of conventional models that attempt to adjust acoustic models to work well with the other fixed components of a speech recognizer.

Our model was inspired by [11, 12] that showed how end-to-end recognition could be performed on the TIMIT phone recognition task. We note a recent paper from the same group that describes an application of these ideas to WSJ [14]. Our paper independently explores the challenges associated with the application of these ideas to large scale conversational speech recognition on a Google voice search task. We defer a discussion of the relationship between these and other methods to section 5.

2. MODEL

In this section, we formally describe LAS. Let $\mathbf{x} = (x_1, \dots, x_T)$ be the input sequence of filter bank spectra features and $\mathbf{y} = (\langle \text{sos} \rangle, y_1, \dots, y_S, \langle \text{eos} \rangle)$, $y_i \in \{a, \cdots, z, 0, \cdots, 9, \langle \text{space} \rangle, \langle \text{comma} \rangle, \langle \text{period} \rangle, \langle \text{apostrophe} \rangle, \langle \text{unk} \rangle\}$ be the output sequence of characters. Here $\langle \text{sos} \rangle$ and $\langle \text{eos} \rangle$ are the special start-of-sentence token, and end-of-sentence tokens, respectively, and $\langle \text{unk} \rangle$ are unknown tokens such as accented characters.

LAS models each character output y_i as a conditional distribution over the previous characters $y_{<i}$ and the input signal $\mathbf{x}$ using the chain rule for probabilities:

$$P(\mathbf{y}|\mathbf{x}) = \prod_i P(y_i|\mathbf{x}, y_{<i}) \tag{1}$$

This objective makes the model a discriminative, end-to-end model, because it directly predicts the conditional probability of character sequences, given the acoustic signal.

LAS consists of two sub-modules: the listener and the speller. The listener is an acoustic model encoder that performs an operation called Listen. The Listen operation transforms the original signal $\mathbf{x}$ into a high level representation $\mathbf{h} = (h_1, \dots, h_U)$ with $U \leq T$. The speller is an attention-based character decoder that performs an operation we call AttendAndSpell. The AttendAndSpell operation

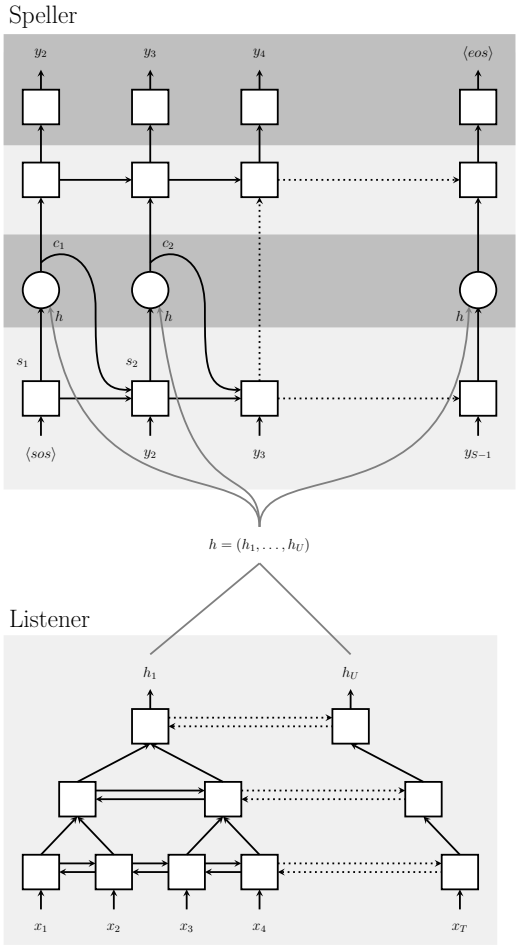

Fig. 1: Listen, Attend and Spell (LAS) model: the listener is a pyramidal BLSTM encoding our input sequence $\mathbf{x}$ into high level features $\mathbf{h}$, the speller is an attention-based decoder generating the $\mathbf{y}$ characters from $\mathbf{h}$.

consumes $\mathbf{h}$ and produces a probability distribution over character sequences:

$$\mathbf{h} = \text{Listen}(\mathbf{x}) \tag{2}$$

$$P(y_i|\mathbf{x}, y_{<i}) = \text{AttendAndSpell}(y_{<i}, \mathbf{h}) \tag{3}$$

Figure 1 depicts these two components. We provide more details of these components in the following sections.

2.1. Listen

The Listen operation uses a Bidirectional Long Short Term Memory RNN (BLSTM) [15, 16, 2] with a pyramidal structure. This modification is required to reduce the length U of $\mathbf{h}$, from T, the length of the input $\mathbf{x}$, because the input speech signals can be hundreds to thousands of frames long. A direct application of BLSTM for the operation Listen converged slowly and produced results inferior to those reported here, even after a month of training time. This is presumably because the operation AttendAndSpell has a hard time extracting the relevant information from a large number of input time steps.

We circumvent this problem by using a pyramidal BLSTM (pBLSTM). In each successive stacked pBLSTM layer, we reduce the time resolution by a factor of 2. In a typical deep BLSTM architecture, the output at the i-th time step, from the j-th layer is computed as follows:

$$h_i^j = \text{BLSTM}(h_{i-1}^j, h_i^{j-1}) \tag{4}$$

In the pBLSTM model, we concatenate the outputs at consecutive steps of each layer before feeding it to the next layer, i.e.:

$$h_i^j = \text{pBLSTM}(h_{i-1}^j, \left[h_{2i}^{j-1}, h_{2i+1}^{j-1}\right]) \tag{5}$$

In our model, we stack 3 pBLSTMs on top of the bottom BLSTM layer to reduce the time resolution $2^3 = 8$ times. This allows the attention model (described in the next section) to extract the relevant information from a smaller number of times steps. In addition to reducing the resolution, the deep architecture allows the model to learn nonlinear feature representations of the data. See Figure 1 for a visualization of the pBLSTM.

The pyramidal structure also reduces the computational complexity. The attention mechanism in the speller U has a computational complexity of $O(US)$. Thus, reducing U speeds up learning and inference significantly. Other neural network architectures have been described in literature with similar motivations, including the hierarchical RNN [17], clockwork RNN [18] and CNN [19].

2.2. Attend and Spell

The AttendAndSpell function is computed using an attention-based LSTM transducer [10, 12]. At every output step, the transducer produces a probability distribution over the next character conditioned on all the characters seen previously. The distribution for y_i is a function of the decoder state s_i and context c_i. The decoder state s_i is a function of the previous state s_{i-1}, the previously emitted character y_{i-1} and context c_{i-1}. The context vector c_i is produced by an attention mechanism. Specifically,

$$c_i = \text{AttentionContext}(s_i, \mathbf{h}) \tag{6}$$

$$s_i = \text{RNN}(s_{i-1}, y_{i-1}, c_{i-1}) \tag{7}$$

$$P(y_i|\mathbf{x}, y_{<i}) = \text{CharacterDistribution}(s_i, c_i) \tag{8}$$

where $\text{CharacterDistribution}$ is an MLP with softmax outputs over characters, and where RNN is a 2 layer LSTM.

At each time step, i, the attention mechanism, AttentionContext generates a context vector, c_i encapsulating the information in the acoustic signal needed to generate the next character. The attention model is content based - the contents of the decoder state s_i are matched to the contents of h_u representing time step u of $\mathbf{h}$, to generate an attention vector α_i. The vectors h_u are linearly blended using α_i to create c_i.

Specifically, at each decoder timestep i, the AttentionContext function computes the scalar energy $e_{i,u}$ for each time step u, using vector $h_u \in \mathbf{h}$ and s_i. The scalar energy $e_{i,u}$ is converted into a probability distribution over times steps (or attention) α_i using a softmax function. The softmax probabilities are used as mixing weights for blending the listener features h_u to the context vector c_i for output time step i:

$$e_{i,u} = \langle \phi(s_i), \psi(h_u) \rangle \tag{9}$$

$$\alpha_{i,u} = \frac{\exp(e_{i,u})}{\sum_{u'} \exp(e_{i,u'})} \tag{10}$$

$$c_i = \sum_u \alpha_{i,u} h_u \tag{11}$$

where ϕ and ψ are MLP networks. After training, the α_i distribution is typically very sharp and focuses on only a few frames of $\mathbf{h}$; c_i can be seen as a continuous bag of weighted features of $\mathbf{h}$. Figure 1 shows the LAS architecture.

2.3. Learning

We train the parameters of our model to maximize the log probability of the correct sequences. Specifically,

$$\tilde{\theta} = \max_{\theta} \sum_{i} \log P(y_i|\mathbf{x}, \tilde{y}_{<i}; \theta) \tag{12}$$

where $\tilde{y}_{i-1}$ is the ground truth previous character or a character randomly sampled (with 10% probability) from the model, i.e. $\mathrm{CharacterDistribution}(s_{i-1}, c_{i-1})$ using the procedure from [20].

2.4. Decoding and Rescoring

During inference we want to find the most likely character sequence given the input acoustics:

$$\hat{\mathbf{y}} = \arg\max_{\mathbf{y}} \log P(\mathbf{y}|\mathbf{x}) \tag{13}$$

We use a simple left-to-right beam search similar to [8]. We can also apply language models trained on large external text corpora alone, similar to conventional speech systems [21]. We simply rescore our beams with the language model. We find that our model has a small bias for shorter utterances so we normalize our probabilities by the number of characters $|\mathbf{y}|_c$ in the hypothesis and combine it with a language model probability $P_{\mathrm{LM}}(\mathbf{y})$:

$$s(\mathbf{y}|\mathbf{x}) = \frac{\log P(\mathbf{y}|\mathbf{x})}{|\mathbf{y}|_c} + \lambda \log P_{\mathrm{LM}}(\mathbf{y}) \tag{14}$$

where λ is our language model weight and can be determined by a held-out validation set.

3. EXPERIMENTS

We used a dataset with three million Google Voice Search utterances (representing 2000 hours of data) for our experiments. Approximately 10 hours of utterances were randomly selected as a held-out validation set. Data augmentation was performed using a room simulator, adding different types of noise and reverberations; the noise sources were obtained from YouTube and environmental recordings of daily events [22]. This increased the amount of audio data by 20 times with a SNR between 5dB and 30dB [22]. We used 40-dimensional log-mel filter bank features computed every 10ms as the acoustic inputs to the listener. A separate set of 22K utterances representing approximately 16 hours of data were used as the test data. A noisy test set was also created using the same corruption strategy that was applied to the training data. All training sets are anonymized and hand-transcribed, and are representative of Google's speech traffic.

The text was normalized by converting all characters to lower case English alphanumerics (including digits). The punctuations: space, comma, period and apostrophe were kept, while all other tokens were converted to the unknown $\langle \mathrm{unk} \rangle$ token. As mentioned earlier, all utterances were padded with the start-of-sentence $\langle \mathrm{sos} \rangle$ and the end-of-sentence $\langle \mathrm{eos} \rangle$ tokens.

The state-of-the-art model on this dataset is a CLDNN-HMM system that was described in [22]. The CLDNN system achieves a WER of 8.0% on the clean test set and 8.9% on the noisy test set. However, we note that the CLDNN uses unidirectional LSTMs and would certainly benefit from the use of a BLSTM architecture. Additionally, the LAS model does not use convolutional filters which have been reported to yield 5-7% WER relative improvement [22].

Table 1: WER comparison on the clean and noisy Google voice search task. The CLDNN-HMM system is the state-of-the-art, the Listen, Attend and Spell (LAS) models are decoded with a beam size of 32. Language Model (LM) rescoring can be beneficial.

Model	Clean WER	Noisy WER
CLDNN-HMM [22]	8.0	8.9
LAS	14.1	16.5
LAS + LM Rescoring	10.3	12.0

For the Listen function we used 3 layers of 512 pBLSTM nodes (i.e., 256 nodes per direction) on top of a BLSTM that operates on the input. This reduced the time resolution by $8 = 2^3$ times. The Spell function used a two layer LSTM with 512 nodes each. The weights were initialized with a uniform distribution $\mathcal{U}(-0.1, 0.1)$. Asynchronous Stochastic Gradient Descent (ASGD) was used for training our model [23]. A learning rate of 0.2 was used with a geometric decay of 0.98 per 3M utterances (i.e., $1/20$-th of an epoch). We used the DistBelief framework [23] with 32 replicas, each with a minibatch of 32 utterances. In order to further speed up training, the sequences were grouped into buckets based on their frame length [8]. The model was trained until the results on the validation set stopped improving, taking approximately two weeks. The model was decoded using N-best list decoding with beam size of $N = 32$.

4. RESULTS AND DISCUSSION

We achieved 14.1% WER on the clean test set and 16.5% WER on the noisy test set without any dictionary or language model. We found that constraining the beam search with a dictionary had no impact on the WER. Rescoring the top 32 beams with the same n-gram language model that was used by the CLDNN system using a language model weight of $\lambda = 0.008$ improved the results for the clean and noisy test sets to 10.3% and 12.0% respectively. Note that for convenience, we did not decode with a language model, but rather only rescored the top 32 beams. It is possible that further gains could have been achieved by using the language model during decoding. Table 1 summarizes the WER results.

The content-based attention mechanism creates an explicit alignment between the characters and audio signal. We can visualize the attention mechanism by recording the attention distribution on the acoustic sequence at every character output timestep. Figure 2 visualizes the attention alignment between the characters and the filterbanks for the utterance "how much would a woodchuck chuck". For this particular utterance, the model learnt a monotonic distribution without any location priors. The words "woodchuck" and "chuck" have acoustic similarities, the attention mechanism was slightly confused when emitting "woodchuck" with a dilution in the distribution. The attention model was also able to identify the start and end of the utterance properly.

We observed that LAS can learn multiple spelling variants given the same acoustics. Table 2 shows top beams for the utterance that includes "triple a". As can be seen, the model produces both "triple a" and "aaa" within the top four beams. The decoder is able to generate such varied parses, because the next step prediction model makes no assumptions on the probability distribution by using the chain rule

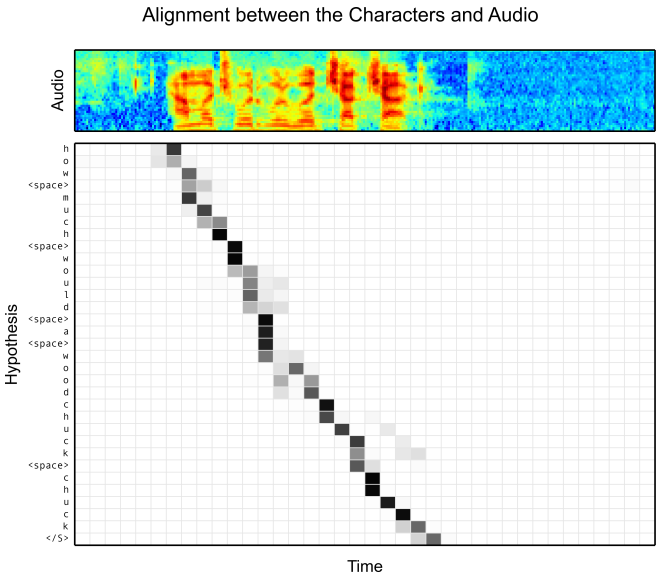

Fig. 2: Alignments between character outputs and audio signal produced by the Listen, Attend and Spell (LAS) model for the utterance "how much would a woodchuck chuck". The content based attention mechanism was able to identify the start position in the audio sequence for the first character correctly. The alignment produced is generally monotonic without a need for any location based priors.

Table 2: Example 1: "triple a" vs. "aaa" spelling variants.

Beam	**Text**	log P	**WER**
Truth	call aaa roadside assistance	-	-
1	call aaa roadside assistance	-0.57	0.00
2	call triple a roadside assistance	-1.54	50.00
3	call trip way roadside assistance	-3.50	50.00
4	call xxx roadside assistance	-4.44	25.00

decomposition. It would be difficult to produce such differing transcripts using CTC due to the conditional independence assumptions, where the distribution of the output y_i at time i is conditionally independent of distribution y_{i+1} at time $i+1$. Conventional DNN-HMM systems would require both spellings to be in the pronunciation dictionary to generate both transcriptions.

5. RELATED WORK

There has recently been an explosion in methods for end-to-end trained speech models because of their inherent simplicity compared to current speech recognition systems [2, 24, 6, 25, 14]. However these methods have inherent shortcomings that our model attempts to address. Here we describe in more detail the relationship between our work and prior approaches.

Initially, [2] showed that CTC could perform end-to-end speech recognition on WSJ, going straight from audio to character sequences. [24, 25] subsequently showed strong results with CTC on larger datasets and Switchboard. However it was noted in [24, 2] that good accuracy could only be achieved through the use of a strong language model during beam search decoding; the language models use are themselves fixed and trained independently of the CTC objective.

CTC has also been applied to *end-to-end training* with phoneme targets and n-gram language models using FSTs in [26, 27, 6]. However, unlike the methods above, these methods use pronunciation dictionaries and language models within FSTs. End-to-end training here implies training of the acoustic models with fixed dictionaries and language models, instead of training models that recognize character sequences directly. In this respect these models are end-to-end trained systems, rather than end-to-end models.

While CTC has shown tremendous promise in end-to-end speech recognition, it is limited by the assumptions of independence between frames - the output at one frame has no influence at the outputs at the other frames - much like the unary potential of Conditional Random Fields. The only way to ameliorate this problem is through the use of a strong language model [2].

The model proposed here is based on the sequence-to-sequence architecture [8, 10] and does not suffer from the above shortcoming. LAS models the output sequence given the input sequence using the chain rule decomposition, starting at the first character. As such this model makes no assumptions about the probability distribution and is only limited by the capacity of the recurrent neural network in modeling such a complicated distribution. Further, this single model encompasses all aspects of a speech recognition system - the acoustic, pronunciation and language models are all encoded within its parameters. We argue that this makes it not only an *end-to-end trained* system, but an *end-to-end model*. This makes it a very powerful model for end-to-end speech recognition. Future work is likely to explore how to use increasingly more complicated models for improved performance over what was achieved in this paper. Further, these models are likely to benefit from even larger datasets since the decoder is able to overfit the small number of transcripts[1].

The model described in [14] is the closest to our model, with some slight differences. We use a pyramidal encoder while they use an encoder in which the higher layers subsample the hidden states of the layers below. In addition they use an FST to incorporate a language model, while we use language model rescoring and a length-dependent language model blending (see section 2.4). We note that these two works were performed concurrently and independently.

6. CONCLUSIONS

We have presented Listen, Attend and Spell (LAS), a neural speech recognizer that can transcribe acoustic signals to characters directly without using any of the traditional components of a speech recognition system, such as HMMs, language models and pronunciation dictionaries. We submit that it is not only an *end-to-end trained* system, but an *end-to-end model*. LAS accomplishes this goal by making no conditional independence assumptions about the output sequence using the sequence-to-sequence framework. This distinguishes it from models like CTC, DNN-HMM and other models that can be *trained end-to-end* but make various conditional independence assumptions to accomplish this. We showed how this model learns an implicit language model that can generate multiple spelling variants given the same acoustics. We also showed how an external language model, trained on additional text, can be used to re-rank the top hypotheses. We demonstrated that such an end-to-end model can be trained and be competitive with state-of-the-art CLDNN-HMM systems. We are optimistic that this approach will pave the way to new neural speech recognizers that are simpler to train and achieve even better accuracies than the best current speech recognition systems.

[1] We note that we used three million utterances for training but that is a very small corpus for an RNN language model

7. REFERENCES

[1] A. Graves, A. Mohamed, and G. Hinton, "Speech Recognition with Deep Recurrent Neural Networks," in *IEEE International Conference on Acoustics, Speech and Signal Processing*, 2013.

[2] A. Graves and N. Jaitly, "Towards End-to-End Speech Recognition with Recurrent Neural Networks," in *International Conference on Machine Learning*, 2014.

[3] G. Hinton, L. Deng, D. Yu, G. Dahl, A. Mohamed, N. Jaitly, A. Senior, V. Vanhoucke, P. Nguyen, T. Sainath, and B. Kingsbury, "Deep Neural Networks for Acoustic Modeling in Speech Recognition: The Shared Views of Four Research Groups," *IEEE Signal Processing Magazine*, Nov. 2012.

[4] K. Vesely, A. Ghoshal, L. Burget, and D. Povey, "Sequence-discriminative training of deep neural networks," in *INTERSPEECH*, 2013.

[5] H. Sak, O. Vinyals, G. Heigold, A. Senior, E. McDermott, R. Monga, and M. Mao, "Sequence Discriminative Distributed Training of Long Short-Term Memory Recurrent Neural Networks," in *INTERSPEECH*, 2014.

[6] Y. Miao, M. Gowayyed, and F. Metze, "EESEN: End-to-End Speech Recognition using Deep RNN Models and WFST-based Decoding," in *Http://arxiv.org/abs/1507.08240*, 2015.

[7] Y. Kubo, T. Hori, and A. Nakamura, "Integrating Deep Neural Networks into Structured Classification Approach based on Weighted Finite-State Transducers," in *INTERSPEECH*, 2012.

[8] I. Sutskever, O. Vinyals, and Q. Le, "Sequence to Sequence Learning with Neural Networks," in *Neural Information Processing Systems*, 2014.

[9] K. Cho, B. van Merrienboer, C. Gulcehre, D. Bahdanau, F. Bougares, H. Schwen, and Y. Bengio, "Learning Phrase Representations using RNN Encoder-Decoder for Statistical Machine Translation," in *Conference on Empirical Methods in Natural Language Processing*, 2014.

[10] D. Bahdanau, K. Cho, and Y. Bengio, "Neural Machine Translation by Jointly Learning to Align and Translate," in *International Conference on Learning Representations*, 2015.

[11] J. Chorowski, D. Bahdanau, K. Cho, and Y. Bengio, "End-to-end Continuous Speech Recognition using Attention-based Recurrent NN: First Results," in *Neural Information Processing Systems: Workshop Deep Learning and Representation Learning Workshop*, 2014.

[12] J. Chorowski, D. Bahdanau, D. Serdyuk, K. Cho, and Y. Bengio, "Attention-Based Models for Speech Recognition," in *Neural Information Processing Systems*, 2015.

[13] K. Xu, J. Ba, R. Kiros, K. Cho, A. Courville, R. Salakhutdinov, R. Zemel, and Y. Bengio, "Show, Attend and Tell: Neural Image Caption Generation with Visual Attention," in *International Conference on Machine Learning*, 2015.

[14] D. Bahdanau, J. Chorowski, D. Serdyuk, P. Brakel, and Y. Bengio, "End-to-end attention-based large vocabulary speech recognition," in *Http://arxiv.org/abs/1508.04395*, 2015.

[15] S. Hochreiter and J. Schmidhuber, "Long Short-Term Memory," *Neural Computation*, vol. 9, no. 8, pp. 1735–1780, Nov. 1997.

[16] A. Graves, N. Jaitly, and A. Mohamed, "Hybrid Speech Recognition with Bidirectional LSTM," in *Automatic Speech Recognition and Understanding Workshop*, 2013.

[17] S. Hihi and Y. Bengio, "Hierarchical Recurrent Neural Networks for Long-Term Dependencies," in *Neural Information Processing Systems*, 1996.

[18] J. Koutnik, K. Greff, F. Gomez, and J. Schmidhuber, "A Clockwork RNN," in *International Conference on Machine Learning*, 2014.

[19] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proceedings of the IEEE*, vol. 86, pp. 2278–2324, 11 Nov. 1998.

[20] S. Bengio, O. Vinyals, N. Jaitly, and N. Shazeer, "Scheduled Sampling for Sequence Prediction with Recurrent Neural Networks," in *Neural Information Processing Systems*, 2015.

[21] D. Povey, A. Ghoshal, G. Boulianne, L. Burget, O. Glembek, N. Goel, M. Hannenmann, P. Motlicek, Y. Qian, P. Schwarz, J. Silovsky, G. Stemmer, and K. Vesely, "The Kaldi Speech Recognition Toolkit," in *Automatic Speech Recognition and Understanding Workshop*, 2011.

[22] T. N. Sainath, O. Vinyals, A. Senior, and H. Sak, "Convolutional, Long Short-Term Memory, Fully Connected Deep Neural Networks," in *IEEE International Conference on Acoustics, Speech and Signal Processing*, 2015.

[23] J. Dean, G. S. Corrado, R. Monga, K. Chen, M. Devin, Q. V. Le, M. Z. Mao, M. Ranzato, A. Senior, P. Tucker, K. Yang, and A. Y. Ng, "Large Scale Distributed Deep Networks," in *Neural Information Processing Systems*, 2012.

[24] A. Hannun, C. Case, J. Casper, B. Catanzaro, G. Diamos, E. Elsen, R. Prenger, S. Satheesh, S. Sengupta, A. Coates, and A. Ng, "Deep Speech: Scaling up end-to-end speech recognition," in *Http://arxiv.org/abs/1412.5567*, 2014.

[25] A. Maas, Z. Xie, D. Jurafsky, and A. Ng, "Lexicon-free conversational speech recognition with neural networks," in *North American Chapter of the Association for Computational Linguistics*, 2015.

[26] H. Sak, A. Senior, K. Rao, O. Irsoy, A. Graves, F. Beaufays, and J. Schalkwyk, "Learning acoustic frame labeling for speech recognition with recurrent neural networks," in *IEEE International Conference on Acoustics, Speech, and Signal Processing*, 2015.

[27] H. Sak, A. Senior, K. Rao, and F. Beaufays, "Fast and Accurate Recurrent Neural Network Acoustic Models for Speech Recognition," in *INTERSPEECH*, 2015.