Learn To Remember: Transformer with Recurrent Memory for Document-Level Machine Translation

Yukun Feng[1], Feng Li[2], Ziang Song[1], Boyuan Zheng[1], and Philipp Koehn[1]
[1]Department of Computer Science, Johns Hopkins University
[2]Department of Computer Science, University of Illinois Urbana-Champaign
{yfeng55, zsong17, bzheng12, phi}@jhu.edu, {fengl3}@illinois.edu

Abstract

The Transformer architecture has led to significant gains in machine translation. However, most studies focus on only sentence-level translation without considering the context dependency within documents, leading to the inadequacy of document-level coherence. Some recent research tried to mitigate this issue by introducing an additional context encoder or translating with multiple sentences or even the entire document. Such methods may lose the information on the target side or have an increasing computational complexity as documents get longer. To address such problems, we introduce a recurrent memory unit to the vanilla Transformer, which supports the information exchange between the sentence and previous context. The memory unit is recurrently updated by acquiring information from sentences, and passing the aggregated knowledge back to subsequent sentence states. We follow a two-stage training strategy, in which the model is first trained at the sentence level and then finetuned for document-level translation. We conduct experiments on three popular datasets for document-level machine translation and our model has an average improvement of 0.91 s-BLEU over the sentence-level baseline. We also achieve state-of-the-art results on TED and News, outperforming the previous work by 0.36 s-BLEU and 1.49 d-BLEU on average.

1 Introduction

Most previous machine translation methods are designed for sentence-level translation. Recent studies have shown that the effective use of contextual information between sentences can achieve better performance in document-level machine translation (Garcia et al., 2015; Maruf and Haffari, 2018; Miculicich et al., 2018; Zhang et al., 2020; Bao et al., 2021). Built on the Transformer model (Vaswani et al., 2017), a general approach is to incorporate neighboring sentence states (Tiedemann and Scherrer, 2017; Zheng et al., 2020) into the attention mechanism, which has also been widely used in many long sequence modeling methods (Dai et al., 2019; Rae et al., 2020; Yang et al., 2019; Beltagy et al., 2020). Zhang et al. (2018); Maruf et al. (2019) have introduced an additional context encoder to solve the limitation of sentence-level translation, which, however, is separated from the original translation model and context states is only applied on the source side. Other works (Junczys-Dowmunt, 2019; Scherrer et al., 2019; Zhang et al., 2020; Bao et al., 2021) concatenated sentences or the entire document and feed into the attention module of the Transformer. Since more extended contexts may confound attention on meaningful portions of the current sentence, the model is difficult to select valuable inputs from extra contexts to navigate the redundancy of information. Such methods also suffer from the quadratically increasing complexity when documents get longer.

We solve such problems by introducing a memory mechanism to recurrently integrate contextualized knowledge from intermediate state in Transformer layers. As recurrent memory has been widely researched since RNN (Rumelhart et al., 1986), which has been incorporated with Transformer by Transformer-XL (Dai et al., 2019) and further extended by Rae et al. (2020) who compress previous states into a two-layer hidden memory. In our approach, we update the memory through an attention module to select practical information from sentences and reduce the context space into multiple dense vectors in the memory. Besides, we use another attention module to pass the knowledge retained in the memory back to the sentence state in the next step. Such information exchange is expected to convey contextualized dependency between sentences. This memory mechanism can be applied in each layer for both the source and target documents, and our study shows that incorporating memory only in the last layer achieves the

Findings of the Association for Computational Linguistics: NAACL 2022, pages 1409 - 1420
July 10-15, 2022 ©2022 Association for Computational Linguistics

best performance. Also, as sentences are ordered in documents, our model reads one sentence pair at each step, keeping the computational cost as same as the sentence-level translation.

We experiment across three widely used datasets for document-level translation: TED, NEWS, and Europarl, and evaluate our model with s-BLEU and d-BLEU. We first train a vanilla Transformer on sentence-level translation as the baseline and finetune the model for the documents by initializing the memory mechanism to the Transformer. Our model outperforms previous SOTA work by 0.5 s-BLEU and 2.30 d-BLEU on TED, and 0.21 s-BLEU and 0.57 d-BLEU on News. We do not achieve the SOTA result on Europarl, which might be caused by the different results between the baselines for sentence-level translation. However, we further evaluate the improvement of previous works from their reported baseline Transformer, and we achieve the most relative gain on all three datasets. We also analyze our model from the memory usage, long-range effect, context dependency, and computational complexity, and demonstrate the effectiveness and efficiency of our approach in the general understanding of the document machine translation.

Overall, this paper makes several contributions: **(i)** Our work reduces the contextualized knowledge space of sentences states to multiple dense vectors, and considers the sentence dependency for both source and target documents, while keeping computational complexity in sentence-level. **(ii)** Our model significantly improves the sentence level baseline by 0.91 s-BLEU average and achieved the SOTA results on TED and News. **(iii)** Our model shows the effective use of memory, long-range influence, context-dependency across sentences, and decoding efficiency through convincing analysis.

2 Related Works

Recurrent Sequence Modeling RNN (Rumelhart et al., 1986) was the first class of models that introduced hidden states as the memory in neural models. Although improved on sequential-oriented tasks, RNN has unsatisfactory learning of long-term information due to gradient vanishing and explosion. LSTM (Hochreiter and Schmidhuber, 1997) improved RNN by introducing gate mechanisms to selectively retain knowledge at each step.

This RNN variant dominated NLP models until the Transformer replaced the memory unit with a self-attention mechanism and achieved great success in a wide range of NLP applications. Although we cannot deny the robustness and effectiveness of the Transformer model, the quadratically increased computational cost as the increase of token numbers makes Transformer unable to fit the long-range sequence. Some studies (Parmar et al., 2018; Child et al., 2019; Beltagy et al., 2020; Ainslie et al., 2020; Qiu et al., 2020; Zaheer et al., 2020; Martins et al., 2021) try to mitigate this issue by reducing the complexity of the attention module. However, such work still suffers from the problems by unlimited the document length and the document modeling is hard to solve.

Transformer-XL (Dai et al., 2019) broke this dilemma by introducing the recurrent memory into Transformer-based models. It cached previous hidden sentences computation and mapped such states to subsequent sentences states. Theoretically, Transformer-XL could handle infinite length text but storing uncompressed hidden state requires tremendous memory space, which impeded Transformer-XL from good performance on dealing with practical long-sequence tasks. The Compressive Transformer (Rae et al., 2020) further addressed this problem by mapping the evicted hidden state from cached memory to a more compressed representation. However, two-layer caching still requires a huge memory space and may be improved with trainable memories.

Document Machine Translation Machine Translation has been a widely researched area for decades. A series of models have addressed various translation problems (Koehn et al., 2003; Kalchbrenner and Blunsom, 2013; Bahdanau et al., 2015; Luong et al., 2015). As most of them target translation at the sentence level, document-level translation poses a fundamental challenge requiring models to pass intra-sentential information throughout consecutive sequences of sentences, and it has been addressed by Gong et al. (2011); Hardmeier et al. (2013); Pouget-Abadie et al. (2014); Garcia et al. (2015); Koehn and Knowles (2017); Läubli et al. (2018); Agrawal et al. (2018) among others.

Recent studies have attempted to incorporate additional contextual information into the Transformer structure to improve the performance of neural machine translation models further. The intuitive way is to leverage neighboring sentences from paragraphs or the documents (Tiedemann and

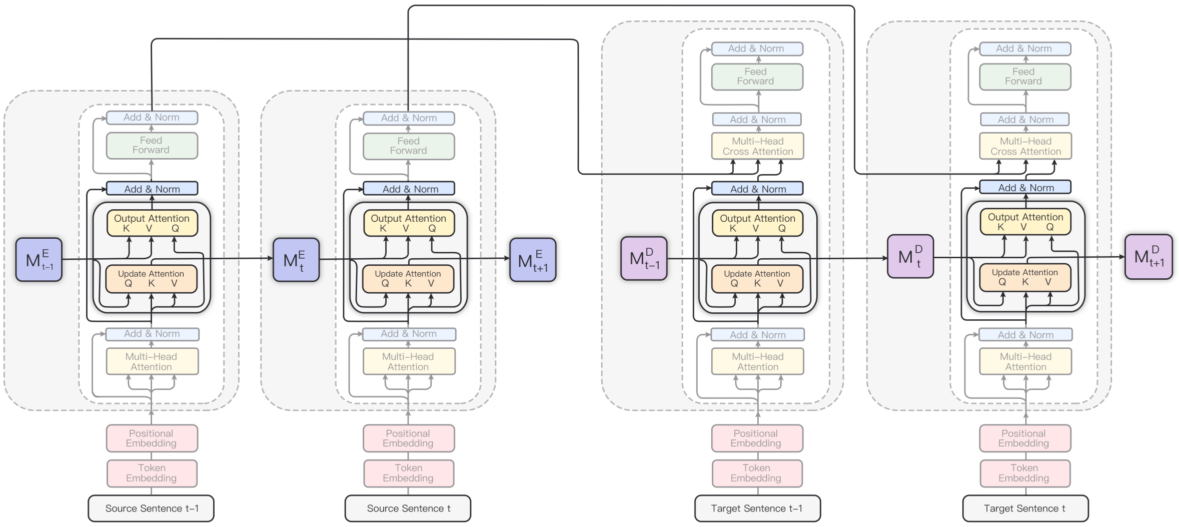

Figure 1: An overview of the model architecture, where E and D refers to Encoder and Decoder respectively.

Scherrer, 2017; Maruf and Haffari, 2018; Zheng et al., 2020), demonstrating the effectiveness of the additional contexts. Specifically, in the first class of methodologies for document-level translation, independent from the architecture of vanilla Transformer processing current sentences, some studies (Miculicich et al., 2018; Zhang et al., 2018; Maruf et al., 2019; Voita et al., 2019a,b; Ma et al., 2020; Donato et al., 2021) introduces context-aware components only attend to source or target contexts and usually jointly train with the rest of the network from scratch. The second class of models follows the pattern of concatenating multiple sentences for translation (Agrawal et al., 2018; Scherrer et al., 2019; Junczys-Dowmunt, 2019; Zhang et al., 2020). Such a method is expected to capture the contextual correlations between sentences. However, one of its drawbacks is the quadratically increased computational complexity in the face of longer contexts sequences. Also, longer sequences usually confound document-level attention and sometimes even overlook key information on the current sentences. Bao et al. (2021) uses group masks to introduce locality constraints to reinforce sentence information in multi-head attention to resolve the confounding issue in long contexts.

Our work incorporates the idea of the recurrent memory to document-level machine translation. It follows the locality assumptions by reducing the context space into multiple memory vectors and passes dependencies between sentences. The mechanism to update and output memory is similar to models which store cached bilingual sentence pairs in the memory to enhance the sentence-level translation (Feng et al., 2017; He et al., 2021; Jiang et al., 2021). We believe our approach is intuitive to efficiently store sentence states and transfer context information across sentences.

3 Approach

Our model is shown in Figure 1. Additional to the vanilla Transformer, we introduce a contextual memory unit and two attention modules to manipulate the memory defined as Update Attention and Output Attention. These modules can be applied at each layer in both the encoder and decoder.

As input sentences are ordered from left to right in the document, our model only reads one sentence every time. The memory is expected to store contextualized information from the input sentence states and convey such knowledge to the next sentence. At each step, the Update Attention step maps the contextual information from the sentence state to the memory, and updates the memory to the next step. Meanwhile, the Output Attention step fuses the information from the current sentence and the contextual memory, and outputs the aggregated knowledge to the remaining modules of the layer.

Formally, we define h^{t} as the sentence state from self-attention module in Transformer layer, and M^{t} refers to the contextual memory M at step t, where t refers to the index of t^{th} sentence in the document. M^{t} and h^{t} are updated and outputted as:

$$\begin{aligned} \mathrm{M}^{\mathrm{t}+1} &= \mathrm{UpdateAttention}(\mathrm{M}^{\mathrm{t}}, \mathrm{h}^{\mathrm{t}}) \\ \widetilde{\mathrm{h}}^{\mathrm{t}} &= \mathrm{OutputAttention}(\mathrm{M}^{\mathrm{t}}, \mathrm{h}^{\mathrm{t}}) \end{aligned} \quad (1)$$

3.1 Contextual Memory

Memory $\mathrm{M} \in \mathbb{R}^{\mathrm{d_M} \times \mathrm{d_{model}}}$ where $\mathrm{d_{model}}$ refers to the hidden dimension and $\mathrm{d_M}$ is a hyper-parameter, indicating how many vectors will be allocated for memory. To avoid the redundancy of memory space, we set $\mathrm{d_M}$ to 16. Detailed analysis is discussed later.

3.2 Update Attention

We update contextual memory through an attention module (Vaswani et al., 2017). Attention is a mapping function between input vectors of query (Q) and key-value (K-V) pairs. The output is the weighted sum of values with corresponding scores.

$$\mathrm{Attention}(\mathrm{Q}, \mathrm{K}, \mathrm{V}) = \mathrm{Softmax}(\frac{\mathrm{QK^T}}{\sqrt{\mathrm{d_k}}})\mathrm{V}$$

Multi-Head attention extends the vanilla attention by projecting input vectors $(\mathrm{Q}, \mathrm{K}, \mathrm{V})$ to different representation subspaces, and attention is performed in parallel in each head. Attention outputs from multiple heads will be concatenated and projected to the expected space.

$$\mathrm{MHA}(\mathrm{Q}, \mathrm{K}, \mathrm{V}) = \mathrm{Concat}(\mathrm{head_1}, .., \mathrm{head_n})\mathrm{W^o}$$
$$\mathrm{head_i} = \mathrm{Attention}(\mathrm{QW_i^q}, \mathrm{KW_i^k}, \mathrm{VW_i^v})$$

where $\mathrm{d_k}$ is the hidden dimension of the K, $\mathrm{W^q}$, $\mathrm{W^k}$, $\mathrm{W^v} \in \mathbb{R}^{\mathrm{d_{model}} \times \mathrm{d_h}}$, and $\mathrm{W^o} \in \mathbb{R}^{\mathrm{n} \times \mathrm{d_h} \times \mathrm{d_{model}}}$ are learnable parameters. $\mathrm{d_{model}}$ and $\mathrm{d_h}$ refer to the hidden dimension of the model and each head.

To update the contextual memory $\mathrm{M^t}$ to next step, sentence state $\mathrm{h^t}$ is mapped to $\mathrm{M^t}$ through the Multi-Head Attention. Both the memory and context state are projected into different sub-spaces and contextualized knowledge is expected to be mapped to each memory vector from different perspectives. The memory at step t is updated as:

$$\widetilde{\mathrm{M^t}} = \mathrm{AddNorm}(\mathrm{MHA}(\mathrm{M^t}, \mathrm{h^t}, \mathrm{h^t})) \quad (2)$$

A Feed-Forward Network is then used to further enhance the memory representation from the attention output.

$$\mathrm{M^{t+1}} = \mathrm{AddNorm}(\mathrm{FeedForward}(\widetilde{\mathrm{M^t}})) \quad (3)$$

In the memory matrix M, each vector is expected to select contextualized information from different perspectives. However, it is hard to distinguish such vectors since they do not have actual positional meanings, and the same key-value pairs are mapped to these vectors in the attention phase resulting in the same representation in each memory vector. To solve such a problem, we use the positional encoding $\mathrm{PE}()$ as introduced in Vaswani et al. (2017) to differentiate multiple memory vectors. M is added by such position-level bias in each update phase.

$$\mathrm{M^t} = \mathrm{M^t} + \mathrm{PE}(\mathrm{M^t}) \quad (4)$$

3.3 Output Attention

To map the contextualized knowledge from $\mathrm{M^t}$ to the sentence state $\mathrm{h^t}$, multi-head attention is used to take the representation of $\mathrm{h^t}$ and $\mathrm{M^t}$ as query and key-value, respectively.

$$\widetilde{\mathrm{h^t}} = \mathrm{MHA}(\mathrm{h^t}, \mathrm{M^t}, \mathrm{M^t}) \quad (5)$$

$\widetilde{\mathrm{h_t}}$ will be passed to the subsequent modules in the Transformer layer.

Similar approaches have been discussed in previous works. Simply increasing the context space does not help but introduces a lot of noise. Instead of incorporating multiple sentences to the context attention, we compress contextualized information into multiple memorized vectors and map such vectors back to the sentence state at the next step. We find that both the BLEU score and the information gained from the context attention space do not increase when the memory length increases from 64 to 128. Therefore, a large context space in M seems redundant for the model to learn, and we find $\mathrm{d_M} = 16$ for the most effectiveness and efficiency.

3.4 Document Neural Machine Translation

In the task of document-level machine translation, the source and target documents are represented as sequences of sentences $\mathrm{X} = \{\mathrm{x_t} | 1 \leq \mathrm{t} \leq \mathrm{n}\}$, and $\mathrm{Y} = \{\mathrm{y_t} | 1 \leq \mathrm{t} \leq \mathrm{n}\}$ respectively, where t refers to the sentence index. Given a vanilla Transformer and its parameters θ, the objective is to maximize the target document probability conditioned on the source document.

$$\underset{\theta}{\mathrm{argmax}}\, \mathrm{P}(\mathrm{Y}|\mathrm{X}, \theta)$$

Our approach recurrently translates an ordered document sentence by sentence, and the objective is:

$$\underset{\widetilde{\theta}}{\mathrm{argmax}} \prod_{\mathrm{t}} \mathrm{P}(\mathrm{Y_t}|\mathrm{X_{\leqslant t}}, \mathrm{Y_{<t}}, \widetilde{\theta})$$

where $\widetilde{\theta}$ refers to Transformer parameters including Memory, Update Attention and Output Attention.

Model	TED		News		Europarl	
	s-BLEU	d-BLEU	s-BLEU	d-BLEU	s-BLEU	d-BLEU
Vaswani et al. (2017)	23.10	-	22.40	-	29.40	-
Miculicich et al. (2018)	24.58	-	25.03	-	28.60	-
Maruf et al. (2019)	24.42	-	24.84	-	29.75	-
Ma et al. (2020)	24.87	-	23.55	-	30.09	-
Zheng et al. (2020)	25.10	-	24.91	-	30.40	-
Bao et al. (2021)	25.12 (+0.30)	27.17	25.52 (+0.33)	27.11	**32.39** (+1.02)	**34.08**
Sentence Baseline	24.73	-	25.18	-	30.13	-
Finetune on Sentence	**25.62 (+0.89)**	**29.47**	**25.73 (+0.55)**	**27.78**	31.41 (**+1.28**)	33.50

Table 1: Experiments results of BLEU scores on three datasets. The improvement from the Transformer baseline for previous models are also reported as in "()". It indicates the score improved from sentence-level translation provided by their implementations. Results are averaged from two runs.

Data	# of Docs	# of Sents/Doc
TED	1.7K/93/23	123/98/105
News	6.1K/71/155	40/25/20
Europarl	118K/240/360	14/15/14

Table 2: Dataset Statistics for Train/Valid/Test

As suggested by Beltagy et al. (2020); Bao et al. (2021), context would be better applied in higher layers and keep only local information in lower layers. Therefore we only apply the memory unit M in the top layer of encoder and decoder, and in lower layers, we keep using the original Transformer structure. Analysis regarding the location of memory is discussed in Section 5.1.

Training During training, our model takes an input of $\mathrm{x_t}$ and $\mathrm{y_t}$, which refer to sequences of tokens of the $\mathrm{t^{th}}$ sentence in source and target documents. Memory unit M is initialized trainable parameters before the first input of each document, and it will be updated after each input sentence pair, which are batched as the sentence order in the document. For computational convenience, the gradients are only back-propagated to the current sentence and the most recent sentence in each update step, and we stop the gradient for M before it is passed to the next step.

Inference In the decoding phase, our model translates the source document sentence by sentence. In the generation of each sentence, tokens are decoded in an auto-regressive order until the stop sign or exceeds the max length. The memory M will not be updated until the complete sentence is generated since the update of M depends on all tokens in the current sentence. If M is updated after each token generation, the attention space in the output attention does not represent the complete contextualized information of the expected sentence. The computational complexity keeps in sentence-level since we only feed one sentence every time, and there is no cache vector besides M.

4 Experiment

4.1 Datasets

We experiment across three widely used datasets for English→German document translation.

TED Training data for TED comes from IWSLT'17. We use tst2016-2017 as test set and a held-out set from training as valid.

News The corpus comes from News Commentary v11. We use tst2016-2017 as test set and a held-out set from training as valid.

Europarl Train, valid and test sets are extracted from the corpus Europarl v7, as mentioned in (Maruf et al., 2019).

Detailed statistics for the datasets is in Table 2. Moses (Koehn et al., 2007) is used for data processing and BPE (Sennrich et al., 2016) is used with vocab-size of 30K for all datasets.

4.2 Settings

We adopt Transformer model with the transformer-base configurations as the baseline, which has six layers with a hidden size of 512 and an intermediate size of 2048. Token embedding is shared for source and target languages, and token indexes are encoded with a learnable embedding matrix. We first train a baseline model with vanilla Transformer architecture for sentence-level translation and finetune our model based on the sentence-level baseline. We use the AdamW optimizer with an ini-

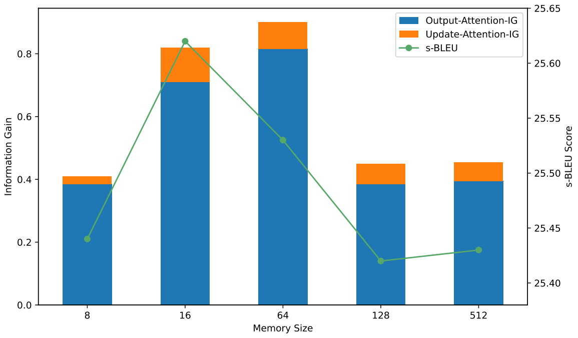

Figure 2: Evaluation on TED with different memory sizes.

tial learning rate of 5×10^{-4} and warm-up steps of 4000 for training sentence-level baseline. The drop-out rate is set to 0.3 for TED and News and 0.1 for the Europarl. As for finetuning after sentence-level Transformer, the learning rate is set as 3×10^{-4} for newly initialized parameters and 6×10^{-5} for pre-trained parameters, and warm-up steps of 1000 are set for TED and 2000 for News and Europarl. The drop-out rate is set to 0.1 for the Europarl and 0.2 for the TED and News during finetuning. We also apply gradient accumulation, and detailed studies are discussed in the section 5.3. Models are trained with a patience of 5 for both sentence-level and document-level. We use the beam size of 5 during inference and compute the BLEU score in a max order of 4 after removing BPE-tokens. s-BLEU and d-BLEU are used as evaluation metrics, where s-BLEU refers to the BLEU score for sentences, and d-BLEU is the score for documents.

4.3 Results

Experiment results are shown in Table 1. Our method shows consistent improvements over three datasets from sentence-level Transformer. We achieve the state-of-the-art results of s-BLEU of 25.62 and d-BLEU of 29.47 on TED and s-BLEU of 25.73 and d-BLEU of 27.78 on News . Though our results do not outperform G-Transformer (Bao et al., 2021) on Europarl, we think the difference mostly comes from the gap between sentence-level baselines. Such difference may be caused by the implementation framework and computing resources, which they use Fairseq library and multiple GPUs, while we adopt the code from HuggingFace and only a single 1080-Ti GPU is used for our training. We further report the score of works gained from their reported baseline, and our model makes the greatest improvement on all three datasets. Overall, the results could demonstrate the advantages of our method in the general understanding of the document machine translation.

Side	Index	s-BLEU	d-BLEU
Source+Target	0-1	25.31	29.13
Source+Target	2-3	25.30	29.23
Source+Target	4-5	25.43	29.22
Source+Target	**5**	**25.62**	**29.47**
Source Only	5	25.42	29.33
Target Only	5	25.43	29.25

Table 3: Evaluation on TED with memory on different sides and layers. We adopt the 6-layer Transformer model finetued on the sentence-level baseline, and 0 refers to the first layer, and 5 refers to the last layer.

5 Analysis

In this section, we discuss our model from memory usage, long-range modeling, context effect, and computational complexity, respectively. Experiments are conducted with the model finetuned on the sentence baseline and evaluated on the TED, since TED has the most average sentence number per document, which is more likely to reflect the performance of our model for long documents.

5.1 Discussion of Memory

Memory Size Memory size is evaluated through information gain (IG) between the random initialized memory and well trained memory. It is calculated from attention maps in Update Attention and Output Attention. IG from Update Attention indicates the difference of selected information in the memory, and IG from Output Attention refers to how much contextualized knowledge in memory is mapped to the next sentence state. Figure 2 shows IG keeps increasing as memory size increases from 8 to 64, but it dramatically drops at the size of 128 and 512. While increasing the memory size can fit more contextual information, an excessively large memory space is likely to introduce redundant noise. Therefore, it indicates that contextualized knowledge should be better distributed into a relatively dense space. Based on the corresponding s-BLEU score, we set memory size to 16 in all other experiments for the most effectiveness.

Memory Side To analyze the effect of the memory on source and target documents, we set the memory on encoder, decoder and both sides respectively. We find that it is not only necessary to have the memory to convey the dependency between sentences on the source side but also in the decoding process for the target document. As shown in Table 3, applying the memory on either side can outperform the baseline but the model achieves better scores when incorporating the memory on both sides. It indicates the necessity of contextualized information for both source and target documents.

Memory Position Previous work (Bao et al., 2021; Beltagy et al., 2020) has shown that Transformer lower layers are more likely to have local information while the context is better incorporated into higher layers. We set the memory in lower, intermediate, and higher layers respectively. The results as shown in Table 3 are consistent with the claim. Applying memory in higher layers outperforms the others, and it is even better to have it on only the top layer, which satisfies that the model is more likely to focus on the locality on lower layers and fuse the contextualized information on the top.

5.2 Discussion of Long Dependency

Metric Breakdown To find out on what kind of sentences our model outperforms the sentence-level Transformer, we evaluate the TED dataset with respect to the sentence index in the document. Sentences are ordered fed into the model. We compute and average the s-BLEU for sentences at each sentence index in the document. We further average the scores for every ten index range. As in Figure 3, the x-axis refers to the index range of sentences (e.g., 20 refers to sentences with indexes from 10 to 20), and the y-axis indicates the s-BLEU difference between our model and sentence Transformer. Our model has consistently greater performance, especially for sentences in later part of documents, indicating our model has the superiority than the sentence-level Transformer on longer document translation and long-range modeling.

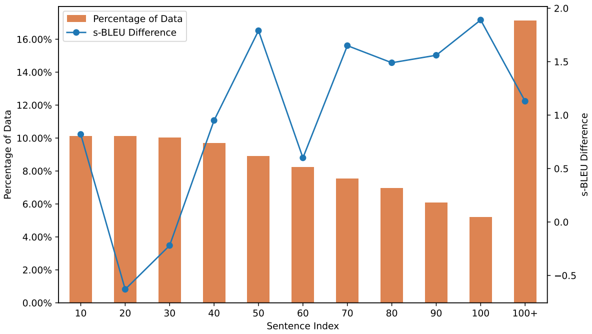

Figure 3: TED datset separated by sentences from different indexes in documents, evaluated with Sentence-Transformer and Context-Aware Model.

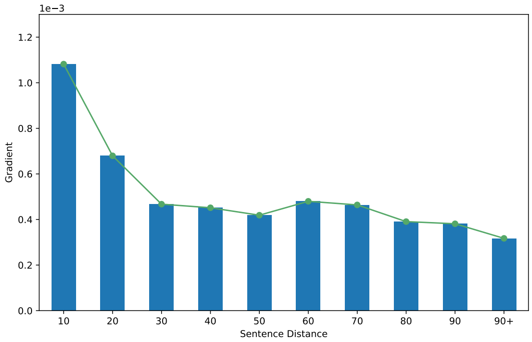

Figure 4: TED dataset evaluated by gradients computed from different sentences ranges. x-axis refers to the difference between the sentence indexes for gradient calculation and loss computation.

Long-Range Influence We also analyze the long-range dependency of our model through gradient attribution test introduced by Ancona et al. (2018); Sundararajan et al. (2017). The gradient attribution test reflects the significance of the model input feature to its output prediction. We perform this test by calculating the gradients of our well-trained model on the test set of TED. Since sentences are ordered when fed into the model, evaluating previous sentences' gradient attribution to the current sentence infers if the model supports the long-range dependency. More formally, we define the gradient of the previous sentence i computed by the loss propagated from current sentence j as:

$$\mathrm{G}(\mathrm{Sent_i}, \mathrm{Sent_j}).$$

Specifically, the gradient of a certain token in previous sentences is retrieved from its corresponding embedding weight. We conducted experiments for different sentence ranges k for the test with ten sentences intervals, and the gradient for each range k is computed as:

$$\mathrm{Score(k)} = \mathrm{Avg}(\sum_{\mathrm{d=1}}^{\mathrm{D}} \sum_{\mathrm{s=1}}^{\mathrm{S_d}} \sum_{\mathrm{i=s+k}}^{\mathrm{s+k+10}} \mathrm{G}(\mathrm{Sent_s}, \mathrm{Sent_i}))$$

where D refers to number of documents, $\mathrm{S_d}$ refers to number of sentences in Document d. To prevent the gradient attribution accumulated by the same token within the evaluated range, only unique tokens within this range are considered. As shown in Figure 4, our model has gradients propagated to sentence tokens even by 90+ sentences from the computed loss, indicating our model does have the ability for long-range sequence modeling.

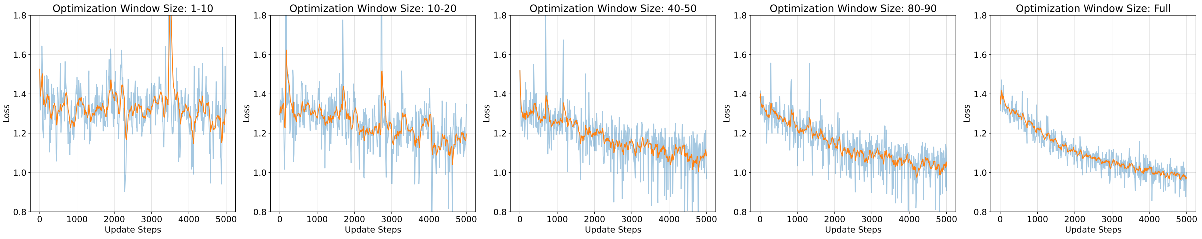

Figure 5: Training Loss on TED Dataset, with different optimization window sizes

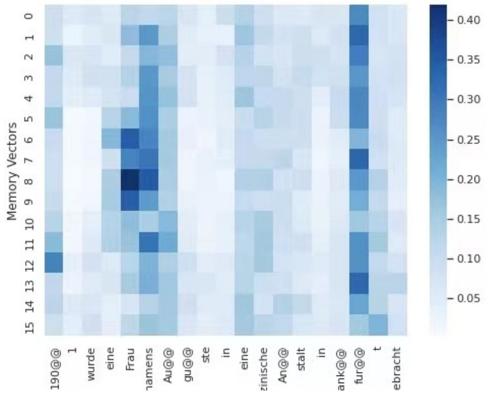

Figure 6: Attention map from Update Attention, each token at sentence t is mapped to each memory vector.

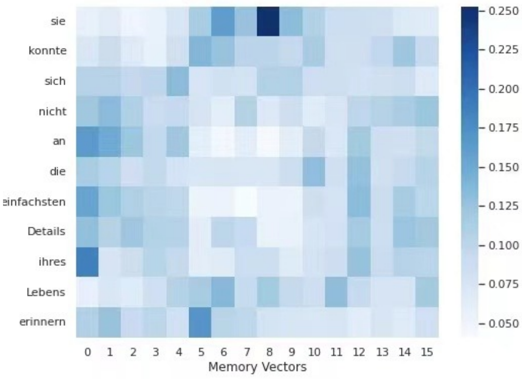

Figure 7: Attention map from Output Attention, memory vectors are mapped to each token in sentence t+1.

5.3 Discussion of Context

Convergence Our model is trained concerning the sentence order in the document. We find the model hard to converge during training as the loss oscillates within a wide range. Because of the various distribution of consecutive sentences in documents, the directions of continuing optimization steps vary greatly, resulting in an unstable convergence curve. To mitigate this issue, we use group optimization to update the model, considering the dependency among sentences. Specifically, a number from the optimization window is randomly sampled, and the gradients are accumulated. The model will not be updated until the accumulated steps reach the sampled number. We conduct experiments with different optimization window sizes for the update of 5000 steps, and the loss curves are shown in Figure 5, where full means the total number of sentences in the document. The result shows that the model converges faster and more stable with increasing optimization window size. Such improvement benefits from the grouped update steps concerning the difference of contextualized distribution among sentences.

Dependency Across Sentences We evaluate the attention maps from Update Attention and Output Attention to determine what contextualized information is passed in and out from memory. In Figure 6, tokens from t^{th} sentence are mapped to each memory vector, and the 8^{th} memory vector has a substantial attention weight on token "Frau". Figure 7 shows memory vectors are mapped back to the following sentence and the token "sie" has a high probability on the 8^{th} memory vector. German words "Frau" and "sie" refer to "Mrs" and "she" in English. Hence, the memory mechanism has the ability to parse the word dependency between sentences at different steps.

5.4 Discussion of Complexity

We further analyze our model's space and time complexity during the inference phase. Since we only evaluate the decoding speed and memory efficiency in this case, we use dummy tokens to perform the inference. We randomly generate a sequence of tokens as the source inputs and let the model decode the same number of tokens as the target. We compare our model with both the sentence-level Transformer and document-level Transformer. For the sentence-level Transformer, we split the sequence of tokens into chunks, and each chunk has a length of 100. The decoding complexity is

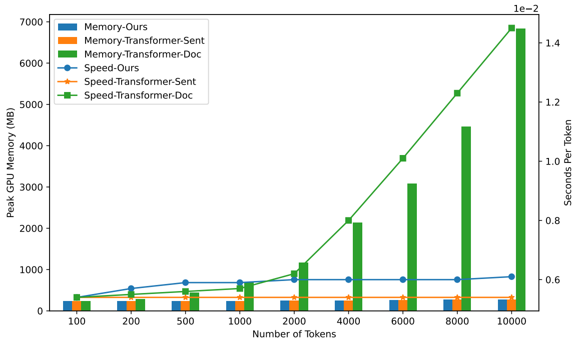

Figure 8: Space and Time Complexity for different number of tokens during inference.

evaluated over all chunks. For the document-level Transformer, we use the entire sequence of tokens as the source input and evaluate the complexity of decoding the entire target sequence. Similar to the sentence-level Transformer, our model is evaluated by the chunk by chunk decoding, and meanwhile, we keep the contextual memory updated. As shown in Figure 8, our model keeps the same space complexity as the sentence-level Transformer and takes a slightly more time cost because of the update of contextual memory. However, the document-level Transformer has an increasing cost for both space and time complexity, especially when the target sequence has a length greater than 1,000 tokens. Overall, results have shown the decoding efficiency of our model, which keeps the computational complexity as low as the sentence-level Transformer, even in the case of over thousands of tokens.

6 Conclusion

This paper introduces a memory unit that recurrently maps information into and out of Transformer intermediate states and addresses the limitation about the context dependency and computational complexity in document-level machine translation. We have achieved the SOTA score on TED and News and a great improvement from the sentence-level baseline. Our model demonstrates the effectiveness and efficiency of reduced memory space, context dependency for both source and target document, and long range influence across documents. The limitation of our work is the training cost since we accumulate the update steps and retain the graph for memory update at each step. Our work does not conduct experiments for pretrained settings due to the time limitation. However, it should be easy to apply our method to any Transformer-based pretrained models, such as Liu et al. (2020). Also, this paper only experiments on document-level machine translation, and future works may apply this approach for other tasks that need long-range sequence modeling.

Acknowledgements

We sincerely thank the reviewers from ACL Rolling Review for their helpful feedback, and the suggestions regarding figure plotting from Senior UI designer Xiangru Chen. We also appreciate the support of the computing resources from the Center for Language and Speech Processing (CLSP) at Johns Hopkins University (JHU).

References

Ruchit Agrawal, Marco Turchi, and Matteo Negri. 2018. Contextual handling in neural machine translation: Look behind, ahead and on both sides.

Joshua Ainslie, Santiago Ontanon, Chris Alberti, Vaclav Cvicek, Zachary Fisher, Philip Pham, Anirudh Ravula, Sumit Sanghai, Qifan Wang, and Li Yang. 2020. ETC: Encoding long and structured inputs in transformers. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 268–284, Online. Association for Computational Linguistics.

Marco Ancona, Enea Ceolini, Cengiz Öztireli, and Markus Gross. 2018. Towards better understanding of gradient-based attribution methods for deep neural networks. In *International Conference on Learning Representations*.

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2015. Neural machine translation by jointly learning to align and translate. *CoRR*, abs/1409.0473.

Guangsheng Bao, Yue Zhang, Zhiyang Teng, Boxing Chen, and Weihua Luo. 2021. G-transformer for document-level machine translation. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 3442–3455, Online. Association for Computational Linguistics.

Iz Beltagy, Matthew E. Peters, and Arman Cohan. 2020. Longformer: The long-document transformer. *arXiv:2004.05150*.

Rewon Child, Scott Gray, Alec Radford, and Ilya Sutskever. 2019. Generating long sequences with sparse transformers.

Zihang Dai, Zhilin Yang, Yiming Yang, Jaime Carbonell, Quoc Le, and Ruslan Salakhutdinov. 2019. Transformer-XL: Attentive language models beyond a fixed-length context. In *Proceedings of the 57th*

Annual Meeting of the Association for Computational Linguistics, pages 2978–2988, Florence, Italy. Association for Computational Linguistics.

Domenic Donato, Lei Yu, and Chris Dyer. 2021. Diverse pretrained context encodings improve document translation. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 1299–1311, Online. Association for Computational Linguistics.

Yang Feng, Shiyue Zhang, Andi Zhang, Dong Wang, and Andrew Abel. 2017. Memory-augmented neural machine translation. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 1390–1399, Copenhagen, Denmark. Association for Computational Linguistics.

Eva Martínez Garcia, Cristina España-Bonet, and Lluís Màrquez. 2015. Document-level machine translation with word vector models. In *Proceedings of the 18th Annual Conference of the European Association for Machine Translation*, pages 59–66, Antalya, Turkey.

Zhengxian Gong, Min Zhang, and Guodong Zhou. 2011. Cache-based document-level statistical machine translation. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing*, pages 909–919, Edinburgh, Scotland, UK. Association for Computational Linguistics.

Christian Hardmeier, Sara Stymne, Jörg Tiedemann, and Joakim Nivre. 2013. Docent: A document-level decoder for phrase-based statistical machine translation. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, pages 193–198, Sofia, Bulgaria. Association for Computational Linguistics.

Qiuxiang He, Guoping Huang, Qu Cui, Li Li, and Lemao Liu. 2021. Fast and accurate neural machine translation with translation memory. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 3170–3180, Online. Association for Computational Linguistics.

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural computation*, 9(8):1735–1780.

Shu Jiang, Rui Wang, Zuchao Li, Masao Utiyama, Kehai Chen, Eiichiro Sumita, Hai Zhao, and Bao liang Lu. 2021. Document-level neural machine translation with associated memory network.

Marcin Junczys-Dowmunt. 2019. Microsoft translator at WMT 2019: Towards large-scale document-level neural machine translation. In *Proceedings of the Fourth Conference on Machine Translation (Volume 2: Shared Task Papers, Day 1)*, pages 225–233, Florence, Italy. Association for Computational Linguistics.

Nal Kalchbrenner and Phil Blunsom. 2013. Recurrent continuous translation models. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 1700–1709, Seattle, Washington, USA. Association for Computational Linguistics.

Philipp Koehn, Hieu Hoang, Alexandra Birch, Chris Callison-Burch, Marcello Federico, Nicola Bertoldi, Brooke Cowan, Wade Shen, Christine Moran, Richard Zens, Chris Dyer, Ondřej Bojar, Alexandra Constantin, and Evan Herbst. 2007. Moses: Open source toolkit for statistical machine translation. In *Proceedings of the 45th Annual Meeting of the Association for Computational Linguistics Companion Volume Proceedings of the Demo and Poster Sessions*, pages 177–180, Prague, Czech Republic. Association for Computational Linguistics.

Philipp Koehn and Rebecca Knowles. 2017. Six challenges for neural machine translation. In *Proceedings of the First Workshop on Neural Machine Translation*, pages 28–39, Vancouver. Association for Computational Linguistics.

Philipp Koehn, Franz J. Och, and Daniel Marcu. 2003. Statistical phrase-based translation. In *Proceedings of the 2003 Human Language Technology Conference of the North American Chapter of the Association for Computational Linguistics*, pages 127–133.

Samuel Läubli, Rico Sennrich, and Martin Volk. 2018. Has machine translation achieved human parity? a case for document-level evaluation. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 4791–4796, Brussels, Belgium. Association for Computational Linguistics.

Yinhan Liu, Jiatao Gu, Naman Goyal, Xian Li, Sergey Edunov, Marjan Ghazvininejad, Mike Lewis, and Luke Zettlemoyer. 2020. Multilingual denoising pre-training for neural machine translation. *Transactions of the Association for Computational Linguistics*, 8:726–742.

Thang Luong, Hieu Pham, and Christopher D. Manning. 2015. Effective approaches to attention-based neural machine translation. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 1412–1421, Lisbon, Portugal. Association for Computational Linguistics.

Shuming Ma, Dongdong Zhang, and Ming Zhou. 2020. A simple and effective unified encoder for document-level machine translation. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 3505–3511, Online. Association for Computational Linguistics.

Pedro Henrique Martins, Zita Marinho, and André F. T. Martins. 2021. ∞-former: Infinite memory transformer.

Sameen Maruf and Gholamreza Haffari. 2018. Document context neural machine translation with memory networks. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1275–1284, Melbourne, Australia. Association for Computational Linguistics.

Sameen Maruf, André F. T. Martins, and Gholamreza Haffari. 2019. Selective attention for context-aware neural machine translation. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 3092–3102, Minneapolis, Minnesota. Association for Computational Linguistics.

Lesly Miculicich, Dhananjay Ram, Nikolaos Pappas, and James Henderson. 2018. Document-level neural machine translation with hierarchical attention networks. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2947–2954, Brussels, Belgium. Association for Computational Linguistics.

Niki Parmar, Ashish Vaswani, Jakob Uszkoreit, Lukasz Kaiser, Noam Shazeer, Alexander Ku, and Dustin Tran. 2018. Image transformer. In *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pages 4055–4064. PMLR.

Jean Pouget-Abadie, Dzmitry Bahdanau, Bart van Merriënboer, Kyunghyun Cho, and Yoshua Bengio. 2014. Overcoming the curse of sentence length for neural machine translation using automatic segmentation. In *Proceedings of SSST-8, Eighth Workshop on Syntax, Semantics and Structure in Statistical Translation*, pages 78–85, Doha, Qatar. Association for Computational Linguistics.

Jiezhong Qiu, Hao Ma, Omer Levy, Wen-tau Yih, Sinong Wang, and Jie Tang. 2020. Blockwise self-attention for long document understanding. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 2555–2565, Online. Association for Computational Linguistics.

Jack W. Rae, Anna Potapenko, Siddhant M. Jayakumar, Chloe Hillier, and Timothy P. Lillicrap. 2020. Compressive transformers for long-range sequence modelling. In *International Conference on Learning Representations*.

David E. Rumelhart, Geoffrey E. Hinton, and Ronald J. Williams. 1986. Learning Representations by Backpropagating Errors. *Nature*, 323(6088):533–536.

Yves Scherrer, Jörg Tiedemann, and Sharid Loáiciga. 2019. Analysing concatenation approaches to document-level NMT in two different domains. In *Proceedings of the Fourth Workshop on Discourse in Machine Translation (DiscoMT 2019)*, pages 51–61, Hong Kong, China. Association for Computational Linguistics.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016. Neural machine translation of rare words with subword units. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1715–1725, Berlin, Germany. Association for Computational Linguistics.

Mukund Sundararajan, Ankur Taly, and Qiqi Yan. 2017. Axiomatic attribution for deep networks. In *Proceedings of the 34th International Conference on Machine Learning*, volume 70 of *Proceedings of Machine Learning Research*, pages 3319–3328. PMLR.

Jörg Tiedemann and Yves Scherrer. 2017. Neural machine translation with extended context. In *Proceedings of the Third Workshop on Discourse in Machine Translation*, pages 82–92, Copenhagen, Denmark. Association for Computational Linguistics.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc.

Elena Voita, Rico Sennrich, and Ivan Titov. 2019a. Context-aware monolingual repair for neural machine translation. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 877–886, Hong Kong, China. Association for Computational Linguistics.

Elena Voita, Rico Sennrich, and Ivan Titov. 2019b. When a good translation is wrong in context: Context-aware machine translation improves on deixis, ellipsis, and lexical cohesion. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 1198–1212, Florence, Italy. Association for Computational Linguistics.

Zhilin Yang, Zihang Dai, Yiming Yang, Jaime G. Carbonell, Ruslan Salakhutdinov, and Quoc V. Le. 2019. Xlnet: Generalized autoregressive pretraining for language understanding. In *NeurIPS*.

Manzil Zaheer, Guru Guruganesh, Kumar Avinava Dubey, Joshua Ainslie, Chris Alberti, Santiago Ontanon, Philip Pham, Anirudh Ravula, Qifan Wang, Li Yang, et al. 2020. Big bird: Transformers for longer sequences. *Advances in Neural Information Processing Systems*, 33.

Jiacheng Zhang, Huanbo Luan, Maosong Sun, Feifei Zhai, Jingfang Xu, Min Zhang, and Yang Liu. 2018. Improving the transformer translation model with document-level context. In *Proceedings of the 2018*

Conference on Empirical Methods in Natural Language Processing, pages 533–542, Brussels, Belgium. Association for Computational Linguistics.

Pei Zhang, Boxing Chen, Niyu Ge, and Kai Fan. 2020. Long-short term masking transformer: A simple but effective baseline for document-level neural machine translation. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1081–1087, Online. Association for Computational Linguistics.

Zaixiang Zheng, Xiang Yue, Shujian Huang, Jiajun Chen, and Alexandra Birch. 2020. Towards making the most of context in neural machine translation. In *Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence, IJCAI-20*, pages 3983–3989. International Joint Conferences on Artificial Intelligence Organization. Main track.