Interspeech 2018
2-6 September 2018, Hyderabad

Forward-Backward Attention Decoder

Masato Mimura, Shinsuke Sakai, Tatsuya Kawahara

Kyoto University, School of Informatics,
Sakyo-ku, Kyoto 606-8501, Japan
{mimura, sakai, kawahara}@sap.ist.i.kyoto-u.ac.jp

Abstract

This paper investigates how forward and backward attentions can be integrated to improve the performance of attention-based sequence-to-sequence (seq2seq) speech recognition systems. In the proposed approach, speech is decoded from left to right as well as from right to left utilizing forward and backward attention vectors, and the best sentence hypothesis is searched for according to combined probabilities provided by the decoders of two directions. Our method takes advantage of two distinct and complementary ways of extracting information from the asymmetric time structure of speech. It also mitigates a drawback of attention-based models that they tend to output less reliable labels due to error accumulation when the utterance becomes longer. We also show the effectiveness of a multitask learning in which the forward decoder is jointly trained with backward decoding sharing a single encoder. The proposed forward-backward decoding improved word error rates (WERs) of word-level attention models by up to 12.7 % relative in speech recognition experiments using large-scale spontaneous speech corpora. They achieve much higher performances than a state-of-the-art hybrid DNN-HMM system while retaining the advantage of very low latency.

Index Terms: Sequence-to-sequence speech recognition, attention, acoustic-to-word models, forward-backward decoding, multitask learning

1. Introduction

Deep learning-based hybrid acoustic models have drastically improved the performance of automatic speech recognition (ASR) [1]. It was recently reported that even a human-level recognition performance can be achievable when the hybrid models are coupled with bidirectional LSTMs and very deep convolutional networks with residual connections [2][3]. However, in exchange for these excellent performances, these ASR systems have very complicated structures consisting of complex decoders, large language models, and carefully designed pronunciation dictionaries. They have a large runtime latency and less portability.

On the other hand, we have seen the rapid development of an alternative sequence-to-sequence (seq2seq) approaches to speech recognition using connectionist temporal classification (CTC) [4][5][6][7], attention-based encoder-decoder models [8][9][10][11] or RNN-transducers [12][13]. Their remarkable advantage is that they get rid of dependency on frame-level probabilistic state transition models such as HMMs. An extreme example of the seq2seq approach is acoustic-to-word models [14][15][16] which directly map acoustic signals into word sequences. They do not require any external decoders or language models, leading to an extremely simplified architecture of ASR systems and very low latency. Outputting words rather than phones or characters is also an essential requirement for subsequent natural language processing such as dialogue, translation and information query.

We have shown in [16] that attention-based models which explicitly incorporate label transition probabilities are significantly better than CTC-based models for word-level seq2seq speech recognition. In this paper, we further seek to improve the attention-based speech recognition system. Specifically, we look at attending the encoded feature stream from right to left as well as left to right, and investigate how this backward attention can be utilized together with the usual forward attention to improve the performance of attention-based models. In the proposed approach, speech is decoded from right to left as well as from left to right utilizing forward and backward attention vectors, and the best sentence hypothesis is searched for based on combined probabilities provided by the decoders for two directions. We also propose a multitask learning in which the forward decoder is trained with a subtask of backward decoding sharing a single encoder and vice versa, aiming at a regularization effect of encoder optimization for both directions. We demonstrate that acoustic-to-word attention-based models enhanced with the proposed approach achieve much higher performances than a state-of-the-art hybrid DNN-HMM system with a decoding speed faster by a factor of 50 through speech recognition experiments using large-scale spontaneous Japanese corpora.

2. Attention-based speech recognition

This section presents a brief review on attention-based seq2seq speech recognition, including a decoding algorithm based on beam search. In attention-based speech recognition, we model seq2seq mapping between speech and label sequences using an encoder-decoder architecture [8][9]. This architecture has two distinct sub-networks. One is the encoder which transforms an acoustic feature sequence to a sequential representation of length T. Based on this encoded acoustic information, the other decoder sub-network predicts a label sequence whose length L is usually shorter than the input length T. The decoder uses only a relevant portion of the encoded sequential representation for predicting a label at each time step using the attention mechanism. The encoder is implemented as multi-layer bidirectional RNN such as LSTM [17], and the decoder usually consists of a single layer of unidirectional LSTM followed by a softmax output layer.

The attention-based models are formulated as follows. The encoder transforms input acoustic features $\boldsymbol{X} = (\boldsymbol{x}_1, ..., \boldsymbol{x}_T)$ to a sequential representation $\boldsymbol{H} = (\boldsymbol{h}_1, ..., \boldsymbol{h}_T)$ that summarizes the characteristics of the input. In the following decoding step, the hidden state activation of the RNN-based decoder at the l-th time step is computed as:

$$\boldsymbol{r}_l = Recurrency\left(\boldsymbol{r}_{l-1}, \boldsymbol{g}_l, \boldsymbol{y}_{l-1}\right), \tag{1}$$

Algorithm 1 ForwardBeamSearch($N, \boldsymbol{X}$)

```
1:  $F$ : set of completed label sequences
2:  $NewSeqs$ : set of label sequences at current output time
3:  $Seqs$ : set of label sequences up to the last output time
4:  $F \Leftarrow \{\emptyset\}$, $score(\langle \text{sos} \rangle) = 0$, $Seqs \Leftarrow \{(\langle \text{sos} \rangle)\}$
5:  $w \Leftarrow N$
6:  while $w > 0$ do
7:      $NewSeqs \Leftarrow \{\emptyset\}$
8:      for sequence $\boldsymbol{s} \in Seqs$ do
9:          $B \Leftarrow$ The $w$ best words in terms of $p(y|\boldsymbol{s}, \boldsymbol{X})$
10:         for word $y \in B$ do
11:             $\boldsymbol{s}^+ \Leftarrow concat(\boldsymbol{s}, y)$
12:             $score(\boldsymbol{s}^+) = score(\boldsymbol{s}) + \log(p(y|\boldsymbol{s}, \boldsymbol{X}))$
13:             if $y = \langle \text{eos} \rangle$ then
14:                 add $\boldsymbol{s}^+$ to $F$
15:                 $w \Leftarrow w - 1$
16:             else
17:                 add $\boldsymbol{s}^+$ to $NewSeqs$
18:             end if
19:         end for
20:     end for
21:     $Seqs \Leftarrow$ The $w$ best sequences in $NewSeqs$ in terms of
        $score(\boldsymbol{s}^+)$
22: end while
23: return set of sentence candidates $F$
```

where g_l and y_{l-1} denote the "glimpse" at the l-th time step and the predicted label at the previous step, respectively. The glimpse g_l is a weighted sum of the encoder output sequence as:

$$\boldsymbol{g}_l = \sum_t \alpha_{l,t} \boldsymbol{h}_t, \tag{2}$$

where $\alpha_{l,t}$ is an attention weight of $\boldsymbol{h}_t$. It is calculated as:

$$e_{l,t} = Score(\boldsymbol{r}_{t-1}, \boldsymbol{h}_t, \boldsymbol{\alpha}_{l-1}), \tag{3}$$

$$\alpha_{l,t} = \exp(e_{l,t}) / \sum_{t'=1}^{T} \exp(e_{l,t'}). \tag{4}$$

There are many choices for implementation of the score function (4). In this paper, we adopt the hybrid location and content-based attention mechanism [9] as follows:

$$e_{l,t} = \boldsymbol{w}^T tanh(\boldsymbol{W}\boldsymbol{r}_{t-1} + \boldsymbol{V}\boldsymbol{h}_t + \boldsymbol{U}\boldsymbol{f}_{l,t} + \boldsymbol{b}), \tag{5}$$

$$\boldsymbol{f}_l = \boldsymbol{F} * \boldsymbol{\alpha}_{l-1}, \tag{6}$$

where $*$ denotes 1-dimensional convolution. Using $\boldsymbol{g}_l$ and $\boldsymbol{r}_{l-1}$, the decoder predicts the next label y_l as:

$$y_l \sim Generate\left(\boldsymbol{r}_{l-1}, \boldsymbol{g}_l\right), \tag{7}$$

where the Generate function is implemented as:

$$\boldsymbol{R} \tanh\left(\boldsymbol{P}\boldsymbol{r}_{l-1} + \boldsymbol{Q}\boldsymbol{g}_l\right). \tag{8}$$

The objective for training the attention models is a cross entropy loss calculated between the predicted label sequences and the target label sequences.

A runtime decoding algorithm for word-level attention-based models is presented in Algorithm 1. This algorithm returns the N-best sentence candidates using a decreasing beam width initialized with N. It is simple since we do not need to incorporate external dictionaries or language models. $\langle$sos$\rangle$ and $\langle$eos$\rangle$ are special symbols for the start and end of a sentence. The posterior probability of a word at each decoding step

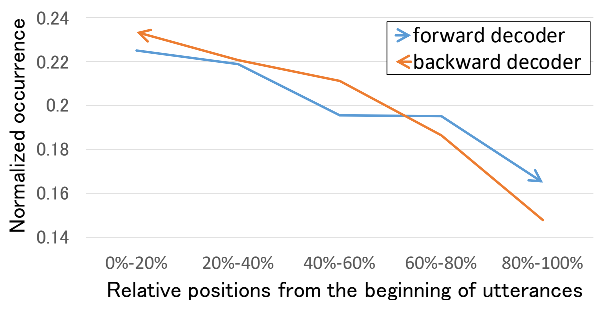

Figure 1: *Normalized occurrence counts of substitution errors at different relative positions of utterances in CSJ-TESTSET1*

$p(y|\boldsymbol{s}, \boldsymbol{X})$ on line 9 and 12 is calculated using formulas from (1) to (8). After performing Algorithm 1, we rescore each sentence candidate $\boldsymbol{s}$ in F using an insertion penalty λ as follows:

$$rescore(\boldsymbol{s}) = score(\boldsymbol{s}) - \lambda|\boldsymbol{s}|, \tag{9}$$

where $|\boldsymbol{s}|$ is the length of sequence $\boldsymbol{s}$, and $score(\boldsymbol{s})$ is the value calculated on line 12 of Algorithm 1. We output the sentence with the largest $rescore(\boldsymbol{s})$.

A backward decoder for right-to-left decoding and its encoder network can be trained in the same way as a forward decoder, with an exception that it takes flipped versions of acoustic features and labels as input and target, respectively. In the runtime of backward decoding, $\langle$sos$\rangle$ and $\langle$eos$\rangle$ are swapped in Algorithm 1 correspondingly.

3. Forward-backward attention

Here, we describe in depth the proposed method that integrate forward and backward attentions. This approach is motivated by the following two considerations.

First, attention-based encoder-decoder models are generally not good at decoding long utterances as mentioned in [9]. They tend to output less reliable labels in the latter part of utterances due to error accumulation. Figure 1 shows how many substitution errors word-level forward and backward decoders made at different relative positions of utterances in a Japanese speech recognition experiment [1]. In this figure, the forward decoder made more errors than the backward counterpart in the later part of utterances, while we observe an opposite tendency in the earlier part. This shows that unidirectional decoders tend to be more accurate at their early decoding steps, which leads to a strategy that concatenates reliable parts of sequences obtained from two directional decoders. These newly generated hypotheses can have fewer errors than the original ones, if this concatenation process is managed in an appropriate way.

Second, because speech has an asymmetric time structure, forward and backward decoders may have strengths in different ways which may contribute to better accuracy complementarily. Therefore, we expect a performance gain from simply combining their recognition results, considering the consistent effectiveness of the system combination approaches [18][19] in many ASR tasks such as noisy speech recognition [20].

3.1. Decoding

We propose a 3-pass decoding algorithm utilizing forward and backward attention decoders which do not undermine the

[1]This is the same experiment as shown in the first column of Table 1. The mean and standard deviation of utterance length are 5.6s and 3.9s.

Algorithm 2 ForwardBackwardBeamSearch($N, \boldsymbol{X}$)

```
 1: $F^f \Leftarrow$ ForwardBeamSearch($N, \boldsymbol{X}$)
 2: $F^b \Leftarrow$ BackwardBeamSearch($N, flip(\boldsymbol{X})$)
 3: $F \Leftarrow \{\emptyset\}$
 4: for $\boldsymbol{s}^f \in F^f$, $\boldsymbol{s}^b \in F^b$ do
 5:     $j\_next = |\boldsymbol{s}^b|$
 6:     for $i$ in 1,...,$|\boldsymbol{s}^f|$ do
 7:         for $j$ in $j\_next$,...,1 do
 8:             if $s_i^f = s_j^b$ and $time(s_i^f) \sim time(s_j^b)$ then
 9:                 $\boldsymbol{s}^* \Leftarrow concat(\boldsymbol{s}^f_{<=i}, flip(\boldsymbol{s}^b_{<j}))$
10:                 $score(\boldsymbol{s}^*) = score^f(\boldsymbol{s}^f_{<i}) + score^b(\boldsymbol{s}^b_{<j}) + \log(\max(p^f(s_i^f|\boldsymbol{s}^f_{<i}, \boldsymbol{X}), p^b(s_j^b|\boldsymbol{s}^b_{<j}, flip(\boldsymbol{X})))$
11:                 add $s^*$ to $F$
12:                 $j\_next = j - 1$
13:                 break
14:             end if
15:         end for
16:     end for
17: end for
18: return set of sentence candidates $F$
```

low-latency advantage of word-level models. We summarize it in Algorithm 2. It was inspired by decoding methods for HMM acoustic models based on forward-backward search [21][22][23]. Pass 1 performs left-to-right decoding using the forward decoder. In Pass 2, speech is decoded from right to left using the backward decoder taking flipped acoustic features, which we denote as $flip(\boldsymbol{X})$ in Algorithm 2, as input. Pass 3 integrates the sentence hypotheses from Pass 1 and Pass 2 and generates new candidates by concatenating the partial sentences.

The key points for this algorithm are (1) how to find the partial sequences to be concatenated from hypotheses obtained with the two decoders, and (2) how to give reliable scores to the newly generated sentence candidates. As for the first issue, we first pick up two sentence hypotheses which share the same word appearing at approximately the same time frame. We then split each of these two sentence hypotheses at this word and concatenate the earlier half of the forward hypothesis and the later half of the backward hypothesis to generate a new sentence hypothesis.

More precisely, we judge that the i-th word s_i^f in a forward sentence candidate $\boldsymbol{s}^f$ and the j-th word s_j^b in a backward candidate $\boldsymbol{s}^b$ appear at an approximately same time frame (which we denote $time(s_i^f) \sim time(s_j^b)$), if the following condition is met:

$$time(s_{j+1}^b) < time(s_i^f) < time(s_{j-1}^b), \qquad (10)$$

where we define the occurrence time of words using forward and backward attention vectors as:

$$time(s_i^f) \overset{\text{def}}{=} \arg\max(\boldsymbol{\alpha}_i^f), \qquad (11)$$

$$time(s_j^b) \overset{\text{def}}{=} T - \arg\max(\boldsymbol{\alpha}_j^b), \qquad (12)$$

where T is the frame length of the input speech. $\boldsymbol{\alpha}_i^f$ and $\boldsymbol{\alpha}_j^b$ are the forward attention vector at the i-th forward decoding step and the backward attention vector at the j-th backward decoding step, respectively. This is a natural choice because attention vectors have their maximum values around the center frame of the corresponding words in most cases.

The generation process of new sentence candidates is as follows. From any pair of forward and backward sentence hypotheses, we pick pairs of words whose occurrence time are potentially similar based on their positions in hypotheses. We examine if the condition given in equation (10) is true for the word pair and if these word labels are the same. If s_i^f and s_j^b satisfy the conditions, we generate a new sentence candidate $\boldsymbol{s}^*$ as:

$$\boldsymbol{s}^* = concat(\boldsymbol{s}^f_{<=i}, flip(\boldsymbol{s}^b_{<j})), \qquad (13)$$

where $\boldsymbol{s}^f_{<=i}$ is a partial sequence beginning from ⟨sos⟩ and ending at the i-th word of the original forward hypothesis, and $\boldsymbol{s}^f_{<j}$ is a partial sequence beginning from ⟨eos⟩ and ending at the $(j-1)$-th word of the original backward hypothesis[2]. Because these two partial sentences end at the same word appearing at an approximately same time frame, the score for this concatenated sequence $\boldsymbol{s}^*$ is defined as:

$$score(\boldsymbol{s}^*) = score^f(\boldsymbol{s}^f_{<i}) + score^b(\boldsymbol{s}^b_{<j}) + \log(\max(p^f(s_i^f|\boldsymbol{s}^f_{<i}, \boldsymbol{X}), p^b(s_j^b|\boldsymbol{s}^b_{<j}, flip(\boldsymbol{X}))). \qquad (14)$$

Note that the double loop from line 6 to 16 runs in $\mathcal{O}(L)$ time where L is the length of the longest sentence candidate of F^f and F^b, since we only need to check the pairs of words from $\boldsymbol{s}^f_{>i}$ and $\boldsymbol{s}^b_{<j}$ after we confirm that s_i^f and s_j^b satisfy the condition on line 8. After all candidates are generated, we rescore them using a formula (9) and output the sentence with the largest score.

3.2. Multitask learning of decoders

We also propose to integrate the forward and backward attentions in the training stage of decoders. This is implemented as a multitask learning (MTL) framework, in which the forward and backward decoders are trained sharing a single encoder. We aim to regularize the optimization process of one decoder using the loss of the other decoder. The loss function for MTL of the forward decoder is defined as:

$$loss_f^{(MTL)} = \alpha \cdot loss_f + (1 - \alpha) \cdot loss_b, \qquad (15)$$

where $loss_f$ and $loss_b$ are the original cross-entropy loss functions of the forward and backward decoders, respectively. α is the weight for the main task, which may be set to be larger than 0.5. The shared encoder is trained so that the encoded acoustic representation is expected to be suitable for both of left-to-right and right-to-left decoding.

4. Experimental evaluations

We evaluated our methods through speech recognition tasks using the Corpus of Spontaneous Japanese (CSJ) [24]. CSJ includes two distinct subcorpora, namely, CSJ-APS and CSJ-SPS. CSJ-APS consists of academic presentation speeches on several topics such as science, engineering, humanities and social science. CSJ-SPS consists of simulated presentation speeches on three general themes. These subsets have their own official test sets, namely, CSJ-TESTSET1 and CSJ-TESTSET3.

A 40-dimensional vector consisting of 40-channel log Mel-scale filterbank (lmfb) outputs is used as acoustic features for attention models. Non-overlapping frame stacking [6] was applied to these features in which we stack and skip three frames to make a new super frame. The acoustic encoders in our attention models consist of 3 or 5-layers of bidirectional LSTMs with

[2]This algorithm guarantees that sentences identical to original candidates are also generated due to the existence of ⟨sos⟩ and ⟨eos⟩.

Table 1: *ASR performance for two tasks (WER(%))*

decoder	CSJ-APS (224 hrs) CSJ-TESTSET1	CSJ-SPS (251 hrs) CSJ-TESTSET3
forward (baseline)	14.51	11.41
+ MTL	14.27	11.21
backward	13.82	11.57
+ MTL	13.58	11.26
forward-backward	**12.92**	**10.38**
+ MTL	**12.67**	**10.24**

Table 2: *Recognition error rates of different types for CSJ-TESTSET1 (%)*

	error type			
decoder	del	sub	ins	total
forward	2.24	9.67	2.60	14.51
+ MTL	2.28	9.71	2.27	14.27
backward	2.47	9.37	1.99	13.82
+ MTL	2.41	9.31	1.86	13.58
forward-backward	2.36	8.70	1.86	12.92
+ MTL	2.24	8.71	1.70	12.67

320 cells. Dropout [25] was used for training each LSTM layer with a dropout rate of 0.2. The decoder consists of a 1-layer LSTM with 320 cells and a softmax output layer with the nodes for vocabulary words, ⟨sos⟩, ⟨eos⟩ and ⟨UNK⟩ special token for out-of-vocabulary words. The sizes of word vocabularies for CSJ-APS and CSJ-SPS are 19,146 and 24,826, respectively. We used Adam [26] for optimizing network parameters. We also used gradient clipping with a threshold of 5.0. All network parameters were initialized with random values drawn from a uniform distribution with range (-0.1, 0.1). The input data are sorted by the length of frames before creating minibatches of 30 sentences. While we used Chainer toolkit [27] to train the networks with a 3-layer encoder, we used Pytorch [28] for the larger models with a 5-layer encoder exploiting its advantage of memory efficiency. All experiments were carried out on a system with a 3.60 GHz Intel Xeon and a NVIDIA TITAN X.

4.1. Results with small models

First, we show the results of a number of comparative experiments using smaller models with a 3-layer encoder. The beam width N was set to be 4 in all experiments which gave the lowest WERs. Table. 1 shows the word error rates (WERs) obtained for two tasks. By comparing the results against the baseline forward and backward decoders, we can find the proposed forward-backward decoder achieves significant improvement in both tasks. Our MTL method yielded small but consistent improvements for all types of decoders. We set the weight α in the MTL to be 0.8. The proposed forward-backward decoding combined with the MTL improved WERs of word-attention models by 12.7 % and 10.3 % relative for CSJ-TESTSET1 and CSJ-TESTSET3, respectively.

We show the error rates of different types (deletion, substitution and insertion) separately in Table 2 in order to investigate in depth the reason for the performance gains from our methods. From this table, we understand that the MTL mainly contributes in reducing insertion errors. On the other hand, the forward-backward decoding improved substitution error rates significantly, in addition to reducing deletion and insertion errors to the level of the lower rate of either decoder.

Table 3: *ASR performance and real time factor (RTF) of proposed method implemented and evaluated with larger models trained using larger data for CSJ-TESTSET1*

model		WER(%)	RTF
DNN-HMM + LM		13.62	0.925
phone-CTC + LM		14.15	0.581
forward	N=4	11.27	0.057
	N=2	11.36	0.020
	N=1	11.87	0.010
forward-backward	N=4	**10.17**	**0.119**
	N=2	**10.09**	**0.040**
	N=1	**10.17**	**0.019**

4.2. Results with large models

Here, we present the results using larger models with a 5-layer encoder which have much higher baseline performances. These models were trained using both of CSJ-APS and CSJ-SPS. The vocabulary size was increased to 34,331 accordingly. We also used scheduled sampling [29] and label smoothing [30] to improve the optimization. A hybrid DNN-HMM model and a phone CTC model were also build using the same training set for comparing their ASR performances and real time factors against our attention-based word models. The DNN-HMM model has seven hidden layers with 2k rectified linear units (ReLUs) and a softmax output layer with 3k nodes. The CTC-based model has 5 layers of bidirectional LSTM with 320 memory cells. For decoding with the hybrid DNN-HMM and the phone CTC, we used the Julius decoder [23] and the EESEN WFST decoder [31], respectively.

The WERs and real time factors (RTF) for CSJ-TESTSET1 are shown in Table 3. We also show the results with narrower beam widths than 4 for attention models. The proposed method yielded a further significant performance gain (9.8 % relative) on top of the better baseline model when the beam width was 4. It only took exactly twice as long time as the forward decoder for completing recognition, showing that the Pass 3 in Algorithm 2 required only a negligible time as expected. Interestingly, the proposed method is not affected by a narrow beam width, while the performance of the forward decoder degraded as the beam width became narrower. The results with $N = 1$ show that our algorithm can recover many correct words from only single pair of candidates from two decoders, which the conventional unidirectional decoder can never find even with a wide beam width. The proposed method with $N = 1$ achieved a WER reduction of 25.3 % relative from the DNN-HMM coupled with a trigram language model with a decoding speed faster by a factor of 50.

5. Conclusion

We have proposed an effective decoding method by integrating forward and backward attentions for attention-based seq2seq models. The acoustic-to-word model enhanced with the proposed method achieved a WER reduction of 25.3 % relative from a standard hybrid DNN-HMM system with a decoding speed faster by a factor of 50. As future work, we are also interested in constructing an algorithm based on earlier integration of forward and backward decoding as in algorithms based on best-first search [22][23].

6. References

[1] G.E.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," *IEEE Signal Processing Magazine*, vol. 29, no. 6, pp. 82–97, 2012.

[2] W. Xiong, J. Droppo, X. Huang, F. Seide, M. Seltzer, A. Stolcke, D. Yu, and G. Zweig, "Achieving human parity in conversational speech recognition," in *arXiv preprint arXiv:1610.0525*, 2016.

[3] G. Saon, G. Kurata, T. Sercu, K. Audhkhasi, S. Thomas, D. Dimitriadis, X. Cui, B. Ramabhadran, M. Picheny, L.-L. Lim, B. Roomi, and P. Hall, "English conversational telephone speech recognition by humans and machines," in *arXiv preprint arXiv:1703.02136*, 2017.

[4] A. Graves, S. Fernandez, F. Gomez, and J. Schmidhuber, "Connectionist temporal classification : Labelling unsegmented sequence data with recurrent neural networks," in *Proc. of the 23st International Conference on Machine Learning*, 2006, pp. 369–376.

[5] A. Graves and N. Jaitly, "Towards end-to-end speech recognition with recurrent neural networks," in *Proc. of the 31st International Conference on Machine Learning*, 2014, pp. 1764–1772.

[6] H. Sak, A. Senior, K. Rao, and F. Beaufays, "Fast and accurate recurrent neural network acoustic models for speech recognition," in *arXiv preprint arXiv:1607.06947*, 2015.

[7] H. Sak, F. de Chaumont Quitry, T. Sainath, and K. Rao, "Acoustic modelling with CD-CTC-SMBR LSTM RNNs," in *ASRU*, 2015, pp. 604–609.

[8] J. Chorowski, D. Bahdanau, K. Cho, and Y. Bengio, "End-to-end continuous speech recognition using attention-based recurrent NN: first results," in *NIPS: Workshop Deep Learning and Representation Learning Work- shop*, 2014.

[9] J. Chorowski, D. Bahdanau, D. Serdyuk, K. Cho, and Y. Bengio, "Attention-based models for speech recognition," in *Advances in Neural Information Processing Systems (NIPS)*, 2015, pp. 577–585.

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

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

[12] A. Graves, "Sequence transduction with recurrent neural networks," in *LCML*, 2012, pp. 4945–4949.

[13] A. Graves, A. rahman Mohamed, and G. Hinton, "Speech recognition with deep recurrent neural networks," in *ICASSP*, 2013, pp. 6645–6649.

[14] K. Audhkhasi, B. Ramabhadran, G. Saon, M. Picheny, and D. Nahamoo, "Direct acoustics-to-word models for English conversational speech recognition," in *Interspeech*, 2017, pp. 959–963.

[15] H. Soltau, H. Liao, and H. Sak, "Neural speech recognizer: Acoustic-to-word LSTM model for large vocabulary speech recognition," in *Interspeech*, 2017, pp. 3707–3711.

[16] S. Ueno, H. Inaguma, M. Mimura, and T. Kawahara, "Acoustic-to-word attention-based model complemented with character-level CTC-based model," in *ICASSP*, 2018.

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

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

[19] L. Mangu, E. Brill, and A. Stolcke, "Finding consensus in speech recognition: word error minimization and other applications of confusion networks," vol. 14, no. 1, pp. 373–400, 2000.

[20] T. Menne, J. Heymann, A. Alexandridis, K. Irie, A. Zeyer, M. Kitza, P. Golik, I. Kulikov, L. Drude, R. Schlüter, H. Ney, R. Häb-Umbach, and A. Mouchtaris, "The RWTH/UPB/FORTH system combination for the 4th CHiME challenge evaluation," 2016.

[21] S. Audtin, R. Schwartz, and P. Placeway, "The forward-backward search algorithm," in *ICASSP*, 1991.

[22] F. K. Soong and E.-F. Huang, "A tree-trellis based fast search for finding the N best sentence hypotheses in continuous speech recognition," in *ICSLP*, 1990.

[23] A. Lee, T. Kawahara, and K. Shikano, "Julius : an open source real-time large vocabulary recognition engine," in *EUROSPEECH*, pp. 1691–1694,.

[24] K.Maekawa, "Corpus of Spontaneous Japanese: Its design and evaluation," in *Proc. ISCA & IEEE Workshop on Spontaneous Speech Processing and Recognition*, 2003, pp. 7–12.

[25] 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.

[26] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," in *arXiv preprint arXiv:1412.6980*, 2014.

[27] S. Tokui, K. Oono, S. Hido, and J. Clayton, "Chainer: a next-generation open source framework for deep learning," in *Proceedings of Workshop on Machine Learning Systems (LearningSys) in The Twenty-ninth Annual Conference on Neural Information Processing Systems (NIPS)*, 2015.

[28] 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," 2017.

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

[30] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna, "Rethinking the inception architecture for computer vision," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2016.

[31] 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.