INTERSPEECH 2020
October 25–29, 2020, Shanghai, China

Transformer-based Long-context End-to-end Speech Recognition

Takaaki Hori, Niko Moritz, Chiori Hori, Jonathan Le Roux

Mitsubishi Electric Research Laboratories (MERL), USA
{thori, moritz, chori, leroux}@merl.com

Abstract

This paper presents an approach to long-context end-to-end automatic speech recognition (ASR) using Transformers, aiming at improving ASR accuracy for long audio recordings such as lecture and conversational speeches. Most end-to-end ASR systems are basically designed to recognize independent utterances, but contextual information (e.g., speaker or topic) over multiple utterances is known to be useful for ASR. There are some prior studies on RNN-based models that utilize such contextual information, but very few on Transformers, which are becoming more popular in end-to-end ASR. In this paper, we propose a Transformer-based architecture that accepts multiple consecutive utterances at the same time and predicts an output sequence for the last utterance. This is repeated in a sliding-window fashion with one-utterance shifts to recognize the entire recording. Based on this framework, we also investigate how to design the context window and train the model effectively in monologue (one speaker) and dialogue (two speakers) scenarios. We demonstrate the effectiveness of our approach using monologue benchmarks on CSJ and TED-LIUM3 and dialogue benchmarks on SWITCHBOARD and HKUST, showing significant error reduction from single-utterance ASR baselines with or without speaker i-vectors.

Index Terms: end-to-end speech recognition, transformer, long context ASR

1. Introduction

Recent advancement of deep learning technology has opened a new paradigm for automatic speech recognition (ASR), the so-called end-to-end ASR, which consists in training and using a single deep network that directly converts a speech signal or its features into target text. Unlike typical hybrid ASR systems, end-to-end systems typically rely only on paired acoustic and language data without relying on linguistic knowledge, and the whole system is trained using a single algorithm. This approach makes it thus feasible to build ASR systems without expert knowledge.

Several end-to-end models have been proposed and applied to ASR, such as connectionist temporal classification (CTC) [1], attention-based encoder decoder [2, 3], RNN Transducer (RNN-T) [4], Transformer [5], and their combinations [6–9]. Specifically, Transformer has recently provided significant performance gain over RNN-based models in most major sequence-to-sequence tasks including ASR [10, 11]. Therefore, Transformer-based approaches are being studied with increasing attention to further improve ASR accuracy.

However, most end-to-end ASR systems are basically designed to recognize independent utterances, despite the fact that contextual information over multiple utterances, such as information on the speaker or topic, is known to be useful for ASR. There are several approaches to incorporating contextual information in end-to-end ASR, such as i-vector approaches that utilize speaker context [12–15] and hierarchical RNN decoders that utilize discourse context [16, 17]. Besides, RNN-T and attention models have been applied to long-form speech recognition [18], where the authors mainly focused on the scalability to long-form speeches in the inference phase. Thus, no effective methods have been reported yet for Transformer ASR. In this work, we extend the Transformer model to incorporate contextual information in training and decoding to improve the recognition accuracy for long audio recordings such as lecture and conversational speeches. The proposed method simply concatenates multiple utterances and trains a Transformer to recognize the last of these utterances. The previous utterances can thus be used to normalize or adapt the acoustic and linguistic features at every encoder/decoder layer for recognizing the last utterance.

Our decoding strategy is based on a sliding window with one-utterance shifts to recognize long audio, where utterance boundaries are given manually or by voice activity detection. It is also possible to decode multiple utterances at the same time rather than the last utterance, but this can lead to larger processing delay than that of the sliding window approach. In this study, we assume that at most one utterance delay is allowed and future context is not available. This is reasonable for typical (non-streaming) end-to-end ASR[1].

We also investigate how to design the context window and train the model effectively in monologue and dialogue scenarios. We demonstrate the effectiveness of our approach using monologue benchmarks (speech from one speaker) on CSJ [20] and TED-LIUM3 [21], and dialogue benchmarks (speech from two speakers) on SWITCHBOARD [22] and HKUST [23], comparing it with single-utterance ASR baselines with and without speaker i-vectors.

2. Related work

There are some prior studies on Transformers that can utilize long contextual information for natural language processing (NLP) tasks. Transformer-XL [24] is an extension of Transformer language model (LM), which learns dependency beyond a fixed-length context by reusing hidden vectors of the previous segment computed together with further previous segments. Compressive Transformers [25] can further incorporate longer context using a compressive memory. These methods are analogous with our approach in terms of segment level processing. However, since these approaches assume that every segment has the same size, it is not easy to directly apply them to encoder-decoder architectures for ASR, which need to handle variable-length utterances consisting of speech/transcript pairs.

Moreover, several extended Transformers have been proposed, which enable to accept an entire document or a full video stream, e.g., Hierarchical Transformer [26] for document sum-

[1]The proposed method could be extended to streaming ASR with our triggered attention technique for Transformers [19]. This will be studied in future work.

http://dx.doi.org/10.21437/Interspeech.2020-2928

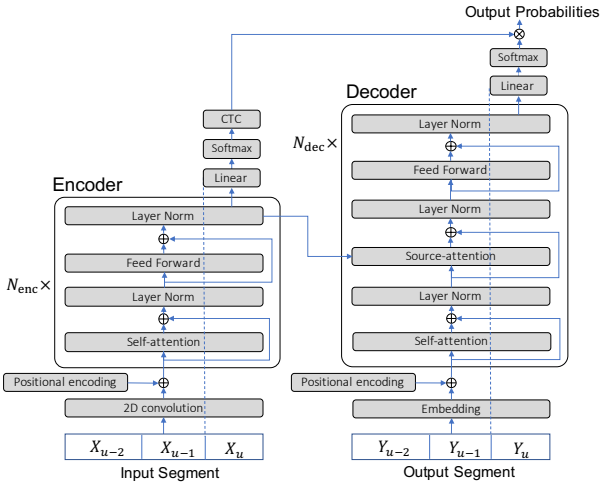

Figure 1: *Context-expanded Transformer.*

marization, Longformer [27] for document LM, and Masked Transformer [28] for dense video captioning. These models are trained to extract global contextual information throughout the long sequence using the self-attention mechanism. Our approach also aims at utilizing global information over multiple utterances for ASR. To the best of our knowledge, this type of Transformers have not yet been investigated for ASR.

Speaker-aware Speech-Transformer [14] incorporates speaker i-vectors into Transformers. The i-vector contains global information of the speaker, but this represents only input context. Besides, one recent study has reported efficacy of RNN and Transformer LMs trained with longer segments, but it considers only output context in a hybrid ASR system [29]. Our approach considers both input and output contexts in an end-to-end manner, and does not require preliminary steps such as i-vector generation.

3. Context-expanded Transformer

Transformer [5] consists of encoder and decoder networks, which have deep feed-forward architectures including repeated blocks of self-attention and feed-forward layers with residual connections [30] and layer normalization [31]. The decoder network also features a source attention layer in each block to read the encoder's output. Unlike RNN-based models, Transformer does not have recurrent connections, and therefore it yields stable and fast optimization. Moreover, self-attention and source attention mechanisms effectively utilize interdependence between input frames and output tokens to achieve high-accuracy sequence-to-sequence mapping.

Figure 1 illustrates the context-expanded Transformer proposed in this paper. The network architecture is basically the same as the original Transformer, but it accepts multiple utterances at once and predicts output tokens for the last utterance using the previous utterances as contextual information.

Given a feature sequence X_u for a u-th utterance and feature sequences $X_v, \dots, X_{u-1}$ for its previous utterances, where $1 \leq v < u$, we denote the input speech segment as $X_{v:u} = (X_v, \dots, X_{u-1}, X_u)$ and its corresponding output segment as $Y_{v:u} = (Y_v, \dots, Y_{u-1}, Y_u)$. The goal of ASR here is to find the most probable token sequence $\hat{Y}_u$ for the u-th utterance as

$$\hat{Y}_u = \underset{Y_u \in \mathcal{V}^*}{\operatorname{argmax}} p(Y_u | Y_{v:u-1}, X_{v:u}) = \underset{Y_u = y_{u,1:L} \in \mathcal{V}^*}{\operatorname{argmax}} \prod_{i=1}^{L} p(y_{u,i} | Y_{v:u-1}, y_{u,1:i-1}, X_{v:u}), \quad (1)$$

where $y_{u,1:L}$ denotes the token sequence $(y_{u,1}, \dots, y_{u,L})$ of Y_u, and $\mathcal{V}$ is the vocabulary.

The probability of $y_{u,i}$ in Eq. (1) is computed using the Transformer. The encoder first applies 2D convolution (Conv2D) and positional encoding (PosEnc) to all frames of $X_{v:u}$ and adds them to obtain the first hidden vector sequence

$$H^0_{v:u} = \text{Conv2D}(X_{v:u}) + \text{PosEnc}(X_{v:u}). \quad (2)$$

Then, it computes a hidden vector sequence in each encoder block as

$$\bar{H}^n_{v:u} = \xi(H^{n-1}_{v:u} + \text{MHA}(H^{n-1}_{v:u}, H^{n-1}_{v:u}, H^{n-1}_{v:u})) \quad (3)$$

$$H^n_{v:u} = \xi(\bar{H}^n_{v:u} + \text{FFN}(\bar{H}^n_{v:u})), \quad (4)$$

where $\text{MHA}(\cdot,\cdot,\cdot)$, $\text{FFN}(\cdot)$, and $\xi(\cdot)$ represent multi-head attention, feed-forward network, and layer normalization, respectively. $\text{MHA}()$ takes three arguments Q, K, and V, which are query, key, and value vector sequences [5]. For self-attention in the encoder, these arguments are equal to $H^{n-1}_{v:u}$. The encoder states are obtained as the output of the last block, i.e., $H^{N_{\text{enc}}}_{v:u}$, where N_{enc} denotes the number of encoder blocks.

The decoder accepts previous output sequence $(Y_{v:u-1}, y_{u,1:i-1})$ and the encoder states $H^{N_{\text{enc}}}_{v:u}$, and estimates the probability distribution of $y_{u,i}$ in Eq. (1). For simplicity, we rewrite $(Y_{v:u-1}, y_{u,1:i-1})$ as $y'_{u,1:k-1}$, which represents all previous tokens to index $k-1$ in the segment, where $|Y_{v:u-1}| < k \leq |Y_{v:u}|$ and $k = |Y_{v:u-1}| + i$, $|Y_*|$ denoting the number of tokens in sequence Y_*.

The decoder first applies token embedding and positional encoding as

$$g^0_{u,1:k-1} = \text{Embed}(y'_{u,1:k-1}) + \text{PosEnc}(y'_{u,1:k-1}), \quad (5)$$

where $\text{Embed}(\cdot)$ represents the token embedding. Next, the decoder computes hidden vector $g^{v:u,n}_{k-1}$ in each n-th block as

$$\bar{g}^n_{u,k-1} = \xi(g^{n-1}_{u,k-1} + \text{MHA}(g^{n-1}_{u,k-1}, g^{n-1}_{u,1:k-1}, g^{n-1}_{u,1:k-1})) \quad (6)$$

$$\bar{\bar{g}}^n_{u,k-1} = \xi(\bar{g}^n_{u,k-1} + \text{MHA}(\bar{g}^n_{u,k-1}, H^{N_{\text{enc}}}_{v:u}, H^{N_{\text{enc}}}_{v:u})) \quad (7)$$

$$g^n_{u,k-1} = \xi(\bar{\bar{g}}^n_{u,k-1} + \text{FFN}(\bar{\bar{g}}^n_{u,k-1})), \quad (8)$$

and outputs the decoder state from the last block, i.e., $g^{N_{\text{dec}}}_{u,k-1}$, where N_{dec} denotes the number of decoder blocks. Eq. (7) applies source attention over the encoder states, in which $\bar{g}^n_{u,k-1}$ is used for the query vector. Finally, we obtain the Transformer token probability distribution by applying a linear transformation and a softmax function as

$$p_{\text{trs}}(y_{u,i} | Y_{v:u-1}, y_{u,1:i-1}, X_{v:u}) = \text{Softmax}(\text{Linear}(g^{N_{\text{dec}}}_{u,|Y_{v:u-1}|+i-1})). \quad (9)$$

We can also utilize CTC in training and decoding similarly to the CTC-Attention approach in RNN-based architectures [6, 8, 32]. The CTC sequence probability can be computed as

$$p_{\text{ctc}}(Y_u | X_{v:u}) = \text{CTC}(\text{Softmax}(\text{Linear}(H^{N_{\text{enc}}}_u)), Y_u), \quad (10)$$

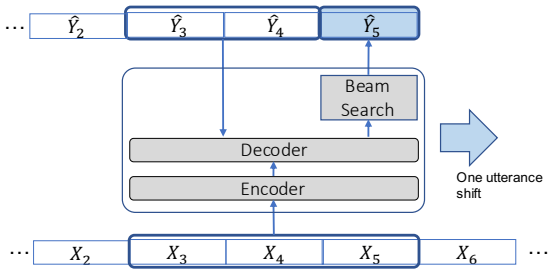

Figure 2: *Sliding-window decoding.*

where $\mathrm{CTC}(P, Y)$ is an operation that marginalizes the posterior probabilities over all possible alignments between P and Y using the forward-backward algorithm [1].

For training, we use the CTC-attention loss computed as

$$\mathcal{L}_u = -\alpha \log p_{\mathrm{trs}}(Y_u^*|Y_{v:u-1}^*, X_{v:u}) - (1-\alpha)\log p_{\mathrm{ctc}}(Y_u^*|X_{v:u}), \quad (11)$$

where Y_u^* and $Y_{v:u-1}^*$ are groundtruth transcripts, and α is a scaling factor to balance the Transformer cross-entropy and CTC losses.

For decoding, we combine Transformer, CTC, and optionally LM scores to find the best hypothesis as

$$\hat{Y}_u = \underset{Y_u \in \mathcal{V}^*}{\mathrm{argmax}}\{\lambda \log p_{\mathrm{trs}}(Y_u|Y_{v:u-1}, X_{v:u}) + (1-\lambda)\log p_{\mathrm{ctc}}(Y_u|X_{v:u}) + \gamma \log p_{\mathrm{lm}}(Y_u)\}, \quad (12)$$

where λ and γ are scaling factors to balance the model scores. Similarly to prior studies, we employ output-synchronous beam search to efficiently find the best hypothesis [7]. In Eq. (12), we can choose a context-dependent LM in a form of $p_{\mathrm{lm}}(Y_u|Y_{1:u-1})$. With an RNN-LM, contextual information beyond one utterance can be used by passing the state information from the previous utterance.

To recognize long audio such as lecture and conversational speeches, we repeat the decoding process of Eq. (12) in a sliding-window fashion with one-utterance shifts. Figure 2 shows the sliding-window-based recognition process. The system decodes the first utterance X_1 without relying on previous context, and outputs the first hypothesis $\hat{Y}_1$. For the second utterance X_2, the system reads input segment $X_{1:2}$, decodes X_2 conditioned on the previous input context X_1 and output context $\hat{Y}_1$, and outputs the second hypothesis $\hat{Y}_2$. In this way, the system recognizes utterances along a long audio input. Due to the limitation of time and memory space, we reduce the segment size by truncating the oldest utterances in the segment if the segment duration exceeds a pre-defined constant `MaxSegmentLength`. When truncating the input segment, we also truncate the corresponding output context. The example in Fig. 2 shows the case decoding hypothesis $\hat{Y}_5$ for input segment $X_{3:5}$ and output context $\hat{Y}_{3:4}$.

We also consider various ways to construct the input segment and the output context. Typically, only a single speaker is included in each segment of lecture speech. However, different speakers can be included in the segment of conversational speech. In segment $X_{3:5}$, X_3 can be spoken by speaker A, X_4 by speaker B, and X_5 by speaker C. With our approach, speech features from different speakers may have a negative impact on the Transformer in terms of speaker adaptation. To avoid this kind of segments, we propose to make each segment with only a single speaker. For example, if X_1, X_3, and X_5 are spoken by the same speaker, the input segment is constructed as (X_1, X_3, X_5) to recognize X_5. The output context can also be made as $(\hat{Y}_1, \hat{Y}_3)$, or just $(\hat{Y}_3, \hat{Y}_4)$ based on the original manner to avoid disconnecting the conversation context. Depending on whether speaker information is used or not, we refer to these input and output contexts as speaker-dependent (SD) context and speaker-independent (SI) context, respectively. The same manner is also applied in the training phase.

To construct SD segments, we need to know the speaker id of each utterance. This is one limitation of the SD approach, but we can typically obtain each speaker's utterances from a channel associated with the speaker in telephone conversations or in meetings recorded by worn microphones. Speaker diarization techniques are also available to identify speakers when they are recorded with distant microphones.

Table 1: *Corpus information.*

dataset	lang.	type	hours	test sets
CSJ	ja	monolog	581	eval1 / eval2 / eval3
TED-LIUM3	en	monolog	452	dev / test
SWITCHBOARD	en	dialog	260	(eval2000) callhm / swbd
HKUST	zh	dialog	200	train_dev / dev

4. Experiments

4.1. Conditions

We conducted several experiments using monologue benchmarks on CSJ [20] and TED-LIUM3 [21] corpora and dialogue benchmarks on SWITCHBOARD [22] and HKUST [23] corpora. Table 1 summarizes the information of the data sets. The Kaldi toolkit [33] was used to extract 80-dimensional log mel-filter bank acoustic features plus three-dimensional pitch features. We trained Transformers with the architecture in ESPnet [11, 34]. The encoder had one Conv2D module followed by 12 encoder blocks ($N_{\mathrm{enc}} = 12$). The Conv2D included a 2-layer 2D-CNN with 256 channels, a kernel size of 3×3, a stride of size 2, and ReLU activation, which outputs a 256 dimensional vector sequence with the reduced utterance length by a factor of 4. We employed multi-head attention with 4 heads of 256 dimensions. The feed-forward network had one hidden layer with 2,048 units and ReLU non-linearity. The decoder had a token embedding layer followed by 6 decoder blocks ($N_{\mathrm{dec}} = 6$). The self-attention, source attention, and feed forward layers in the decoder had the same dimensions as those in the encoder. The output dimension was dependent on the number of unique tokens in the task. There were 3,260 characters in CSJ, 652 word pieces in TED-LIUM3, 1,996 word pieces in SWITCHBOARD, and 3,653 characters in HKUST.

All models were trained using the end-to-end speech processing toolkit ESPnet [34]. We generally followed the default configuration of each task in ESPnet recipes, where speed perturbation was applied for all the data sets we used, but SpecAugment was applied only for TED-LIUM3 and SWITCHBOARD. We also trained models without speed perturbation to reduce the training time and compare diverse conditions for CSJ and TED-LIUM3, since those data sets were larger than the others. Baseline Transformers were trained with independent utterances without context. Furthermore, i-vector-based speaker adaptation was also tested for comparison, where a speaker i-vector is concatenated to each frame vector right before the first encoder block. To train Transformers with the proposed method, we expanded each utterance to a segment by concatenating it

Table 2: *Monologue ASR results. Numbers indicate CERs [%] for CSJ and WERs [%] for TED-LIUM3.*

	Speed perturb.	CSJ eval1	CSJ eval2	CSJ eval3	TED-LIUM3 dev	TED-LIUM3 test
Baseline		6.0	4.2	4.7	11.9	8.7
+ i-vector		5.9	4.1	4.9	11.2	8.6
Proposed		**5.5**	**3.8**	**4.0**	**10.5**	**8.1**
ESPnet [11]	✓	5.7	4.1	4.5	9.7	8.0
Proposed	✓	**5.3**	**3.6**	**3.8**	**9.2**	**7.5**

Table 3: *Dialogue ASR results. Numbers indicate WERs [%] for SWITCHBOARD and CERs [%] for HKUST.*

	context in.	context out.	SWITCHBOARD callhm	SWITCHBOARD swbd	HKUST train_dev	HKUST dev
Baseline			17.7	8.9	23.9	23.6
+i-vector	SD		17.8	8.8	-	-
ESPnet [11]			18.1	9.0	-	23.5
Proposed	SI	SI	15.6	8.4	22.8	22.5
	SD	SI	15.4	**8.2**	22.5	22.1
	SD	SD	**15.3**	8.3	**22.2**	**21.5**

with previous utterances, where we set `MaxSegmentLength` to 20 seconds as a default setting.

Furthermore, we mainly used segment loss $\mathcal{L}_{v:u} = -\alpha \log p_{\text{trs}}(Y^*_{v:u}|X_{v:u}) - (1-\alpha)\log p_{\text{ctc}}(Y^*_{v:u}|X_{v:u})$ instead of $\mathcal{L}_u$ in Eq. (11) except when we use different contexts, i.e., SD and SI contexts for the encoder and the decoder. This is because we found that the segment loss made the convergence faster in our preliminary experiment. However, we always used $\mathcal{L}_u$ to compute the validation loss. With either loss, the ASR performance reaches the same level, but we can quit training earlier with the segment loss. Detailed investigation on the loss functions will be considered in future work.

Finally, we averaged the top 5 models based on validation loss for recognition. We also trained LSTM-based RNN-LMs using transcripts for CSJ and HKUST, and external text data for TED-LIUM3. The transcripts were concatenated in the same manner as in the context-expanded Transformer training. No LM was used for SWITCHBOARD. ASR performance is measured by character error rate (CER) or word error rate (WER).

4.2. Results

Table 2 shows monologue ASR results on CSJ and TED-LIUM3 data sets. For every data set, the proposed approach significantly reduces ASR errors from the baseline, where the relative error rate reduction ranges from 5 to 15%. The i-vector approach did not consistently reduce the error rate. This may be because Transformer already has a certain ability to normalize speaker variability. A prior study has reported that i-vectors provided only 1.7% relative error reduction in Chinese ASR [14]. In addition, we confirm that our approach outperforms the best performance with ESPnet [11] with speed perturbation.

Table 3 shows dialogue ASR results on SWITCHBOARD and HKUST data sets. For every data set, the proposed approach significantly reduces ASR errors from the baseline, especially with SD context, where the relative error rate reduction ranges from 7 to 13.5%. We also confirm that SD context is better than SI context in dialogue ASR.

4.3. Analysis

We investigate the impact of segment length on the recognition performance. Figure 3 shows the relationships between

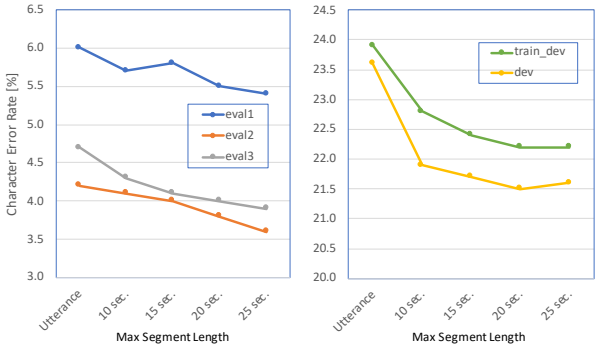

Figure 3: *Maximum segment length vs. CER in CSJ & HKUST.*

Table 4: *Efficacy of input/output/LM context information. Numbers indicate CERs [%].*

	Use context: training in.	training out.	decoding in.	decoding out.	decoding LM	CSJ eval1	HKUST dev
Baseline						6.0	23.6
Proposed	✓	✓	✓	✓	✓	**5.5**	**21.5**
	✓	✓	✓	✓		5.6	21.8
	✓	✓	✓			5.7	22.2
	✓	✓		✓	✓	8.0	25.1
	✓	✓				6.3	25.1
	✓		✓			5.7	22.5
		✓		✓	✓	5.8	23.0

`MaxSegmentLength` and CER in CSJ and HKUST tasks. We can see that as the segment length becomes longer, the error rate reduces further, and 20 s is sufficient to achieve the best performance for HKUST, but there is a potential to further reduce the error with segments longer than 25 s for CSJ.

We further investigate the importance of input, output, and LM contexts. The results are summarized in Table 4. The best performance 5.5% in CSJ and 21.5% in HKUST is obtained when we use all the contexts. By removing the LM context, the error rates slightly increase to 5.6% and 21.8%. As we also remove the output context, the error rates further increase to 5.7% and 22.2%. On the other hand, when removing input context, the error rates get worse than the baseline. By training the models from scratch without input or output context, the error rates are between the baseline and best CERs, which looks reasonable. Thus, these results demonstrate that our approach can effectively incorporate input and output contexts into Transformer-based ASR.

5. Conclusions

In this paper, we have proposed an approach to long-context end-to-end ASR using Transformers for long audio recordings such as lecture and conversational speeches. The proposed Transformer accepts multiple consecutive utterances at the same time and predicts an output sequence for the last utterance. This is repeated in a sliding-window fashion with one-utterance shifts to recognize the entire recording. We have demonstrated the effectiveness of our approach using monologue benchmarks on CSJ and TED-LIUM3 and dialogue benchmarks on SWITCHBOARD and HKUST, achieving 5-15% relative error reduction from utterance-based ASR baselines. Future work will include reduction of computational complexity and memory usage to utilize longer contextual information with small overhead.

6. References

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

[2] D. Bahdanau, K. Cho, and Y. Bengio, "Neural machine translation by jointly learning to align and translate," *arXiv preprint arXiv:1409.0473*, 2014.

[3] W. Chan, N. Jaitly, Q. V. Le, and O. Vinyals, "Listen, attend and spell: A neural network for large vocabulary conversational speech recognition," in *Proc. IEEE ICASSP*, 2015.

[4] A. Graves, A.-r. Mohamed, and G. Hinton, "Speech recognition with deep recurrent neural networks," in *Proc. IEEE ICASSP*, May 2013, pp. 6645–6649.

[5] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, "Attention is all you need," in *Proc. NIPS*, Dec. 2017, pp. 5998–6008.

[6] S. Kim, T. Hori, and S. Watanabe, "Joint CTC-attention based end-to-end speech recognition using multi-task learning," in *Proc. IEEE ICASSP*, Mar. 2017, pp. 4835–4839.

[7] T. Hori, S. Watanabe, Y. Zhang, and W. Chan, "Advances in joint CTC-attention based end-to-end speech recognition with a deep CNN encoder and RNN-LM," in *Proc. ISCA Interspeech*, Aug. 2017.

[8] S. Karita, N. E. Y. Soplin, S. Watanabe, M. Delcroix, A. Ogawa, and T. Nakatani, "Improving transformer-based end-to-end speech recognition with connectionist temporal classification and language model integration," in *Proc. ISCA Interspeech*, Sep. 2019, pp. 1408–1412.

[9] Q. Zhang, H. Lu, H. Sak, A. Tripathi, E. McDermott, S. Koo, and S. Kumar, "Transformer transducer: A streamable speech recognition model with transformer encoders and RNN-T loss," in *Proc. IEEE ICASSP*, May 2020, pp. 7829–7833.

[10] L. Dong, S. Xu, and B. Xu, "Speech-transformer: a no-recurrence sequence-to-sequence model for speech recognition," in *Proc. IEEE ICASSP*, Apr. 2018, pp. 5884–5888.

[11] S. Karita, N. Chen, T. Hayashi, T. Hori, H. Inaguma, Z. Jiang, M. Someki, N. E. Y. Soplin, R. Yamamoto, X. Wang, S. Watanabe, T. Yoshimura, and W. Zhang, "A comparative study on transformer vs RNN in speech applications," in *Proc. IEEE ASRU*, Dec. 2019.

[12] N. Dehak, P. J. Kenny, R. Dehak, P. Dumouchel, and P. Ouellet, "Front-end factor analysis for speaker verification," *IEEE Trans. Audio, Speech, Language Process.*, vol. 19, no. 4, pp. 788–798, 2010.

[13] K. Audhkhasi, B. Ramabhadran, G. Saon, M. Picheny, and D. Nahamoo, "Direct acoustics-to-word models for English conversational speech recognition," *arXiv preprint arXiv:1703.07754*, 2017.

[14] Z. Fan, J. Li, S. Zhou, and B. Xu, "Speaker-aware speech-transformer," in *Proc. IEEE ASRU*, Dec. 2019, pp. 222–229.

[15] L. Sarı, N. Moritz, T. Hori, and J. Le Roux, "Unsupervised speaker adaptation using attention-based speaker memory for end-to-end ASR," in *Proc. IEEE ICASSP*, May 2020, pp. 7384–7388.

[16] S. Kim and F. Metze, "Dialog-context aware end-to-end speech recognition," in *Proc. IEEE SLT*, Dec. 2018, pp. 434–440.

[17] R. Masumura, T. Tanaka, T. Moriya, Y. Shinohara, T. Oba, and Y. Aono, "Large context end-to-end automatic speech recognition via extension of hierarchical recurrent encoder-decoder models," in *Proc. IEEE ICASSP*, May 2019, pp. 5661–5665.

[18] C.-C. Chiu, W. Han, Y. Zhang, R. Pang, S. Kishchenko, P. Nguyen, A. Narayanan, H. Liao, S. Zhang, A. Kannan *et al.*, "A comparison of end-to-end models for long-form speech recognition," *arXiv preprint arXiv:1911.02242*, 2019.

[19] N. Moritz, T. Hori, and J. Le Roux, "Streaming automatic speech recognition with the transformer model," in *Proc. IEEE ICASSP*, May 2020.

[20] K. Maekawa, H. Koiso, S. Furui, and H. Isahara, "Spontaneous speech corpus of Japanese," in *Proc. LREC*, vol. 2, 2000, pp. 947–952.

[21] F. Hernandez, V. Nguyen, S. Ghannay, N. Tomashenko, Y. Esteve, O. Jokisch, and R. Potapova, "TED-LIUM 3: Twice as much data and corpus repartition for experiments on speaker adaptation," in *Speech and Computer*, 2018, pp. 198–208.

[22] J. Godfrey, E. Holliman, and J. McDaniel, "SWITCHBOARD: telephone speech corpus for research and development," in *Proc. IEEE ICASSP*, vol. 1, 1992, pp. 517–520.

[23] Y. Liu, P. Fung, Y. Yang, C. Cieri, S. Huang, and D. Graff, "HKUST/MTS: A very large scale mandarin telephone speech corpus," in *Chinese Spoken Language Processing*. Springer, 2006, pp. 724–735.

[24] Z. Dai, Z. Yang, Y. Yang, J. G. Carbonell, Q. Le, and R. Salakhutdinov, "Transformer-XL: Attentive language models beyond a fixed-length context," in *Proc. ACL*, Jul. 2019, pp. 2978–2988.

[25] J. W. Rae, A. Potapenko, S. M. Jayakumar, and T. P. Lillicrap, "Compressive transformers for long-range sequence modelling," *arXiv preprint arXiv:1911.05507*, 2019.

[26] Y. Liu and M. Lapata, "Hierarchical transformers for multi-document summarization," in *Proc. ACL*, Jul. 2019, pp. 5070–5081.

[27] I. Beltagy, M. E. Peters, and A. Cohan, "Longformer: The long-document transformer," *arXiv preprint arXiv:2004.05150*, 2020.

[28] L. Zhou, Y. Zhou, J. J. Corso, R. Socher, and C. Xiong, "End-to-end dense video captioning with masked transformer," in *Proc. IEEE CVPR*, Jun. 2018, pp. 8739–8748.

[29] K. Irie, A. Zeyer, R. Schlüter, and H. Ney, "Training language models for long-span cross-sentence evaluation," in *Proc. IEEE ASRU*, Dec. 2019, pp. 419–426.

[30] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proc. IEEE CVPR*, Jun. 2016, pp. 770–778.

[31] J. L. Ba, J. R. Kiros, and G. E. Hinton, "Layer normalization," *arXiv preprint arXiv:1607.06450*, 2016.

[32] T. Hori, S. Watanabe, and J. R. Hershey, "Joint CTC/attention decoding for end-to-end speech recognition," in *Proc. ACL*, Jul. 2017.

[33] D. Povey, A. Ghoshal, G. Boulianne, L. Burget, O. Glembek, N. Goel, M. Hannemann, P. Motlicek, Y. Qian, P. Schwarz, J. Silovsky, G. Stemmer, and K. Vesely, "The Kaldi speech recognition toolkit," in *Proc. IEEE ASRU*, Dec. 2011.

[34] S. Watanabe, T. Hori, S. Karita, T. Hayashi, J. Nishitoba, Y. Unno, N.-E. Y. Soplin, J. Heymann, M. Wiesner, N. Chen *et al.*, "ESPnet: End-to-end speech processing toolkit," in *Proc. ISCA Interspeech*, Sep. 2018.