INTERSPEECH 2021
30 August – 3 September, 2021, Brno, Czechia

Online Compressive Transformer for End-to-End Speech Recognition

Chi-Hang Leong, Yu-Han Huang, Jen-Tzung Chien

Dept of Electrical and Computer Engineering, National Yang Ming Chiao Tung University, Taiwan
{harryleong.eed07g,yhhuang.eed08g,jtchien}@nctu.edu.tw

Abstract

Traditionally, transformer with connectionist temporal classification (CTC) was developed for offline speech recognition where the transcription was generated after the whole utterance has been spoken. However, it is crucial to carry out online transcription of speech signal for many applications including live broadcasting and meeting. This paper presents an online transformer for real-time speech recognition where online transcription is generated chunk by chuck. In particular, an online compressive transformer (OCT) is proposed for end-to-end speech recognition. This OCT aims to generate immediate transcription for each audio chunk while the comparable performance with offline speech recognition can be still achieved. In the implementation, OCT tightly combines with both CTC and recurrent neural network transducer by minimizing their losses for training. In addition, this OCT systematically merges with compressive memory to reduce potential performance degradation due to online processing. This degradation is caused by online transcription which is generated by the chunks without history information. The experiments on speech recognition show that OCT does not only obtain comparable performance with offline transformer, but also work faster than the baseline model.

Index Terms: Online processing and learning, compressive transformer, end-to-end speech recognition

1. Introduction

In the past few year, end-to-end (E2E) automatic speech recognition (ASR) have been successfully developed for real-world applications. E2E ASR models basically transfer a sequence of preprocessed acoustic features into a sequence of word tokens directly without the need of training the acoustic, pronunciation and language models separately in traditional methods. E2E ASR has been popularly built on the basis of transformer [1] which is known as an attention-based encoder/decoder framework [2]. This framework achieved state-of-the-art performance in various speech and natural language applications. In general, transformer is efficient to capture a large span of temporal information, and performs better than long short-term memory (LSTM) [3] for E2E ASR [4]. However, low convergence speed causes the transformer hard to train. To cope with this problem, transformer was combined with the connectionist temporal classification (CTC) [5] to carry out joint training and decoding. Typically, CTC is effective to learn the alignments between speech signals and word tokens. However, CTC is weak to characterize temporal information in output sequence. Also, CTC could not handle the task that the length of input sequence is shorter than that of output sequence. Therefore, the recurrent neural network transduer (RNN-T) [6] was proposed to capture the dependencies between outputs based on a prediction network. In addition, although the transformer achieved desirable performance in ASR systems, this model was previously built in an offline manner which was *not* suitable for real-time processing. This weakness is caused since the encoder of conventional transformer required to compute attention weights over the whole input frames. Recently, some related works were developed for online ASR which was formed by either combining transformer with RNN-T [7, 8, 9, 10, 11] or improving the structure of transformer [12, 13, 14, 15]. Nevertheless, most of previous methods have a common issue that the long-term dependencies in a speech signal are loosely represented. The dependencies were only modeled for the relations between adjacent chunks of speech or text.

This paper proposes an online compressive transformer (OCT) for real-time speech recognition similar to neural transducer [16, 17, 18] which attains comparable performance with offline transformer with CTC. This OCT combines the benefit of original transformer with CTC and RNN-T. In particular, we implement the compressive transformer [19] and integrate it into OCT encoder to capture long-term dependencies. Each encoder layer preserves major information from previous chunks through compressive memory. To further improve recognition performance, an CTC loss is minimized to learn encoder outputs and a forward-backward algorithm is utilized to optimize all possible paths of prediction sequence similar to RNN-T [6]. The one-step constrained beam search [20] is applied for inference. This algorithm further controls the number of expansion symbols on each chunk and calculates the probability of candidate hypotheses in parallel. The inference time can be significantly reduced with desirable performance. In the experiments, this OCT model is implemented and evaluated for end-to-end speech recognition. The resulting solution performs as well as that using offline transformer. Attractively, the proposed OCT is inferred chunk by chunk in a real-time manner.

2. Background Survey

2.1. Compressive transformer

Compressive transformer (CT) [19] was proposed for large-span language modeling [21]. CT referred to the Transformer-XL [22] which stored the previous information in memory at each layer n so as to represent a longer history of context. The key idea was to compress the old memory M_n^{old} which is the set of previous text embedding copies, and save it in an additional compressed memory $\widehat{M}_n^{\text{new}}$ instead of discarding them directly where $\widehat{M}_n^{\text{new}} = f^c(M_n^{\text{old}}; \theta_c)$ with parameter θ_c. Different compressive functions $f^c(\cdot)$ were proposed to compress the old memory including the max/mean pooling, dilated convolution and one-dimensional (1d) convolution. Both the memories $\{M_n^{\text{old}}, \widehat{M}_n^{\text{new}}\}$ at each layer n were used to save previous context. The compression function using 1d-convolution performed well and is adopted in this study in a form of $f^c(H; \theta_c) \triangleq \text{Conv1D}(H)$ where H is an input embedding. With the output after attention operation $\text{attn}(H, M) \triangleq \text{Softmax}(HW^q(MW^k)^\top)MW^v$, calculated by using the encoder matrices θ_e of query W^q, key W^k and value W^v, CT is

http://dx.doi.org/10.21437/Interspeech.2021-545

optimized by minimizing the attention reconstruction loss $\mathcal{L}_{\text{attn}}$ over N layers with respect to θ_c, W^q, W^k and W^v where

$$\mathcal{L}_{\text{attn}} = \sum_{n=1}^{N} \| \text{attn}(H_n, M_n^{\text{old}}) - \text{attn}(H_n, \widehat{M}_n^{\text{new}}) \|^2. \quad (1)$$

CT was only exploited for language modeling with transformer encoder, and has not been investigated for full transformer architecture and E2E speech recognition. RNN-T is required.

2.2. Recurrent neural network transducer

RNN-T [6, 23] introduced a transcription network and a prediction network for sequence transduction in E2E ASR. Transcription and prediction networks were individual recurrent neural networks which were used to encoder input sequence $\mathbf{x}$ and output sequence $\mathbf{y}$, respectively. The outputs of two networks were combined to predict the word tokens $\mathbf{y}$. Importantly, a forward-backward algorithm was derived to calculate the conditional likelihood $p(\mathbf{y}|\mathbf{x})$ where the dependencies between individual samples in input and output were represented. Synchronous transformer in [9] was accordingly developed for speech recognition. In the implementation, the forward variable $\alpha(i,j)$ was calculated as the probability of j output tokens $\mathbf{y}_{1:j}$ produced by i input chunks $\mathbf{x}_{1:i}$. Given the i^{th} chunk and the predicted word sequence $\mathbf{y}_{0:j-1}$, the probabilities of predicting blank token, denoted by $\langle \text{blk} \rangle$, and $\mathbf{y}_j$ are represented by $\phi(i, j-1)$ and $y_j(i, j-1)$, respectively. Let the start and end of sentence denoted by $\langle \text{SOS} \rangle$ and $\langle \text{EOS} \rangle$, respectively. Forward variable is obtained from $\langle \text{SOS} \rangle$ via a recursive path expressed by followed, $\alpha(i,j) = \alpha(i-1,j)\phi(i-1,j) + \alpha(i,j-1)y_j(i,j-1)$, where $\alpha(1,0) = 1$, $i \in [1, I]$, and $j \in [0, J]$, I and J denote the length of input and output sequences, respectively. The conditional probability of the whole sequences is calculated as the forward variable at the end of two sequences at $i = I$ and $j = J$, i.e. $p(\mathbf{y}|\mathbf{x}) = \alpha(I,J)\phi(I,J)$. The backward variable $\beta(i,j)$ is calculated in a backward manner from $\langle \text{EOS} \rangle$ to a lattice node (i,j), which is seen as the probability of outputs $\mathbf{y}_{j+1:J}$ produced by inputs $\mathbf{x}_{i:I}$, i.e. $\beta(i,j) = \beta(i+1,j)\phi(i,j) + \beta(i,j+1)y_j(i,j)$, where $\beta(I,J) = \phi(I,J)$. The probability $p(\mathbf{y}_{1:J}|\mathbf{x}_{1:I})$ is then calculated as the product of forward and backward variables over all of the nodes $\forall k \in [1, I+J]$ via the forward-backward algorithm. RNN-T loss $\mathcal{L}_{\text{rnnt}}$ is accordingly measured by

$$-\log p(\mathbf{y}_{1:J}|\mathbf{x}_{1:I}) = -\log \sum_{(i,j):i+j=k} \alpha(i,j)\beta(i,j) \quad (2)$$

In this work, transcription and prediction network are replaced by the transformer encoder θ_e with compressive memory θ_c and transformer decoder θ_d respectively. Hence, $\mathcal{L}_{\text{rnnt}}$ is minimized with respect to the model parameters θ_e, θ_c and θ_d similar to synchronous transformer [9] (denoted as Sync-Transformer).

3. Online Compressive Transformer

Figure 1 depicts the architecture of the proposed online CT (OCT) which is extended from standard transformer [1] consisting of encoder and decoder [24] with multi-head attention.

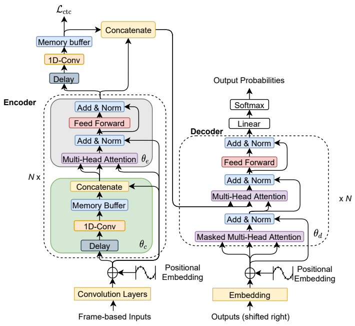


Figure 1: *Architecture of online compressive transformer.*

3.1. Model construction

The input speech signal X is formed as a sequence of 80-dimensional log-Mel filterbank vectors $X_{1:T}$. First, we subsample and transform it into a sequence of 256-dimensional feature vectors $S_{1:T'}$ by two layers of time-domain 2d-convolution [5] where T and T' denote the length of these two sequences. The sequence $S_{1:T'}$ is then spitted into I chunks $C_{1:I}$ where the length of each chunk is W and there is no overlapping between each chunk, i.e. $I = \frac{T'}{W}$. Next, each chunk in $C_{1:I}$ is added with the sinusoidal positional embedding individually to form the input embedding $H_{1,i}^x = C_i$ and passed it through N layers of encoder with $H_{n,i}^x$ at layer n. To enhance the dependency between different chunks, the compressive memory slot [19] at layer n, where $n \in [1, N+1]$ and N is the number of encoder and decoder layers, and chunk i is encoded and augmented as

$$\widehat{M}_{n,i} = \text{Cat}(\widehat{M}_{n,i-1}, f^c(H_{n,i-1}^x))_{[-n_{\widehat{M}}:]}$$

where $n_{\widehat{M}}$ is the number of CT slots $\widehat{M}$, and $\text{Cat}(\cdot,\cdot)_{[-n:]}$ denotes the concatenation by augmenting with new compressed chunk and only storing the last n slots. This memory is constructed as shown in green shaded block. Different from [19], the memory part of CT is removed to speed up the computation during inference. This is reasonable because the information of each frame is usually limited. Next, the multi-head attention operation is similar to transformer at each layer n, $\text{attn}(H_{n,i}^x, \widehat{H}_{n,i}^x) = \text{Softmax}\left(H_{n,i}^x W_n^q (\widehat{H}_{n,i}^x W_n^k)^T\right) \widehat{H}_{n,i}^x W_n^v$, where $\widehat{H}_{n,i}^x = \text{Cat}(\widehat{M}_{n,i}, H_{n,i}^x)$ is concatenated with compressive memory $\widehat{M}_{n,i}$ and hidden state $H_{n,i}^x$, and $\{W_n^q, W_n^k, W_n^v\}$ denote the self attention parameters at layer n. In training stage, the decoder θ_d predicts the target words $\mathbf{y}_{1:J+1}$ by given the previous words $\mathbf{y}_{0:J}$ and the output features of encoder θ_e at last layer $N+1$ by using each chunk C_i and the corresponding compressive memory $\widehat{M}_i$, which are expressed by $H_{N+1,i}^x = \text{Encoder}(C_i, \widehat{M}_i; \theta_e, \theta_c)$, where previous words $\mathbf{y}_{0:J}$ is the right shift of target sequence $\mathbf{y}_{1:J+1}$. The probability of target words using a lattice graph is calculated via N layers of decoder, i.e. $p(\mathbf{y}_{1:J+1}|\mathbf{y}_{0:J}, C_i) = \text{Decoder}(\mathbf{y}_{0:J}, H_{N+1,i}^x; \theta_d)$, where $\mathbf{y}_0 = \mathbf{y}_{J+1} = \langle \text{blk} \rangle$, and $\langle \text{blk} \rangle$ can be either $\langle \text{SOS} \rangle$ or $\langle \text{EOS} \rangle$. However, the decoder of the resulting OCT is still limited with the information of previous chunks from encoder. Alternatively, one variant of OCT is implemented by merging an extra compressive memory $\widehat{M}_{N+1,i}$ after the output of encoder by preserving the previous chunks information as shown in the figure. The features of encoder output $H_{N+1,i}^x$ and decoder output $\mathbf{y}_{1:J+1}$ are integrated with extra memory by

$$p(\mathbf{y}_{1:J+1}|\mathbf{y}_{0:J}, C_i) = \text{Decoder}(\mathbf{y}_{0:J}, \widehat{M}_{N+1,i}, H_{N+1,i}^x; \theta_d).$$

3.2. Objectives for training

In addition, the connectionist temporal classification (CTC) loss $\mathcal{L}_{\text{ctc}}$ [5] is introduced to improve the training convergence for E2E ASR. Using OCT without extra compressive memory, the CTC loss is minimized to reduce the prediction error by using the hidden variable from encoder $H^x_{N+1,1:I}$ provided with the whole input chunks $C_{1:I}$, i.e. $\mathcal{L}^{(1)}_{\text{ctc}} = -\log p_{\text{ctc}}(\mathbf{y}|H^x_{N+1,1:I})$. Using the variant of OCT, the model with extra compressive memory $\widehat{M}_{N+1,1:I}$ in last layer of encoder, $H^x_{N+1,1:I}$ is replaced by the compressive memory to construct the CTC loss $\mathcal{L}^{(2)}_{\text{ctc}} = -\log p_{\text{ctc}}(\mathbf{y}|\widehat{M}_{N+1,1:I})$ where $\widehat{M}_{N+1,1:I} = f^c(H^x_{N+1,1:I})$. The CTC distribution $p_{\text{ctc}}(\cdot)$ is calculated over a many-to-one mapping between encoder outputs and target values and is affected by the encoder and compressor parameters $\{\theta_e, \theta_c\}$. Overall, the learning objectives in OCT include the attention reconstruction loss in (1), the RNN-T loss in (2) and the CTC loss ($\mathcal{L}^{(1)}_{\text{ctc}}$ or $\mathcal{L}^{(2)}_{\text{ctc}}$) where the total loss $\mathcal{L}_{\text{total}} = \mathcal{L}_{\text{rnnt}} + \mathcal{L}_{\text{attn}} + \mathcal{L}_{\text{ctc}}$ is minimized to train $\{\theta_e, \theta_d, \theta_c\}$ via their gradients. Notably, the attention reconstruction loss [25, 26] in OCT is separately minimized for encoder and decoder based on the reconstruction due to compressive memory $\widehat{M}$ evolving the feature extraction process of input $\mathbf{x}$ and output sequences $\mathbf{y}$, respectively, i.e. $\mathcal{L}^e_{\text{attn}} = \sum_{n=1}^{N} \sum_{i=1}^{I} \|\text{attn}(H^x_{n,i}, H^x_{n,i}) - \text{attn}(H^x_{n,i}, \widehat{M}_{n,i})\|^2, \mathcal{L}^d_{\text{attn}} = \sum_{n=1}^{N} \sum_{i=1}^{I} \|\text{attn}(H^y_n, H^x_{N+1,i}) - \text{attn}(H^y_n, \widehat{M}_{N+1,i})\|^2$, where H^y_n denotes the output embedding at decoder layer n. The resulting compressive function $f^c(\cdot)$ is learned to reconstruct original context from memories based on the attention operation. Different from compressive transformer [19], attention reconstruction loss in OCT is optimized with respect to all parameters in model not only the compressive memory module θ_c. The total attention reconstruction loss is formed by $\mathcal{L}_{\text{attn}} = \mathcal{L}^e_{\text{attn}} + \mathcal{L}^d_{\text{attn}}$. To sum up, CTC loss is used for helping encoder learning the alignment knowledge between full-length input and output sequence, and RNN-T loss is used for learning online decoding due to each output token only depends on previous tokens. OCT model trained with RNN-T could keep generating output tokens in each chunk before it generates the blank symbol ⟨blk⟩. After ⟨blk⟩ is generated, OCT starts to generate word tokens from next chunk.

3.3. Online inference in test time

In order to improve the inference speed, the one-step constrained (OSC) beam search [27] is applied. Similar to the RNN-T loss in training, OSC beam search aims to find a decoding path which maximizes the accumulated decoding probability but with a limited beam or expansion size E in each decoding step during testing. This search algorithm recursively adopts the previous hypothesis probability $p(\widehat{\mathbf{y}})$ in hypotheses set $\mathcal{Y}$ to calculate new hypothesis probability for predicting ⟨blk⟩, $p(\langle\text{blk}\rangle|\widehat{\mathbf{y}}, H^x_{N+1,i})$, and all possible words w in a vocabulary $\mathcal{V}$, $p(w|\widehat{\mathbf{y}}, H^x_{N+1,i})$. Let S and Z denote the prediction of ⟨blk⟩ and word w using current chunk C_i, respectively. The probability of each hypothesis in S and Z is formed by the product of previous hypothesis probability $p(\widehat{\mathbf{y}})$ and the output probabilities of decoder, $p(w|\widehat{\mathbf{y}}, H^x_{N+1,i})$ and $p(\langle\text{blk}\rangle|\widehat{\mathbf{y}}, H^x_{N+1,i})$. After E times hypotheses searching and updating in current chunk, the top L hypotheses are selected from S for next decoding step and the top one in $\mathcal{Y}_i$ is generated or yielded as the decoding result of current chunk C_i. In OSC beam search [27], the hypotheses expanded once or twice at each inference step in most cases. In OCT, we also found similar phenomenon, where the hypothesis expands two or three times. Thus, the search space is possible to be reduced and OSC beam search could be applied on decoding the OCT output. In addition, the prefix search is removed due to time-consuming computation. Algorithm 1 shows the online inference procedure of using compressive transformer. Apart from OSC beam search, the compressive memory $\widehat{M}_{1:N+1,i}$ of all $N+1$ layers are calculated at the end of each decoding iteration instead of inside encoder. This can speed up the whole generation process.

Algorithm 1: Online inference procedure using OCT

```
Initialize:
  $\widetilde{M} = \widehat{M}_{1:N+1} = \mathbf{0}, \widetilde{H}^x = \widehat{H}^x_{N+1}, \mathcal{Y}_0 = \{\langle\text{blk}\rangle\}$
Inference
  For $i = 1, \cdots, I$ do
    $\widetilde{H}i^x = \text{Encoder}(C_i, \widetilde{M}_i; \theta_e, \theta_c)$
    $S_0 = \{p(\widehat{\mathbf{y}})p(\langle\text{blk}\rangle|\widehat{\mathbf{y}}, \widetilde{H}i^x)|\widehat{\mathbf{y}} \in \mathcal{Y}_{i-1}\}$
    $Z_0 = \{p(\widehat{\mathbf{y}})p(w|\widehat{\mathbf{y}}, \widetilde{H}^x_i)|w \in \mathcal{V}, \widehat{\mathbf{y}} \in \mathcal{Y}_{i-1}\}$
    For $j = 1, \cdots, E$ do
      $\widehat{Z}_{j-1} =$ top $L$ hypotheses in $Z_{j-1}$
      $S_j = \{p(\widehat{\mathbf{y}})p(\langle\text{blk}\rangle|\widehat{\mathbf{y}}, \widetilde{H}^x_i)|\widehat{\mathbf{y}} \in \widehat{Z}_{j-1}\}$
      $Z_j = \{p(\widehat{\mathbf{y}})p(w|\widehat{\mathbf{y}}, \widetilde{H}^x_i)|w \in \mathcal{V}, \widehat{\mathbf{y}} \in \widehat{Z}_{j-1}\}$
    end
    $\mathcal{Y}_i =$top $L$ hypotheses in $S_{0:E}$
    $\widetilde{M}_{i+1} = \text{Cat}(\widetilde{M}_i, f^c(H^x_i))$
    Yield: best hypothesis in $\mathcal{Y}_i$
  end
  Return: best hypothesis in $\mathcal{Y}_I$
```

4. Experiments and Results

4.1. Dataset

In the experiments, Aishell-1 [28] was used to evaluate the performance of proposed model. Aishell-1 consisted of 178 hours of Chinese speech in 11 domains including finance, sports, news, etc. The training, development and test sets contained speech data with 150 hours, 20 hours, 10 hours, respectively.

4.2. Experimental configuration

The sampling rate of audio signals in Aishell-1 was 16 KHz. We used 80-dimensional filter-bank features as the inputs for all models. The duration and shift of each frame was 25ms and 10ms, respectively. We followed the instructions in ESPNet [29] with the Aishell recipe and detailed script. All models were composed of a convolution module and $N = 6$ encoder and decoder layers. The convolution module consisted of 2 layers of time-domain 2d-convolution with 256 channels and kernel size 3. The feed forward activation was replaced by the gated linear unit [30, 9]. The dimensions of hidden states and the output of first linear layer in feedforward module were 256 and 1280, respectively. The number of compressive memory slots $n_{\widehat{M}}$ was 5 and the expansion size E was 2. The related works [31, 9] were included for comparison.

4.3. Results

4.3.1. Model comparison

Table 1 compares different methods' results. Sync-Transformer [9] is a variant of neural transducer which processes the inputs chunk by chunk in real time similar to OCT model. We implement OCT model with two different settings which is OCT with and without an extra compressive memory at the end of encoder. Table 1 shows the results. OCT with extra compressive memory achieves the best performance in the comparison, and also obtains the performance close to offline standard transformer.

Model	Dev	Test
Transformer [9]	7.8*	**8.6***
RNN-T [31]	10.1*	11.8*
Sync-Transformer [9]	7.9*	8.9*
Chunk-Flow SA-T [31]	8.6*	9.8*
OCT ($\mathcal{L}_{\text{rnnt}}$,$\mathcal{L}_{\text{attn}}$,$\mathcal{L}_{\text{ctc}}^{(1)}$)	8.6	9.7
OCT* ($\mathcal{L}_{\text{rnnt}}$)	8.5	9.6
OCT* ($\mathcal{L}_{\text{rnnt}}$,$\mathcal{L}_{\text{attn}}$)	8.3	9.4
OCT* ($\mathcal{L}_{\text{rnnt}}$,$\mathcal{L}_{\text{attn}}$,$\mathcal{L}_{\text{ctc}}^{(2)}$)	**7.5**	**8.7**

Table 1: *Comparison of character error rate (CER) (%) using different methods. CERs with * are referred from original papers with citations. OCT is the base model, OCT* is the model with an extra compressive memory after the output of encoder.*

4.3.2. *Evaluation on loss function*

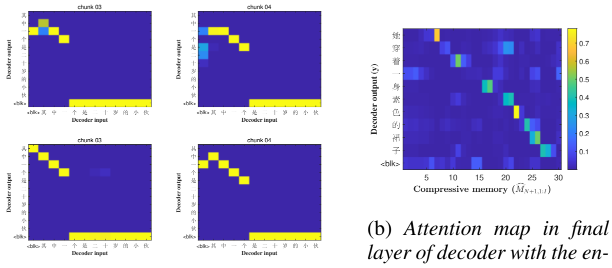
To understand the impact on performance by training with different loss functions, we do an ablation study between the models trained with or without CTC and attention reconstruction losses. The window size (W) and compressive rate of all models are 9 and 3, respectively. The result is shown in Table 1. It is found that an obviously gap between OCT model with and without CTC loss is obtained. CTC loss does help for improving the performance, because it forces the encoder to learn the alignment between input and output sequences.

4.3.3. *Evaluation on chunk information*

Next, the long-range context information in OCT and Sync-Transformer is evaluated. We would like to evaluate which one generates more symbols by using a chunk. Figure 2a shows the decoder input-output probability maps generated by Sync-Transformer and OCT in online inference at chunks 3 and 4. Here, x-axis is the ground-truth label of decoder and y-axis is the output of decoder. The example we take is from an audio file ”BAC009S0012W0456” in Aishell-1 dataset. The label sequence is 其中一個是二十歲的小伙 (One of them is a twenty-year-old guy). We find that OCT model decodes 4 output symbols in chunk 3 as well as chunk 4, but Sync-transformer only decodes 3 output symbols. This shows that compressive memory in OCT model provides helpful information for decoding.

4.3.4. *Online inference speed and performance comparison*

Table 2 compares the results between the models with different beam search settings. For OCT model, there is no input overlapping ($O = 0$) because the compressive memory is enough to preserve the previous information. The real-time factor (RTF) is the metric to measure the speed of ASR system. RTF is defined as the ratio of the model inference time and the input audio duration. All the results in Table 2 are consistently measured by Algorithm 1 and using CPU Intel E5-2620 v4. It is found that the model with larger beam search expansion size ($E = 2$) performs better than smaller size ($E = 1$) since model can generate more tokens in each chunk. Also, the proposed OCT model with extra compressed memory outperforms Sync-Transformer in terms of CER while their RTFs are comparable.



(a) *Decoder outputs of Sync Transformer (first row) and online compressive transformer (second row).*

(b) *Attention map in final layer of decoder with the encoder output replaced by a set of compressive memory slots.*

Figure 2: *Decoder outputs and attention map analysis.*

Model	RTF	CER (%)
greedy search:		
OCT	0.088	9.9
OCT*	0.090	9.0
beam search 5:		
Sync-Transformer ($O = 3, W = 10$)	0.329	9.2
Sync-Transformer ($O = 3, W = 12$)	0.250	9.4
OCT	0.246	9.7
OCT*	0.259	**8.7**
OCT* ($E = 1$)	**0.177**	9.7
OCT* (ESPnet recipe)	0.477	8.8

Table 2: *Comparison of RTF and CER using different methods for online inference. The lower RTF, the faster. E is set to 2 by default. Sync-Transformer is implemented by ourselves.*

4.3.5. *Evaluation on compressive memory*

Finally, we discard all the uncompressed features and only replace them by the compressive memories at the end of encoder as the decoder input to evaluate whether the compressive memory is possible to contain the information of whole speech. Figure 2b shows the result of attention map of decoder in final layer. This example is taken from an audio file ”BAC0009S012W0413” and the label sequence is 她穿著一身素色的裙子 (She wears a plain skirt.). The compressive memories $M_{N+1,1:I}$ of each chunk clearly form a shape of diagonal matrix. This means that compressive memory is capable to preserve information and is successfully used to decode the whole chunks of utterance. We generate all ground-truth symbols by using the proposed OCT. Source codes are accessible at https://github.com/NCTUMLlab/Chi-Hang-Leong.

5. Conclusions

This paper has presented an online automatic speech recognition model with compressive memory where the losses of CTC, RNN-T and attention reconstruction using transformer were minimized. From the experiments, we showed that the compressive memory was effective to memorize the history information, and OCT model could preserve the performance of offline model nearly. Overall, the proposed online model did not only obtain better performance but also achieve faster inference speed for online ASR using Aishell-1 corpus.

6. References

[1] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Łukasz Kaiser, and I. Polosukhin, "Attention is all you need," in *Proc. of Advances in Neural Information Processing Systems*, 2017, p. 6000–6010.

[2] J.-T. Chien and C.-W. Wang, "Hierarchical and self-attended sequence autoencoder," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2021.

[3] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Computation*, pp. 1735–1780, 1997.

[4] 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. of IEEE Automatic Speech Recognition and Understanding Workshop*, 2019, pp. 449–456.

[5] S. Karita, N. Yalta, 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. of Annual Conference of the International Speech Communication Association*, 2019, pp. 1408–1412.

[6] A. Graves, "Sequence transduction with recurrent neural networks," in *Proc. of International Conference of Machine Learning Workshop on Representation Learning*, 2012.

[7] C.-F. Yeh, J. Mahadeokar, K. Kalgaonkar, Y. Wang, D. Le, M. Jain, K. Schubert, C. Fuegen, and M. L. Seltzer, "Transformer-transducer: End-to-end speech recognition with self-attention," *arXiv:1910.12977v1*, 2019.

[8] 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. of IEEE International Conference on Acoustics, Speech and Signal Processing*, 2020, pp. 7829–7833.

[9] Z. Tian, J. Yi, Y. Bai, J. Tao, S. Zhang, and Z. Wen, "Synchronous transformers for end-to-end speech recognition," in *Proc. of IEEE International Conference on Acoustics, Speech and Signal Processing*, 2020, pp. 7884–7888.

[10] G. Saon and J.-T. Chien, "Large-vocabulary continuous speech recognition systems: A look at some recent advances," *IEEE Signal Processing Magazine*, vol. 29, no. 6, pp. 18–33, 2012.

[11] D. Yu, G. Hinton, N. Morgan, J.-T. Chien, and S. Sagayama, "Introduction to the special section on deep learning for speech and language processing," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 20, no. 1, pp. 4–6, 2011.

[12] H. Miao, G. Cheng, C. Gao, P. Zhang, and Y. Yan, "Transformer-based online CTC/attention end-to-end speech recognition architecture," in *Proc. of IEEE International Conference on Acoustics, Speech and Signal Processing*, 2020, pp. 6084–6088.

[13] S. Zhang, Z. Gao, H. Luo, M. Lei, J. Gao, Z. Yan, and L. Xie, "Streaming chunk-aware multihead attention for online end-to-end speech recognition," in *Proc. of Annual Conference of the International Speech Communication Association*, 2020, pp. 2142–2146.

[14] J.-T. Chien and W.-H. Chang, "Dualformer: a unified bidirectional sequence-to-sequence learning," in *Proc. of IEEE International Conference on Acoustics, Speech and Signal Processing*, 2021, pp. 7718–7722.

[15] ——, "Collaborative regularization for bidirectional domain mapping," in *Proc. of International Joint Conference on Neural Networks*, 2021.

[16] N. Jaitly, Q. V. Le, O. Vinyals, I. Sutskever, D. Sussillo, and S. Bengio, "An online sequence-to-sequence model using partial conditioning," in *Proc. of Advances in Neural Information Processing Systems*, 2016, pp. 5067–5075.

[17] T. N. Sainath, C. Chiu, R. Prabhavalkar, A. Kannan, Y. Wu, P. Nguyen, and Z. Chen, "Improving the performance of online neural transducer models," in *Proc. of IEEE International Conference on Acoustics, Speech and Signal Processing*, 2018, pp. 5864–5868.

[18] N. Moritz, T. Hori, and J. L. Roux, "Triggered attention for end-to-end speech recognition," in *Proc. of IEEE International Conference on Acoustics, Speech and Signal Processing*, 2019, pp. 5666–5670.

[19] J. W. Rae, A. Potapenko, S. M. Jayakumar, C. Hillier, and T. P. Lillicrap, "Compressive transformers for long-range sequence modelling," in *Proc. of International Conference on Learning Representations*, 2020.

[20] J. Kim, Y. Lee, and E. Kim, "Accelerating RNN transducer inference via adaptive expansion search," *IEEE Signal Processing Letters*, vol. 27, pp. 2019–2023, 2020.

[21] J.-T. Chien, "Association pattern language modeling," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 14, no. 5, pp. 1719–1728, 2006.

[22] Z. Dai, Z. Yang, Y. Yang, J. Carbonell, Q. Le, and R. Salakhutdinov, "Transformer-XL: Attentive language models beyond a fixed-length context," in *Proc. of Annual Meeting of the Association for Computational Linguistics*, 2019, pp. 2978–2988.

[23] J.-T. Chien and Y.-C. Ku, "Bayesian recurrent neural network for language modeling," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 27, no. 2, pp. 361–374, 2015.

[24] J.-T. Chien and C.-W. Wang, "Self attention in variational sequential learning for summarization," in *Proc. of Annual Conference of the International Speech Communication Association*, 2019, pp. 1318–1322.

[25] J.-T. Chien and T.-A. Lin, "Supportive attention in end-to-end memory networks," in *Proc. of IEEE International Workshop on Machine Learning for Signal Processing*, 2018, pp. 1–6.

[26] J.-T. Chien and Y.-H. Chen, "Continuous-time self-attention in neural differential equation," in *Proc. of IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2021, pp. 3290–3294.

[27] J. Kim and Y. Lee, "Accelerating RNN transducer inference via one-step constrained beam search," *arXiv:2002.03577v1*, 2020.

[28] H. Bu, J. Du, X. Na, B. Wu, and H. Zheng, "AISHELL-1: an open-source mandarin speech corpus and a speech recognition baseline," in *Proc. of Conference of the Oriental Chapter of the International Coordinating Committee on Speech Databases and Speech I/O Systems and Assessment*, 2017, pp. 1–5.

[29] S. Mehta, M. Rastegari, A. Caspi, L. Shapiro, and H. Hajishirzi, "ESPNet: Efficient spatial pyramid of dilated convolutions for semantic segmentation," in *Proc. of European Conference on Computer Vision*, 2018.

[30] N. Shazeer, "GLU variants improve transformer," *arXiv:2002.05202v12*, 2020.

[31] Z. Tian, J. Yi, J. Tao, Y. Bai, and Z. Wen, "Self-attention transducers for end-to-end speech recognition," in *Proc. of Annual Conference of the International Speech Communication Association*, 2019, pp. 4395–4399.