SIP (2020), vol. 9, e25, page 1 of 7 ©. The Author(s), 2020. Published by Cambridge University Press.
doi:10.1017/ATSIP.2020.23

ORIGINAL PAPER

End-to-end recognition of streaming Japanese speech using CTC and local attention

JIAHAO CHEN,[1] RYOTA NISHIMURA[1] AND NORIHIDE KITAOKA[2]

Many end-to-end, large vocabulary, continuous speech recognition systems are now able to achieve better speech recognition performance than conventional systems. Most of these approaches are based on bidirectional networks and sequence-to-sequence modeling however, so automatic speech recognition (ASR) systems using such techniques need to wait for an entire segment of voice input to be entered before they can begin processing the data, resulting in a lengthy time-lag, which can be a serious drawback in some applications. An obvious solution to this problem is to develop a speech recognition algorithm capable of processing streaming data. Therefore, in this paper we explore the possibility of a streaming, online, ASR system for Japanese using a model based on unidirectional LSTMs trained using connectionist temporal classification (CTC) criteria, with local attention. Such an approach has not been well investigated for use with Japanese, as most Japanese-language ASR systems employ bidirectional networks. The best result for our proposed system during experimental evaluation was a character error rate of ***9.87****%.*

Keywords: CTC, Local attention, Speech recognition, Streaming recognition

Received 8 July 2020; Revised 25 October 2020

I. INTRODUCTION

Over the last several years, deep neural networks have been used to achieve state-of-the-art performance in large scale, automatic speech recognition (ASR) tasks. Various types of deep neural networks have been used in ASR systems, such as convolutional neural networks (CNNs) [1, 2] and recurrent neural networks (RNNs) [3]. These networks have also been used as feature extractors in hybrid systems [4] and in end-to-end (E2E) systems [5].

Achieving alignment between system input and output is a major concern in ASR systems. Since the number of input audio frames is much larger than the number of symbols in the output transcript, a large amount of human labor is needed to manually label the audio frames. In order to solve this problem, several neural network-based models have been proposed over the past few years. The three methods which are most widely used are: connectionist temporal classification (CTC) [6], RNN-transducer (RNN-T) [7], and attention-based encoder–decoder architectures [8, 9].

RNN-T-based models are generally used to decode streaming data. The output of the previous time frame is recursively input into the model to reduce the effect of the conditional independency of CTC-based models. However, RNN-T models cannot look ahead to consider future outputs.

Attention-based encoder–decoder systems have achieved good results in various fields such as neural machine translation [10], sentence summarization [11], and image classification [12]. When using this method, there is no need to label the training data frame by frame, making the training process much more efficient. The attention mechanism approach also provides more effective language modeling than other types of neural networks. Google's BERT model, for example, [13] uses only attention blocks [14] to perform natural language processing (NLP) tasks.

Attention mechanism-based architectures such as Transformer use methods similar to local attention, however such methods only use short chunks of data at a time thus they cannot make use of context from the distant past like systems based on long short-time memorys (LSTMs). These methods also require large computational resources when running however, so systems using them must be operated on a cloud server. Furthermore, some speech recognition tasks require short waiting times or real-time capability, so attention mechanism-based architectures are generally not used in such applications.

Exact hard monotonic attention [15] and monotonic chunkwise attention [16] have been proposed for aligning input and output sequences and reducing computational cost. A similar method was proposed in [17], in which the authors constructed a self-attention-based

[1]Tokushima University, 2-1 Minamijohsanjima-cho, Tokushima, Japan
[2]Toyohashi University of Technology, 1-1 Hibarigaoka Tempaku-cho, Toyohashi, Aichi, Japan

Corresponding author:
Norihide Kitaoka
Email: kitaoka@tut.jp

encoder–decoder model to generate streaming recognition results. Unlike CTC-based models, which can use a considerable amount of past context, this method only uses short chunks of data at a time.

In [18], a CTC-based classifier was used to control the activation of the attention-based classifier in order to maintain monotonic frame synchronization, but the proposed system does not support streaming speech recognition.

Unlike translation or other NLP tasks, the attention distributions in speech recognition tasks are almost monotonic, therefore some attention mechanism ASR approaches are based on monotonic alignment [15], which is the same assumption that ASR systems using CTC and RNN-T are based on.

Because both CTC and RNN-T make use of monotonic alignment, both approaches are capable of performing streaming step prediction, which can function as a language model, helping the ASR system achieve better results. But these models are more complex than CTC models, which have a simpler structure compared to other methods and produce a strict left-to-right alignment of the audio frames and transcript symbols. Moreover, CTC is a more acoustic-based model than the other methods, thus it is possible to integrate it with language models in a more theoretically sound manner. CTC is also often used as a sub-technique for attention-based models [19], although the overall system does not have streaming capability. CTC has rarely been evaluated as a primary method for decoding streaming Japanese speech, as research in Japan is mainly focused on improving recognition accuracy rather than on developing the capability to process streaming data.

In this study, we evaluate how such a CTC-based model performs with streaming Japanese speech data. In order to improve recognition performance, we introduce local attention [20] into our model, which is sometimes also referred to as local monotonic attention [21], or time restricted attention [22]. This technique generally seems to be helpful for improving the performance of CTC-based systems. However, in applications where there is a high rate of output, too much repeated and blank output is produced, making it more difficult to train the attention mechanism. Thus, local attention is more effective for reducing word error rate in situations where the output frames are processed at lower rates. To our knowledge, our study is the first to propose downsampling speech frames in order to speed up recognition while maintaining recognition accuracy, and to also investigate the tradeoff between recognition efficiency and accuracy.

The rest of this paper is organized as follows: In Section II, we explain CTC and additive attention, the two primary techniques our end-to end speech recognition model is based on, as well as the use of down-sampling to increase efficiency. We then introduce local attention in Section III, and provide further details about the implementation of our proposed method. The procedure for our experimental evaluation is explained in Section IV, and the results of our experiments are discussed in Section V. Finally, we conclude this paper in Section VI.

II. E2E SPEECH RECOGNITION

Most E2E neural networks, such as the CNNs used for image classification, or the RNNs used for text generation, require corresponding input data and ground truth labels, which means all of the data need to be labeled manually at a high cost, and this manual labeling process is even more difficult for speech recognition tasks. Therefore, well-tuned GMM-HMM-based models are generally used to label the data frame by frame. But thanks to advances in neural network development in recent years, it has also become possible to directly train models using weakly labeled data.

In the following sub-sections, we will discuss some of the key concepts on which our proposed model for automatic recognition of streaming Japanese speech is based.

A) CTC

A CTC-based network contains basically two parts: an encoder and a CTC loss criterion. The encoder can be any type of network, but is usually an RNN. The CTC component aligns prediction and transcription during the training process.

Since the encoder functions at the frame level, the output sequence will always have the same length as the input sequence, but the output of the encoder is always longer than the actual symbol sequence.

In order to solve this problem, CTC-based methods include a blank symbol in the label set and allow the output to include the blank symbol, as well as allowing symbol repetitions. Readable output is only obtained after post-processing.

The encoder generates a possibility prediction distribution lattice $p(y|x)$. If we denote the possibility path as π, and the input as x, this encoding process can be represented as follows:

$$p(\pi|x) = \prod_{t=1}^{T} p(\pi_t|x_t), \tag{1}$$

where T is the duration of input x.

Because of the CTC's assumption of conditional independence, prediction is only based on the current acoustic input, so in order to improve the accuracy of the recognition results, an external language model is usually needed. In other words, a CTC is an almost purely acoustic model, allowing us to integrate it with language models in a theoretically sound manner.[1]

As can be seen in Fig. 1, the direct output of a CTC network contains many repetitions of symbols and blank labels, so in order to obtain the final readable transcription, we merge repeated symbols and drop the blank labels [23].

B) Additive attention mechanism

In the encoder–decoder model, the encoder first encodes the input sequence into a fixed-length vector, which the

[1]The integration of acoustic and language models is outside the scope of this study, but is one of our projects for future research.

https://doi.org/10.1017/ATSIP.2020.23 Published online by Cambridge University Press

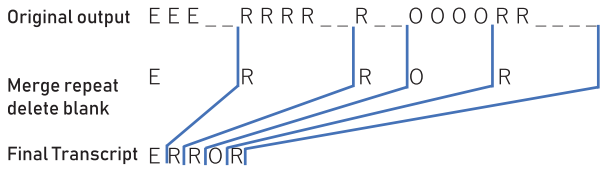

Fig. 1. How a CTC collapses data.

decoder then uses to generate outputs. But compressing all of the information into a fixed-length vector is obviously not ideal for sequential data.

By using an attention mechanism, we can use all of the historic outputs of the encoder instead of only using the last output. This allows us to focus on changes in the encoded vector sequence over time. Thus can be considered as a kind of memory mechanism that can be used to store all of the output history for later use. It also produces a soft alignment between the input and output steps, which can be represented as follows:

$$p(y|x) = \prod_{u=1}^{U} p(y_u|y_{1:u-1}, c_u), \tag{2}$$

where U is the current decoding step and c_u is the context vector at decoding step u, which is calculated as the sum of all of the hidden outputs h_t multiplied by attention weight α:

$$c_u = \sum_{t=1}^{T} \alpha_{u,t} h_t, \tag{3}$$

$$\alpha_u = Attend(h, s_{u-1}), \tag{4}$$

where s_{u-1} indicates the decoder state in the previous time step. Thus, the Attend function here is actually trying to decide how important each hidden output is to the current decoding step:

$$e_{u,t} = Score(s_{u-1}, h_t), \quad t = 1, \ldots, T, \tag{5}$$

$$\alpha_{u,t} = \frac{\exp(e_{u,t})}{\sum_{t'=1}^{T} \exp(e_{u,t'})}, \tag{6}$$

$$e_{u,t} = v^T \tanh(Us_{u-1} + Wh_t + b). \tag{7}$$

The score of each input step is then computed using a simple feed forward network and the parameters of the scoring function (v, U, W, and b), which are learned during the training process.

C) Downsampling

In order to increase the speed of training and testing, various methods can be used to reduce the number of input steps. In conventional speech recognition methods, this is done by stacking several frames together, and by maintaining some overlap in order to prevent cut-offs in the middle of words.

But since we are using a neural network, maxpooling can be used to do this instead. Thus, in our proposed method we have replaced frame stacking with maxpooling [24], and have applied it to some of the CNN layers, with each layer followed by a different width of maxpooling in order to obtain different downsampling rates.

III. DETAILS OF OUR APPROACH

CTC-based methods can use weakly labeled data and have simpler structures than other neural network models, but since CTC is based on an assumption of conditional independence, these methods may not be able to achieve the same level of accuracy as encoder–decoder models, which are based on an assumption of conditional dependence. The use of an attention mechanism in encoder–decoder models has been proven to enhance recognition performance significantly.

Visualizations of attention distributions are almost monotonic in appearance, which leads us to believe that it might not be necessary to perform full attention. Using local attention instead may improve performance while allowing us to retain the ability to perform streaming processing. Therefore, unlike encoder–decoder models, we only use local attention in our CTC-based approach, as shown in Fig. 2.

Our model assumes that the current output depends on several nearby encoding outputs, so the local attention mechanism allows our model to use contextual information from short periods of time in the past and future. Indeed, the final outputs after performing attention and layer normalization are not used, and thus direct conditional dependence among the outputs is not realized, but some dependence among the outputs can be expressed. In other words, our model is a combination of an LSTM, which captures a long period of prior history, and a contextual information chunk, described in [17], which captures information during the short period around the target time frame. This can be expressed as follows:

$$p(\pi|x) = \prod_{t=1}^{T} p(\pi_t|x_t, c_t), \tag{8}$$

where contextual information c_t is calculated at each decoding step from a small local region of hidden outputs:

$$c_t = \sum_{w=-W}^{W} \alpha_{t,w} h_t, \tag{9}$$

$$\alpha_t = Attend(h_{t-W:t+W}, s_{t-1}), \tag{10}$$

where $\alpha_t = \{\alpha_{t,-W} \ldots \alpha_{t+W}\}$, s_t and where W indicates the size of the attention window. Thus, total length of the local region will be $2 \times W + 1$. This gives us the benefit of using an attention mechanism while still allowing the system to retain its ability to process streaming data. During streaming speech recognition, the post processing module, which receives the outputs of the CTC model eliminates the blank and repeated symbols to generate the final output incrementally, without waiting for the end of the input sequence.

https://doi.org/10.1017/ATSIP.2020.23 Published online by Cambridge University Press

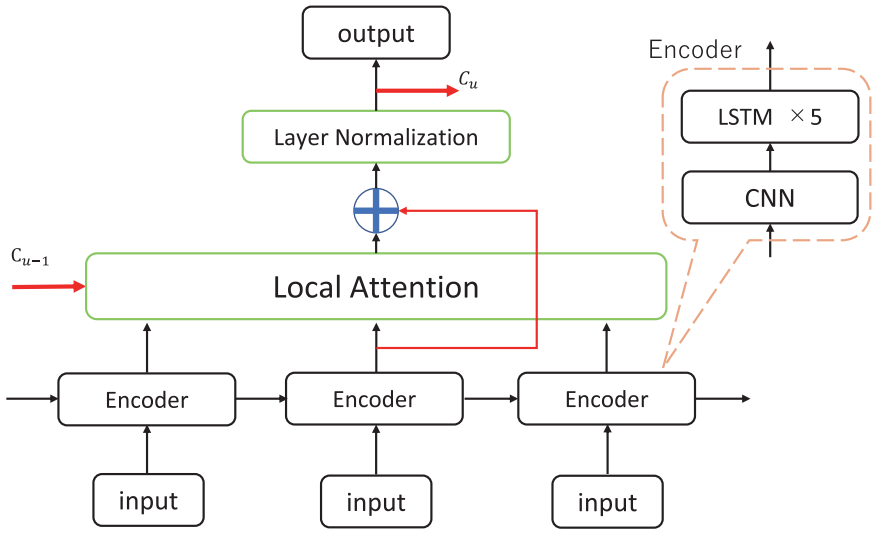

Fig. 2. Overview of the proposed model.

We also added a skip connection [25] to the local attention mechanism, and concatenated the encoder vector to context vector c_t.

IV. EXPERIMENTAL SETUP

A) Data set

The data we used to train the proposed model in this study and evaluate its performance came from the Corpus of Spontaneous Japanese (CSJ), a dataset which contains 661 hours of audio data in total. The dataset is divided into two categories: Academic Presentation Speech (APS) and Simulated Public Speaking (SPS). APS contains live recordings of research presentations from various academic societies, including science, engineering, humanities, and social science. This data set has a bias in terms of age and gender, since most of speakers are male postgraduate students. SPS contains talks on everyday topics given by a wide range of speakers, and is balanced by age and gender as much as possible.

All of the audio data were recorded at a sampling rate of 16 kHz. We preprocessed the data to extract 40 dimensions of log filter bank features with a hop size of 10 ms and window size of 25 ms. We also extended the data by adding delta and delta–delta features, and the data were normalized to have a mean of 0 and a variance of 1.

In order to reduce the number of input frames, we used maxpooling layers as explained in Section II-2.3. The downsampling rate depends on what language is being used as well as on whether word or character level data are being processed. Our system uses Japanese, and outputs the results at character level (i.e. Hiragana, Katakana, and Kanji characters), with a vocabulary size of 3260.

B) Model configuration

Our models consisted of unidirectional LSTMs with five layers, each layer of which contained 512 units, and the

Table 1. Configuration of hyperparameters for model training

Dropout rate	0.5
Gradient clipping	5
Optimizer	Adadelta
LR	1.0
Adadelta ρ	0.95
Adadelta eps	1×10^{-8}
Adadelta eps decay	0.01
Maximum epoch	15
Batchsize	50

Table 2. Configuration of feature extraction CNN

Output
Maxpooling stride = 2
Convolution in = 128, out = 128, kernel size = 3
Convolution in = 64, out = 128, kernel size = 3
Maxpooling stride = 2 or 3
Convolution in = 64, out=64, kernel size = 3
Convolution in = 1, out = 64, kernel size = 3
Input

models were trained using the CSJ dataset. Our baseline model was based on the model provided in the ESPnet-toolkit [19], which is an integration of CTC and encoder–decoder models, however we only used the CTC component. We also added an original beam search component, with a beam size of 20, and extended the baseline model using local attention. Details of the configurations of the hyperparameters used for model training are shown in Table 1.

We also used a VGG-like CNN [26] for feature extraction, the configuration details of which are shown in Table 2. The extracted features were then fed into the CTC.

Sub-sampling using max pooling resulted in 40 ms of latency for 1/4 subsampling and 60 ms of latency for 1/6 subsampling, because the frame shift was 10 ms. Lookahead was six frames in both cases. When the center frame was the target frame, an attention window size of 13 was used, and

https://doi.org/10.1017/ATSIP.2020.23 Published online by Cambridge University Press

Table 3. Model configurations, with their associated latencies and CERs

Model no.	CNN	Attn. unit	Attn. window size	Sub-sampling rate	Latency (ms)	CER eval1	CER eval2	CER eval3	Avg. CER
1				1/4	40	14.30%	11.00%	12.50%	12.60%
2				1/6	60	15.30%	12.10%	13.60%	13.67%
3		✓	13	1/4	240	12.10%	9.00%	10.00%	10.37%
4		✓	13	1/6	360	11.90%	9.00%	10.30%	10.40%
5	✓			1/4	40	11.90%	8.60%	9.60%	10.03%
6	✓			1/6	60	14.10%	10.70%	12.00%	12.27%
7	✓	✓	13	1/4	240	11.60%	8.70%	9.30%	9.87%
8	✓	✓	13	1/6	360	12.10%	8.80%	9.90%	10.27%
9	✓	✓	7[a]	1/6	360	11.90%	9.00%	9.80%	10.23%

[a] The width of the attention window was halved and only look-ahead was performed.

when the last frame was the target frame, an attention window size of 7 was used. Thus, the total latencies were 240 ms (=40 ms × 6 frames) and 360 ms (=60 ms × 6 frames) for 1/4 sub-sampling and 1/6 sub-sampling, respectively, with local attention.

V. RESULTS

We performed our experiments while varying the conditions as follows: with and without the CNN feature extractor, with and without local attention, and setting the sub-sampling rate to 1/4 or 1/6. Our experimental results are shown in Table 3.

A) CNN feature extractor

First, we compared the results with and without the CNN-based feature extractor (without the local attention mechanism). The first and second rows in Table 3 show the results without the CNN feature extractor, while the fifth and sixth rows show the results with the CNN feature extractor. Without the CNN, we obtained a CER of 12.60% on average at a sub-sampling rate of 1/4. When we change the sub-sampling rate to 1/6, performance was degraded by 1.97%. Here, sub-sampling was done by decimating the time sequence in the first and second layers of the LSTMs. When using the CNN, we obtained a CER of 10.03% at a sub-sampling rate of 1/4, which is much better than under the 'without CNN' condition. However, when using a sub-sampling rate of 1/6, we obtained a significantly degraded CER which was 2.24% higher.

Overall, we observed that the smaller the sub-sampling rate, the poorer the recognition performance.

B) Local attention

Next, we included the local attention mechanism which was introduced in Section III, which consisted of a network of 200 units, with one head and a window width of 13 (i.e. $W = 7$ in equation (10)). Results with local attention are shown in the third, fourth, seventh, and eighth rows of Table 3. The third and fourth rows are the results without the CNN, while the seventh and eighth rows are the results with the CNN.

When using local attention without the CNN (third and fourth rows), the CERs fell to 10.37 and 10.40%, respectively, significant absolute reductions of 2.23 and 3.27%. We noted that there was no significant difference in performance when sub-sampling rates of 1/4 or 1/6 were used.

When using local attention with the CNN (seventh and eighth rows), we obtained a CER of 9.87% at a sub-sampling rate of 1/4, which was the best result achieved during these experiments. The improvement in performance when using local attention was 0.16%. At a sub-sampling rate of 1/6, we obtained a CER of 10.27%, which was a significant improvement of 2.00 percentage points compared to when local attention was not used. When comparing the use of 1/4 and 1/6 sub-sampling rates (the seventh and eighth rows of Table 3, respectively), the difference in performance was only 0.4%, demonstrating that the use of local attention lessened the gap in performance between the two sub-sampling rates. Use of high sub-sampling rates results in loss of information, thus recognition performance tends to be degraded. However, sub-sampling can reduce the number of blank symbols and symbol repetitions, allowing the window of the local attention mechanism to capture the informative sequences of the CNN outputs. Thus, the local use of attention can improve recognition performance especially at high sub-sampling rates. We have not tested our model with other languages, so we cannot say with a high degree of certainty that our model is especially suitable for Japanese. However, we believe one of the reasons for our model's superior performance is due to the phenomenon of mora isochronism in spoken Japanese. Japanese morae all have duration of approximately 200 ms (although duration can vary widely among speakers), and our use of local attention allows the model to capture each mora within its short acoustic context, allowing to achieve effective acoustic modeling.

C) Shortening the attention window

Figure 3 shows an image of the attention weight distributions for our model during our experiments, in which the lighter colors represent larger attention weights, revealing that the large attention weights are concentrated in the "future" inputs. This is because the LSTM layers accumulate

https://doi.org/10.1017/ATSIP.2020.23 Published online by Cambridge University Press

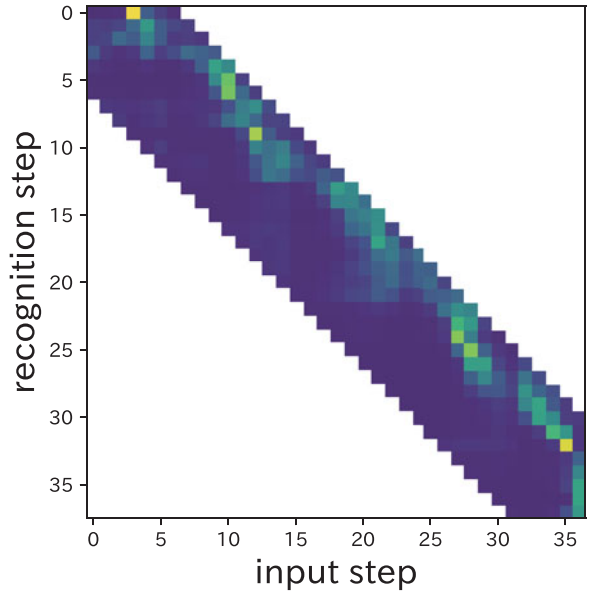

Fig. 3. Attention weight distributions.

information from "past" inputs and thus the attention mechanism does not need to attend the past inputs.

Based on this observation, we tried halving the attention window. This shorter window attends only current "near future" inputs, resulting in the window size shrinking to 7 from the original 13. The ninth row of Table 3 shows that we obtained a CER of 10.23% when using this smaller attention window, which is comparable to the accuracy rate achieved when using a longer window (as shown in the eighth row of the table).

VI. CONCLUSIONS

In this study, we proposed a method of extending a CTC-based model by using local attention to improve accuracy when performing streaming, E2E speech recognition of Japanese. We compared various model settings, specifically, with and without CNN-based feature extraction, at 1/4 and 1/6 subsampling rates, and with and without the use of local attention. Our proposed local attention mechanism was very robust to changes in the use of feature extraction and the in subsampling rate. The best performance was achieved when using the CNN-based feature extractor, the local attention mechanism and a subsampling rate of 1/4, which resulted in a CER of 9.87%, with the CER increasing to 10.27% when a subsampling rate of 1/6 was used. Halving the window size of the attention mechanism, i.e. using only "near future" information, did not degrade speech recognition performance. We were actually able to develop a real-time speech recognizer for Japanese using our proposed model.

Our future study includes integration of our model by using multiple, separately trained language models, to make it easy to adapt the recognizer to other recognition tasks.

FINANCIAL SUPPORT

This study was partially funded by the JSPS KAKENHI Grant-in-Aid for Scientific Research program, grant numbers 19H01125, 19K04311, and 17H01977.

CONFLICT OF INTEREST

The authors have no conflicts of interest to declare.

REFERENCES

[1] Collobert, R.; Puhrsch, C.; Synnaeve, G.: Wav2Letter: An end-to-end ConvNet-based speech recognition system. CoRR abs/1609.03193, 2016.

[2] Palaz, D.; Magimai-Doss, M.; Collobert, R.: Analysis of CNN-based speech recognition system using raw speech as input, in INTERSPEECH, 2015.

[3] Graves, A.; Mohamed, A.-r.; Hinton, G.E.: Speech recognition with deep recurrent neural networks, in ICASSP2013, 2013, 6645–6649.

[4] Hinton, G. *et al.*: Deep neural networks for acoustic modeling in speech recognition. *IEEE Signal Process. Mag.*, **29** (2012), 82–97.

[5] Amodei, D., *et al.*: Deep speech 2: End-to-end speech recognition in English and Mandarin, in International Conference on Machine Learning, 2016, 173–182.

[6] Graves, A.; Fernández, S.; Gomez, F.J.; Schmidhuber, J.: Connectionist temporal classification: Labelling unsegmented sequence data with recurrent neural networks, in ICML, 2006.

[7] Graves, A.: Sequence transduction with recurrent neural networks. arXiv preprint arXiv:1211.3711, 2012.

[8] Chan, W.; Jaitly, N.; Le, Q.; Vinyals, O.: Listen Attend and Spell: A neural network for large vocabulary conversational speech recognition, in 2016 IEEE ICASSP, 2016, 4960–4964.

[9] Chiu, C.-C., *et al.*: State-of-the-art speech recognition with sequence-to-sequence models, in 2018 IEEE ICASSP, 2018, 4774–4778.

[10] Luong, M.-T.; Pham, H.; Manning, C.D.: Effective approaches to attention-based neural machine translation. arXiv preprint arXiv:1508.04025, 2015.

[11] Rush, A.M.; Chopra, S.; Weston, J.: A neural attention model for abstractive sentence summarization. arXiv preprint arXiv:1509.00685, 2015.

[12] Wang, F., *et al.*: Residual attention network for image classification, in Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2017, 3156–3164.

[13] Devlin, J.; Chang, M.-W.; Lee, K.; Toutanova, K.: Bert: Pre-training of deep bidirectional transformers for language understanding. arXiv preprint arXiv:1810.04805, 2018.

[14] Vaswani, A., *et al.*: Attention is all you need, in Advances in Neural Information Processing Systems, 2017, 5998–6008.

[15] Raffel, C.; Luong, M.-T.; Liu, P.J.; Weiss, R.J.; Eck, D.: Online and linear-time attention by enforcing monotonic alignments, ICML2017, 2017.

[16] Chiu, C.-C.; Raffel, C.: Monotonic chunkwise attention. arXiv:1712.05382, 2017.

[17] Linhao, D.; Wang, F.; Xu, B.: Self-attention aligner: A latency-control end-to-end model for ASR using self-attention network and chunk-hopping, in IEEE ICASSP 2019, 2019, 5656–5660.

[18] Moritz, N.; Hori, T.; Le Roux, J.: Triggered attention for end-to-end speech recognition, in IEEE ICASSP2019, 2019, 5666–5670.

https://doi.org/10.1017/ATSIP.2020.23 Published online by Cambridge University Press

[19] Kim, S.; Hori, T.; Watanabe, S.: Joint CTC-attention based end-to-end speech recognition using multi-task learning, in 2017 IEEE ICASSP, 2017, 4835–4839.

[20] Mirasamadi, S.; Barsoum, E.; Zhang, C.: Automatic speech emotion recognition using recurrent neural networks with local attention, in IEEE ICASSP 2017, 2017, 2227–2231.

[21] Tjandra, A. Sakti, S.; Nakamura, S.: Local monotonic attention mechanism for end-to-end speech and language processing, in Proceedings of the Eighth International Joint Conference on Natural Language Processing, 2017, 431–440.

[22] Povey, D.; Hadian, H.; Ghahremani, P.; Li, K.; Khudanpur, S.: A time-restricted self-attention layer for ASR, in IEEE ICASSP 2018, 2018, 5874–5878.

[23] Hannun, A.: Sequence modeling with CTC. Distill, doi:10.23915/distill.00008. https://distill.pub/2017/ctc, 2017.

[24] Dong, L.; Zhou, S.; Chen, W.; Xu, B.: Extending recurrent neural aligner for streaming end-to-end speech recognition in Mandarin. arXiv preprint arXiv:1806.06342, 2018.

[25] He, K.; Zhang, X.; Ren, S.; Sun, J.: Deep residual learning for image recognition, in Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2016, 770–778.

[26] Hori, T.; Watanabe, S.; Zhang, Y.; Chan, W.: Advances in joint CTC-attention based end-to-end speech recognition with a deep CNN encoder and RNN-LM, INTERSPEECH 2017, 2017, 949–953.

Jiahao Chen received his M.S. degree from Tokushima University, Japan. The focus of his study was on end-to-end speech recognition.

Ryota Nishimura received his B.S., M.S., and Ph.D. degrees from the Toyohashi University of Technology (TUT), Japan. He joined TUT as a researcher in 2011. He was a researcher at Nagoya University, Japan, from 2011 to 2012, and an assistant professor at the Nagoya Institute of Technology, Japan, from 2012 to 2015. He was an assistant professor at Keio University, Japan, from 2015 to 2017. He was a researcher at Tokushima University, Japan, from 2017 to 2018, and has been an associate professor there since 2018. His research interests include spoken dialog systems and spoken language information processing. He is a senior member of the IEEE, and a member of the Institute of Electronics, Information and Communication Engineers (IEICE), the Information Processing Society of Japan (IPSJ), the Acoustical Society of Japan (ASJ), the Japanese Society for Artificial Intelligence (JSAI), and Phonetic Society of Japan (PSJ).

Norihide Kitaoka received his B.S. and M.S. degrees from Kyoto University, Japan. In 1994, he joined the DENSO Corporation. In 2000, he received his Ph.D. degree from the Toyohashi University of Technology (TUT), Japan. He joined TUT as a research associate in 2001 and was a lecturer from 2003 to 2006. He was an associate professor at Nagoya University, Japan, from 2006 to 2014, and joined Tokushima University, Japan, as a professor in 2014. He has been a professor at TUT since 2018. His research interests include speech processing, speech recognition, and spoken dialog systems. He is a member of the IEEE, the International Speech Communication Association (ISCA), the Institute of Electronics, the Information and Communication Engineers (IEICE), the Information Processing Society of Japan (IPSJ), the Acoustical Society of Japan (ASJ), the Japanese Society for Artificial Intelligence (JSAI), and the Association for Natural Language Processing.

https://doi.org/10.1017/ATSIP.2020.23 Published online by Cambridge University Press