Improving the Transformer Translation Model with Document-Level Context

Jiacheng Zhang[†], Huanbo Luan[†], Maosong Sun[†], FeiFei Zhai[#], Jingfang Xu[#], Min Zhang[§] and Yang Liu[†‡*]
[†]Institute for Artificial Intelligence
State Key Laboratory of Intelligent Technology and Systems
Department of Computer Science and Technology, Tsinghua University, Beijing, China
[‡]Beijing National Research Center for Information Science and Technology
[#]Sogou Inc., Beijing, China
[§]Soochow University, Suzhou, China

Abstract

Although the Transformer translation model (Vaswani et al., 2017) has achieved state-of-the-art performance in a variety of translation tasks, how to use document-level context to deal with discourse phenomena problematic for Transformer still remains a challenge. In this work, we extend the Transformer model with a new context encoder to represent document-level context, which is then incorporated into the original encoder and decoder. As large-scale document-level parallel corpora are usually not available, we introduce a two-step training method to take full advantage of abundant sentence-level parallel corpora and limited document-level parallel corpora. Experiments on the NIST Chinese-English datasets and the IWSLT French-English datasets show that our approach improves over Transformer significantly. [1]

1 Introduction

The past several years have witnessed the rapid development of neural machine translation (NMT) (Sutskever et al., 2014; Bahdanau et al., 2015), which investigates the use of neural networks to model the translation process. Showing remarkable superiority over conventional statistical machine translation (SMT), NMT has been recognized as the new *de facto* method and is widely used in commercial MT systems (Wu et al., 2016). A variety of NMT models have been proposed to map between natural languages such as RNNencdec (Sutskever et al., 2014), RNNsearch (Bahdanau et al., 2015), ConvS2S (Gehring et al., 2017), and Transformer (Vaswani et al., 2017). Among them, the Transformer model has achieved state-of-the-art translation performance. The capability to minimize the path length between long-distance dependencies in neural networks contributes to its exceptional performance.

However, the Transformer model still suffers from a major drawback: it performs translation only at the sentence level and ignores document-level context. Document-level context has proven to be beneficial for improving translation performance, not only for conventional SMT (Gong et al., 2011; Hardmeier et al., 2012), but also for NMT (Wang et al., 2017; Tu et al., 2018). Bawden et al. (2018) indicate that it is important to exploit document-level context to deal with context-dependent phenomena which are problematic for machine translation such as coreference, lexical cohesion, and lexical disambiguation.

While document-level NMT has attracted increasing attention from the community in the past two years (Jean et al., 2017; Kuang et al., 2017; Tiedemann and Scherrer, 2017; Wang et al., 2017; Maruf and Haffari, 2018; Bawden et al., 2018; Tu et al., 2018; Voita et al., 2018), to the best of our knowledge, only one existing work has endeavored to model document-level context for the Transformer model (Voita et al., 2018). Previous approaches to document-level NMT have concentrated on the RNNsearch model (Bahdanau et al., 2015). It is challenging to adapt these approaches to Transformer because they are designed specifically for RNNsearch.

In this work, we propose to extend the Transformer model to take advantage of document-level context. The basic idea is to use multi-head self-attention (Vaswani et al., 2017) to compute the representation of document-level context, which is then incorporated into the encoder and decoder using multi-head attention. Since large-scale document-level parallel corpora are usually hard to acquire, we propose to train sentence-level model parameters on sentence-level paral-

*Corresponding author: Yang Liu.

[1]The source code is available at https://github.com/Glaceon31/Document-Transformer

Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, pages 533–542
Brussels, Belgium, October 31 - November 4, 2018.

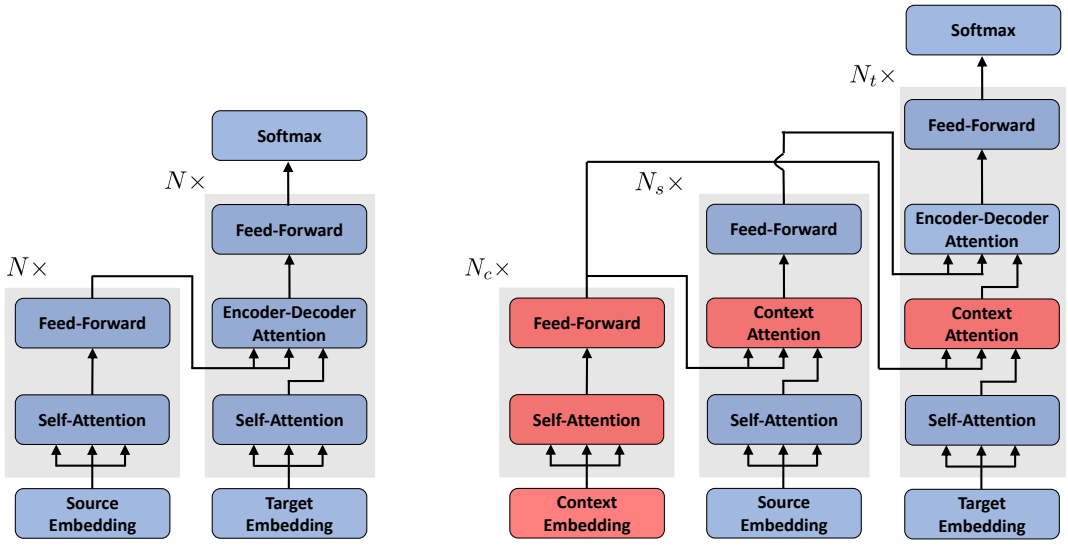

Figure 1: (a) The original Transformer translation model (Vaswani et al., 2017) and (b) the extended Transformer translation model that exploits document-level context. The newly introduced modules are highlighted in red.

lel corpora first and then estimate document-level model parameters on document-level parallel corpora while keeping the learned original sentence-level Transformer model parameters fixed. Our approach has the following advantages:

1. *Increased capability to capture context*: the use of multi-head attention, which significantly reduces the path length between long-range dependencies, helps to improve the capability to capture document-level context;

2. *Small computational overhead*: as all newly introduced modules are based on highly parallelizable multi-head attention, there is no significant slowdown in both training and decoding;

3. *Better use of limited labeled data*: our approach is capable of maintaining the superiority over the sentence-level counterpart even when only small-scale document-level parallel corpora are available.

Experiments show that our approach achieves an improvement of 1.96 and 0.89 BLEU points over Transformer on Chinese-English and French-English translation respectively by exploiting document-level context. It also outperforms a state-of-the-art cache-based method (Kuang et al., 2017) adapted for Transformer.

2 Approach

2.1 Problem Statement

Our goal is to enable the Transformer translation model (Vaswani et al., 2017) as shown in Figure 1(a) to exploit document-level context.

Formally, let $\mathbf{X} = \mathbf{x}^{(1)}, \ldots, \mathbf{x}^{(k)}, \ldots, \mathbf{x}^{(K)}$ be a source-language document composed of K source sentences. We use $\mathbf{x}^{(k)} = x_1^{(k)}, \ldots, x_i^{(k)}, \ldots, x_I^{(k)}$ to denote the k-th source sentence containing I words. $x_i^{(k)}$ denotes the i-th word in the k-th source sentence. Likewise, the corresponding target-language document is denoted by $\mathbf{Y} = \mathbf{y}^{(1)}, \ldots, \mathbf{y}^{(k)}, \ldots, \mathbf{y}^{(K)}$ and $\mathbf{y}^{(k)} = y_1^{(k)}, \ldots, y_j^{(k)}, \ldots, y_J^{(k)}$ represents the k-th target sentence containing J words. $y_j^{(k)}$ denotes the j-th word in the k-th target sentence. We assume that $\langle \mathbf{X}, \mathbf{Y} \rangle$ constitutes a *parallel document* and each $\langle \mathbf{x}^{(k)}, \mathbf{y}^{(k)} \rangle$ forms a *parallel sentence*.

Therefore, the document-level translation probability is given by

$$P(\mathbf{Y}|\mathbf{X};\boldsymbol{\theta}) = \prod_{k=1}^{K} P(\mathbf{y}^{(k)}|\mathbf{X}, \mathbf{Y}_{<k};\boldsymbol{\theta}), \quad (1)$$

where $\mathbf{Y}_{<k} = \mathbf{y}^{(1)}, \ldots, \mathbf{y}^{(k-1)}$ is a partial translation.

For generating $\mathbf{y}^{(k)}$, the source document $\mathbf{X}$ can be divided into three parts: (1) the k-th source sentence $\mathbf{X}_{=k} = \mathbf{x}^{(k)}$, (2) the source-side document-

level context on the left $\mathbf{X}_{<k} = \mathbf{x}^{(1)}, \ldots, \mathbf{x}^{(k-1)}$, and (3) the source-side document-level context on the right $\mathbf{X}_{>k} = \mathbf{x}^{(k+1)}, \ldots, \mathbf{x}^{(K)}$. As the languages used in our experiments (i.e., Chinese and English) are written left to right, we omit $\mathbf{X}_{>k}$ for simplicity.

We also omit the target-side document-level context $\mathbf{Y}_{<k}$ due to the translation error propagation problem (Wang et al., 2017): errors made in translating one sentence will be propagated to the translation process of subsequent sentences. Interestingly, we find that using source-side document-level context $\mathbf{X}_{<k}$, which conveys the same information with $\mathbf{Y}_{<k}$, helps to compute better representations on the target side (see Table 8).

As a result, the document-level translation probability can be approximated as

$$\begin{aligned} & P(\mathbf{Y}|\mathbf{X};\boldsymbol{\theta}) \\ \approx & \prod_{k=1}^{K} P(\mathbf{y}^{(k)}|\mathbf{X}_{<k}, \mathbf{x}^{(k)};\boldsymbol{\theta}), \quad (2) \\ = & \prod_{k=1}^{K}\prod_{j=1}^{J} P(y_j^{(k)}|\mathbf{X}_{<k}, \mathbf{x}^{(k)}, \mathbf{y}_{<j}^{(k)};\boldsymbol{\theta}), \quad (3) \end{aligned}$$

where $\mathbf{y}_{<j}^{(k)} = y_1^{(k)}, \ldots, y_{j-1}^{(k)}$ is a partial translation.

In this way, the document-level translation model can still be defined at the sentence level without sacrificing efficiency except that the source-side document-level context $\mathbf{X}_{<k}$ (or *context* for short) is taken into account.

In the following, we will introduce how to represent the context (Section 2.2), how to integrate the context (Section 2.3), and how to train the model especially when only limited training data is available (Section 2.4).

2.2 Document-level Context Representation

As document-level context often includes several sentences, it is important to capture long-range dependencies and identify relevant information. We use multi-head self-attention (Vaswani et al., 2017) to compute the representation of document-level context because it is capable of reducing the maximum path length between long-range dependencies to $O(1)$ (Vaswani et al., 2017) and determining the relative importance of different locations in the context (Bahdanau et al., 2015). Because of this property, multi-head self-attention has proven to be effective in other NLP tasks such as constituency parsing (Kitaev and Klein, 2018).

As shown in Figure 1(b), we use a self-attentive encoder to compute the representation of $\mathbf{X}_{<k}$. The input to the self-attentive encoder is a sequence of context word embeddings, represented as a matrix. Suppose $\mathbf{X}_{<k}$ is composed of M source words: $\mathbf{X}_{<k} = x_1, \ldots, x_m, \ldots, x_M$. We use $\boldsymbol{x}_m \in \mathbb{R}^{D\times 1}$ to denote the vector representation of x_m that is the sum of word embedding and positional encoding (Vaswani et al., 2017). Therefore, the matrix representation of $\mathbf{X}_{<k}$ is given by

$$\boldsymbol{X}_c = [\boldsymbol{x}_1; \ldots; \boldsymbol{x}_M], \quad (4)$$

where $\boldsymbol{X}_c \in \mathbb{R}^{D\times M}$ is the concatenation of all vector representations of all source contextual words.

The self-attentive encoder is composed of a stack of N_c identical layers. Each layer has two sub-layers. The first sub-layer is a multi-head self-attention:

$$\mathbf{A}^{(1)} = \mathrm{MultiHead}(\boldsymbol{X}_c, \boldsymbol{X}_c, \boldsymbol{X}_c), \quad (5)$$

where $\mathbf{A}^{(1)} \in \mathbb{R}^{D\times M}$ is the hidden state calculated by the multi-head self-attention at the first layer, $\mathrm{MultiHead}(\mathbf{Q}, \mathbf{K}, \mathbf{V})$ is a multi-head self-attention function that takes a query matrix $\mathbf{Q}$, a key matrix $\mathbf{K}$, and a value matrix $\mathbf{V}$ as inputs. In this case, $\mathbf{Q} = \mathbf{K} = \mathbf{V} = \boldsymbol{X}_c$. This is why it is called self-attention. Please refer to (Vaswani et al., 2017) for more details.

Note that we follow Vaswani et al. (2017) to use residual connection and layer normalization in each sub-layer, which are omitted in the presentation for simplicity. For example, the actual output of the first sub-layer is:

$$\mathrm{LayerNorm}(\mathbf{A}^{(1)} + \boldsymbol{X}_c). \quad (6)$$

The second sub-layer is a simple, position-wise fully connected feed-forward network:

$$\mathbf{C}^{(1)} = \left[\mathrm{FNN}(\mathbf{A}_{\cdot,1}^{(1)}); \ldots; \mathrm{FNN}(\mathbf{A}_{\cdot,M}^{(1)})\right] \quad (7)$$

where $\mathbf{C}^{(1)} \in \mathbb{R}^{D\times M}$ is the annotation of $\mathbf{X}_{<k}$ after the first layer, $\mathbf{A}_{\cdot,m}^{(1)} \in \mathbb{R}^{D\times 1}$ is the column vector for the m-th contextual word, and $\mathrm{FNN}(\cdot)$ is a position-wise fully connected feed-forward network (Vaswani et al., 2017).

This process iterates N_c times as follows:

$$\mathbf{A}^{(n)} = \mathrm{MultiHead}\left(\mathbf{C}^{(n-1)}, \mathbf{C}^{(n-1)}, \mathbf{C}^{(n-1)}\right), \quad (8)$$

$$\mathbf{C}^{(n)} = \left[\mathrm{FNN}(\mathbf{A}_{\cdot,1}^{(n)}); \ldots; \mathrm{FNN}(\mathbf{A}_{\cdot,M}^{(n)})\right], \quad (9)$$

where $\mathbf{A}^{(n)}$ and $\mathbf{C}^{(n)}$ $(n = 1, \ldots, N_c)$ are the hidden state and annotation at the n-th layer, respectively. Note that $\mathbf{C}^{(0)} = \boldsymbol{X}_c$.

2.3 Document-level Context Integration

We use multi-head attention to integrate $\mathbf{C}^{(N_c)}$, which is the representation of $\mathbf{X}_{<k}$, into both the encoder and the decoder.

2.3.1 Integration into the Encoder

Given the k-th source sentence $\mathbf{x}^{(k)}$, we use $\boldsymbol{x}_i^{(k)} \in \mathbb{R}^{D\times 1}$ to denote the vector representation of the i-th source word $x_i^{(k)}$, which is a sum of word embedding and positional encoding. Therefore, the initial matrix representation of $\mathbf{x}^{(k)}$ is

$$\boldsymbol{X} = [\boldsymbol{x}_1^{(k)}; \ldots; \boldsymbol{x}_I^{(k)}], \tag{10}$$

where $\boldsymbol{X} \in \mathbb{R}^{D\times I}$ is the concatenation of all vector representations of source words.

As shown in Figure 1(b), we follow (Vaswani et al., 2017) to use a stack of N_s identical layers to encode $\mathbf{x}^{(k)}$. Each layer consists of three sub-layers. The first sub-layer is a multi-head self-attention:

$$\mathbf{B}^{(n)} = \mathrm{MultiHead}\Big(\mathbf{S}^{(n-1)}, \mathbf{S}^{(n-1)}, \mathbf{S}^{(n-1)}\Big), \tag{11}$$

where $\mathbf{S}^{(0)} = \boldsymbol{X}$. The second sub-layer is context attention that integrates document-level context into the encoder:

$$\mathbf{D}^{(n)} = \mathrm{MultiHead}\Big(\mathbf{B}^{(n)}, \mathbf{C}^{(N_c)}, \mathbf{C}^{(N_c)}\Big). \tag{12}$$

The third sub-layer is a position-wise fully connected feed-forward neural network:

$$\mathbf{S}^{(n)} = \Big[\mathrm{FNN}(\mathbf{D}_{\cdot,1}^{(n)}); \ldots; \mathrm{FNN}(\mathbf{D}_{\cdot,I}^{(n)})\Big], \tag{13}$$

where $\mathbf{S}^{(n)} \in \mathbb{R}^{D\times I}$ is the representation of the source sentence $\mathbf{x}^{(k)}$ at the n-th layer $(n = 1, \ldots, N_s)$.

2.3.2 Integration into the Decoder

When generating the j-th target word $y_j^{(k)}$, the partial translation is denoted by $\mathbf{y}_{<j}^{(k)} = y_1^{(k)}, \ldots, y_{j-1}^{(k)}$. We follow Vaswani et al. (2017) to offset the target word embeddings by one position, resulting in the following matrix representation of $\mathbf{y}_{<j}^{(k)}$:

$$\boldsymbol{Y} = [\boldsymbol{y}_0^{(k)}, \ldots, \boldsymbol{y}_{j-1}^{(k)}], \tag{14}$$

where $\mathbf{y}_0^{(k)} \in \mathbb{R}^{D\times 1}$ is the vector representation of a begin-of-sentence token and $\boldsymbol{Y} \in \mathbb{R}^{D\times j}$ is the concatenation of all vectors.

As shown in Figure 1(b), we follow (Vaswani et al., 2017) to use a stack of N_t identical layers to compute target-side representations. Each layer is composed of four sub-layers. The first sub-layer is a multi-head self-attention:

$$\mathbf{E}^{(n)} = \mathrm{MultiHead}\Big(\mathbf{T}^{(n-1)}, \mathbf{T}^{(n-1)}, \mathbf{T}^{(n-1)}\Big), \tag{15}$$

where $\mathbf{T}^{(0)} = \boldsymbol{Y}$. The second sub-layer is context attention that integrates document-level context into the decoder:

$$\mathbf{F}^{(n)} = \mathrm{MultiHead}\Big(\mathbf{E}^{(n)}, \mathbf{C}^{(N_c)}, \mathbf{C}^{(N_c)}\Big). \tag{16}$$

The third sub-layer is encoder-decoder attention that integrates the representation of the corresponding source sentence:

$$\mathbf{G}^{(n)} = \mathrm{MultiHead}\Big(\mathbf{F}^{(n)}, \mathbf{S}^{(N_s)}, \mathbf{S}^{(N_s)}\Big). \tag{17}$$

The fourth sub-layer is a position-wise fully connected feed-forward neural network:

$$\mathbf{T}^{(n)} = \Big[\mathrm{FNN}(\mathbf{G}_{\cdot,1}^{(n)}); \ldots; \mathrm{FNN}(\mathbf{G}_{\cdot,j}^{(n)}),\Big], \tag{18}$$

where $\mathbf{T}^{(n)} \in \mathbb{R}^{D\times j}$ is the representation at the n-th layer $(n = 1, \ldots, N_t)$. Note that $\mathbf{T}^{(0)} = \boldsymbol{Y}$.

Finally, the probability distribution of generating the next target word $y_j^{(k)}$ is defined using a softmax layer:

$$P(y_j^{(k)}|\mathbf{X}_{<k}, \mathbf{x}^{(k)}, \mathbf{y}_{<j}^{(k)}; \boldsymbol{\theta}) \propto \exp(\mathbf{W}_o \mathbf{T}_{\cdot,j}^{(N_t)}) \tag{19}$$

where $\mathbf{W}_o \in \mathbb{R}^{|\mathcal{V}_y|\times D}$ is a model parameter, $\mathcal{V}_y$ is the target vocabulary, and $\mathbf{T}_{\cdot,j}^{(N_t)} \in \mathbb{R}^{D\times 1}$ is a column vector for predicting the j-th target word.

2.3.3 Context Gating

In our model, we follow Vaswani et al. (2017) to use residual connections (He et al., 2016) around each sub-layer to shortcut its input to its output:

$$\mathrm{Residual}(\mathbf{H}) = \mathbf{H} + \mathrm{SubLayer}(\mathbf{H}), \tag{20}$$

where $\mathbf{H}$ is the input of the sub-layer.

While residual connections prove to be effective for building deep architectures, there is one potential problem for our model: the residual connections after the context attention sub-layer might increase the influence of document-level context

$\mathbf{X}_{<k}$ in an uncontrolled way. This is undesirable because the source sentence $\mathbf{x}^{(k)}$ usually plays a more important role in target word generation.

To address this problem, we replace the residual connections after the context attention sub-layer with a *position-wise context gating* sub-layer:

$$\text{Gating}(\mathbf{H}) = \lambda\mathbf{H} + (1-\lambda)\text{SubLayer}(\mathbf{H}). \quad (21)$$

The gating weight is given by

$$\lambda = \sigma(\mathbf{W}_i\mathbf{H} + \mathbf{W}_s\text{SubLayer}(\mathbf{H})), \quad (22)$$

where $\sigma(\cdot)$ is a sigmoid function, $\mathbf{W}_i$ and $\mathbf{W}_s$ are model parameters.

2.4 Training

Given a document-level parallel corpus D_d, the standard training objective is to maximize the log-likelihood of the training data:

$$\hat{\boldsymbol{\theta}} = \arg\max_{\boldsymbol{\theta}} \left\{ \sum_{\langle \mathbf{X},\mathbf{Y}\rangle \in D_d} \log P(\mathbf{Y}|\mathbf{X};\boldsymbol{\theta}) \right\}. \quad (23)$$

Unfortunately, large-scale document-level parallel corpora are usually unavailable, even for resource-rich languages such as English and Chinese. Under small-data training conditions, document-level NMT is prone to underperform sentence-level NMT because of poor estimates of low-frequency events.

To address this problem, we adopt the idea of freezing some parameters while tuning the remaining part of the model (Jean et al., 2015; Zoph et al., 2016). We propose a two-step training strategy that uses an additional sentence-level parallel corpus D_s, which can be larger than D_d. We divide model parameters into two subsets: $\boldsymbol{\theta} = \boldsymbol{\theta}_s \cup \boldsymbol{\theta}_d$, where $\boldsymbol{\theta}_s$ is a set of original sentence-level model parameters (highlighted in blue in Figure 1(b)) and $\boldsymbol{\theta}_d$ is a set of newly-introduced document-level model parameters (highlighted in red in Figure 1(b)).

In the first step, sentence-level parameters $\boldsymbol{\theta}_s$ are estimated on the combined sentence-level parallel corpus $D_s \cup D_d$: [2]

$$\hat{\boldsymbol{\theta}}_s = \arg\max_{\boldsymbol{\theta}_s} \sum_{\langle \mathbf{x},\mathbf{y}\rangle \in D_s \cup D_d} \log P(\mathbf{y}|\mathbf{x};\boldsymbol{\theta}_s). \quad (24)$$

Note that the newly introduced modules (highlighted in red in Figure 1(b)) are inactivated in this step. $P(\mathbf{y}|\mathbf{x};\boldsymbol{\theta}_s)$ is identical to the original Transformer model, which is a special case of our model.

In the second step, document-level parameters $\boldsymbol{\theta}_d$ are estimated on the document-level parallel corpus D_d only:

$$\hat{\boldsymbol{\theta}}_d = \arg\max_{\boldsymbol{\theta}_d} \sum_{\langle \mathbf{X},\mathbf{Y}\rangle \in D_d} \log P(\mathbf{Y}|\mathbf{X};\hat{\boldsymbol{\theta}}_s,\boldsymbol{\theta}_d). \quad (25)$$

Our approach is also similar to pre-training which has been widely used in NMT (Shen et al., 2016; Tu et al., 2018). The major difference is that our approach keeps $\hat{\boldsymbol{\theta}}_s$ *fixed* when estimating $\boldsymbol{\theta}_d$ to prevent the model from overfitting on the relatively smaller document-level parallel corpora.

3 Experiments

3.1 Setup

We evaluate our approach on Chinese-English and French-English translation tasks. In Chinese-English translation task, the training set contains 2M Chinese-English sentence pairs with 54.8M Chinese words and 60.8M English words. [3] The document-level parallel corpus is a subset of the full training set, including 41K documents with 940K sentence pairs. On average, each document in the training set contains 22.9 sentences. We use the NIST 2006 dataset as the development set and the NIST 2002, 2003, 2004, 2005, 2008 datasets as test sets. The development and test sets contain 588 documents with 5,833 sentences. On average, each document contains 9.9 sentences.

In French-English translation task, we use the IWSLT bilingual training data (Mauro et al., 2012) which contains 1,824 documents with 220K sentence pairs as training set. For development and testing, we use the IWSLT 2010 development and test sets, which contains 8 documents with 887 sentence pairs and 11 documents with 1,664 sentence pairs respectively. The evaluation metric for both tasks is case-insensitive BLEU score as calculated by the *multi-bleu.perl* script.

In preprocessing, we use byte pair encoding (Sennrich et al., 2016) with 32K merges to segment words into sub-word units for all languages. For the original Transformer model and our extended model, the hidden size is set to 512 and the

[2] It is easy to create a sentence-level parallel corpus from D_d.

[3] The training set consists of sentence-level parallel corpora LDC2002E18, LDC2003E07, LDC2003E14, news part of LDC2004T08 and document-level parallel corpora LDC2002T01, LDC2004T07, LDC2005T06, LDC2005T10, LDC2009T02, LDC2009T15, LDC2010T03.

# sent.	1	2	3
MT06	49.38	**49.69**	49.49

Table 1: Effect of context length on translation quality. The BLEU scores are calculated on the development set.

# Layer	MT06
1	**49.69**
2	49.38
3	49.54
4	49.59
5	49.31
6	49.43

Table 2: Effect of self-attention layer number (i.e., N_c) on translation quality. The BLEU scores are calculated on the development set.

filter size is set to 2,048. The multi-head attention has 8 individual attention heads. We set $N = N_s = N_t = 6$. In training, we use Adam (Kingma and Ba, 2015) for optimization. Each mini-batch contains approximately 24K words. We use the learning rate decay policy described by Vaswani et al. (2017). In decoding, the beam size is set to 4. We use the length penalty (Wu et al., 2016) and set the hyper-parameter α to 0.6. We use four Tesla P40 GPUs for training and one Tesla P40 GPU for decoding. We implement our approach on top of the open-source toolkit THUMT (Zhang et al., 2017). [4]

3.2 Effect of Context Length

We first investigate the effect of context length (i.e., the number of preceding sentences) on our approach. As shown in Table 1, using two preceding source sentences as document-level context achieves the best translation performance on the development set. Using more preceding sentences does not bring any improvement and increases computational cost. This confirms the finding of Tu et al. (2018) that long-distance context only has limited influence. Therefore, we set the number of preceding sentences to 2 in the following experiments. [5]

3.3 Effect of Self-Attention Layer Number

Table 2 shows the effect of self-attention layer number for computing representations of document-level context (see Section 2.2) on translation quality. Surprisingly, using only one self-attention layer suffices to achieve good performance. Increasing the number of self-attention layers does not lead to any improvements. Therefore, we set N_c to 1 for efficiency.

3.4 Comparison with Previous Work

In Chinese-English translation task, we compare our approach with the following previous methods:

1. (Wang et al., 2017): using a hierarchical RNN to integrate document-level context into the RNNsearch model. They use a document-level parallel corpus containing 1M sentence pairs. Table 3 gives the BLEU scores reported in their paper.

2. (Kuang et al., 2017): using a cache which stores previous translated words and topical words to incorporate document-level context into the RNNsearch model. They use a document-level parallel corpus containing 2.8M sentence pairs. Table 3 gives the BLEU scores reported in their paper.

3. (Vaswani et al., 2017): the state-of-the-art NMT model that does not exploit document-level context. We use the open-source toolkit THUMT (Zhang et al., 2017) to train and evaluate the model. The training dataset is our sentence-level parallel corpus containing 2M sentence pairs.

4. (Kuang et al., 2017)*: adapting the cache-based method to the Transformer model. We implement it on top of the open-source toolkit THUMT. We also use the same training data (i.e., 2M sentence pairs) and the same two-step training strategy to estimate sentence- and document-level parameters separately.

As shown in Table 3, using the same data, our approach achieves significant improvements over the original Transformer model (Vaswani et al., 2017) ($p < 0.01$). The gain on the concatenated test set (i.e., "All") is 1.96 BLEU points. It also outperforms the cache-based method (Kuang et al., 2017) adapted for Transformer significantly ($p < 0.01$), which also uses the two-step training strategy. Table 4 shows that our model also outperforms Transformer by 0.89 BLEU points on French-English translation task.

[4] https://github.com/thumt/THUMT

[5] If there is no preceding sentence, we simply use a single begin-of-sentence token.

Method	Model	MT06	MT02	MT03	MT04	MT05	MT08	All
(Wang et al., 2017)	RNNsearch	37.76	-	-	-	36.89	27.57	-
(Kuang et al., 2017)	RNNsearch	-	34.41	-	38.40	32.90	31.86	-
(Vaswani et al., 2017)	Transformer	48.09	48.63	47.54	47.79	48.34	38.31	45.97
(Kuang et al., 2017)*	Transformer	48.14	48.97	48.05	47.91	48.53	38.38	46.37
this work	Transformer	**49.69**	**50.96**	**50.21**	**49.73**	**49.46**	**39.69**	**47.93**

Table 3: Comparison with previous works on Chinese-English translation task. The evaluation metric is case-insensitive BLEU score. (Wang et al., 2017) use a hierarchical RNN to incorporate document-level context into RNNsearch. (Kuang et al., 2017) use a cache to exploit document-level context for RNNsearch. (Kuang et al., 2017)* is an adapted version of the cache-based method for Transformer. Note that "MT06" is not included in "All".

Method	Dev	Test
Transformer	29.42	35.15
this work	**30.40**	**36.04**

Table 4: Comparison with Transformer on French-English translation task. The evaluation metric is case-insensitive BLEU score.

	>	=	<
Human 1	24%	45%	31%
Human 2	20%	55%	25%
Human 3	12%	52%	36%
Overall	19%	51%	31%

Table 5: Subjective evaluation of the comparison between the original Transformer model and our model. ">" means that Transformer is better than our model, "=" means equal, and "<" means worse.

3.5 Subjective Evaluation

We also conducted a subjective evaluation to validate the benefit of exploiting document-level context. All three human evaluators were asked to compare the outputs of the original Transformer model and our model of 20 documents containing 198 sentences, which were randomly sampled from the test sets.

Table 5 shows the results of subjective evaluation. Three human evaluators generally made consistent judgements. On average, around 19% of Transformer's translations are better than that of our model, 51% are equal, and 31% are worse. This evaluation confirms that exploiting document-level context helps to improve translation quality.

3.6 Evaluation of Efficiency

We evaluated the efficiency of our approach. It takes the original Transformer model about 6.7 hours to converge during training and the training speed is 41K words/second. The decoding speed is 872 words/second. In contrast, it takes our model about 7.8 hours to converge in the second step of training. The training speed is 31K words/second. The decoding speed is 364 words/second.

Method	Training	Decoding
Transformer	41K	872
this work	31K	364

Table 6: Evaluation of training and decoding speed. The speed is measured in terms of word/second (wps).

Therefore, the training speed is only reduced by 25% thanks to the high parallelism of multi-head attention used to incorporate document-level context. The gap is larger in decoding because target words are generated in an autoregressive way in Transformer.

3.7 Effect of Two-Step Training

Table 7 shows the effect of the proposed two-step training strategy. The first two rows only use sentence-level parallel corpus to train the original Transformer model (see Eq. 24) and achieve BLEU scores of 39.53 and 45.97. The third row only uses the document-level parallel corpus to directly train our model (see Eq. 23) and achieves a BLEU score of 36.52. The fourth and fifth rows use the two-step strategy to take advantage of both sentence- and document-level parallel corpora and achieve BLEU scores of 40.22 and 47.93, respectively.

We find that document-level NMT achieves much worse results than sentence-level NMT (i.e., 36.52 vs. 39.53) when only small-scale document-level parallel corpora are available. Our two-step training method is capable of addressing this problem by exploiting sentence-level corpora, which

sent.	doc.	MT06	MT02	MT03	MT04	MT05	MT08	All
940K	-	36.20	42.41	43.12	41.02	40.93	31.49	39.53
2M	-	48.09	48.63	47.54	47.79	48.34	38.31	45.97
-	940K	34.00	38.83	40.51	38.30	36.69	29.38	36.52
940K	940K	37.12	43.29	43.70	41.42	41.84	32.36	40.22
2M	940K	49.69	50.96	50.21	49.73	49.46	39.69	47.93

Table 7: Effect of two-step training. "sent." denotes sentence-level parallel corpus and "doc." denotes document-level parallel corpus.

Integration	MT06	MT02	MT03	MT04	MT05	MT08	All
none	48.09	48.63	47.54	47.79	48.34	38.31	45.97
encoder	48.88	50.30	49.34	48.81	49.75	39.55	47.51
decoder	49.10	50.31	49.83	49.35	49.29	39.07	47.48
both	49.69	50.96	50.21	49.73	49.46	39.69	47.93

Table 8: Effect of context integration. "none" means that no document-level context is integrated, "encoder" means that the document-level context is integrated only into the encoder, "decoder" means that the document-level context is integrated only into the decoder, and "both" means that the context is integrated into both the encoder and the decoder.

Gating	MT06	MT02	MT03	MT04	MT05	MT08	All
w/o	49.33	50.56	49.74	49.29	50.11	39.02	47.55
w/	49.69	50.96	50.21	49.73	49.46	39.69	47.93

Table 9: Effect of context gating.

leads to significant improvements across all test sets.

3.8 Effect of Context Integration

Table 8 shows the effect of integrating document-level context to the encoder and decoder (see Section 2.3). It is clear that integrating document-level context into the encoder (Eq. 12) brings significant improvements (i.e., 45.97 vs. 47.51). Similarly, it is also beneficial to integrate document-level context into the decoder (Eq. 16). Combining both leads to further improvements. This observation suggests that document-level context does help to improve Transformer.

3.9 Effect of Context Gating

As shown in Table 9, we also validated the effectiveness of context gating (see Section 2.3.3). We find that replacing residual connections with context gating leads to an overall improvement of 0.38 BLEU point.

3.10 Analysis

We use an example to illustrate how document-level context helps translation (Table 10). In order to translate the source sentence, NMT has to disambiguate the multi-sense word "*yundong*", which is actually impossible without the document-level context. The exact meaning of "*rezhong*" is also highly context dependent. Fortunately, the sense of "*yundong*" can be inferred from the word "*saiche*" (car racing) in the document-level context and "*rezhong*" is the antonym of "*yanjuan*" (tired of). This example shows that our model learns to resolve word sense ambiguity and lexical cohesion problems by integrating document-level context.

4 Related Work

Developing document-level models for machine translation has been an important research direction, both for conventional SMT (Gong et al., 2011; Hardmeier et al., 2012; Xiong et al., 2013a,b; Garcia et al., 2014) and NMT (Jean et al., 2017; Kuang et al., 2017; Tiedemann and Scherrer, 2017; Wang et al., 2017; Maruf and Haffari, 2018; Bawden et al., 2018; Tu et al., 2018; Voita et al., 2018).

Most existing work on document-level NMT has focused on integrating document-level context into the RNNsearch model (Bahdanau et al.,

Context	$\cdots$*ziji ye yinwei queshao jingzheng duishou er dui* ***saiche*** *youxie* ***yanjuan*** *shi*$\cdots$
Source	*wo rengran feichang* ***rezhong*** *yu zhexiang* ***yundong***.
Reference	I'm still very **fond of** the **sport**.
Transformer	I am still very **enthusiastic about** this **movement**.
Our work	I am still very **keen on** this **sport**.

Table 10: An example of Chinese-English translation. In the source sentence, "*yundong*" (sport or political movement) is a multi-sense word and "*rezhong*" (fond of) is an emotional word whose meaning is dependent on its context. Our model takes advantage of the words "*saiche*" (car racing) and "*yanjuan*" (tired of) in the document-level context to translate the source words correctly.

2015). These approaches can be roughly divided into two broad categories: computing the representation of the full document-level context (Jean et al., 2017; Tiedemann and Scherrer, 2017; Wang et al., 2017; Maruf and Haffari, 2018; Voita et al., 2018) and using a cache to memorize most relevant information in the document-level context (Kuang et al., 2017; Tu et al., 2018). Our approach falls into the first category. We use multi-head attention to represent and integrate document-level context.

Voita et al. (2018) also extended Transformer to model document-level context, but our work is different in modeling and training strategies. The experimental part is also different. While Voita et al. (2018) focus on anaphora resolution, our model is able to improve the overall translation quality by integrating document-level context.

5 Conclusion

We have presented a method for exploiting document-level context inside the state-of-the-art neural translation model Transformer. Experiments on Chinese-English and French-English translation tasks show that our method is able to improve over Transformer significantly. In the future, we plan to further validate the effectiveness of our approach on more language pairs.

Acknowledgments

Yang Liu is supported by the National Natural Science Foundation of China (No. 61432013), National Key R&D Program of China (No. 2017YFB0202204), National Natural Science Foundation of China (No. 61761166008), Advanced Innovation Center for Language Resources (TYR17002), and the NExT++ project supported by the National Research Foundation, Prime Ministers Office, Singapore under its IRC@Singapore Funding Initiative. This research is also supported by Sogou Inc.

References

Dzmitry Bahdanau, KyungHyun Cho, and Yoshua Bengio. 2015. Sequence to sequence learning with neural networks. In *Proceedings of ICLR*.

Rachel Bawden, Rico Sennrich, Alexandra Birch, and Barry Haddow. 2018. Evaluating discourse phenomena in neural machine translation. In *Proceedings of NAACL*.

Eva Martínez Garcia, Cristina Espãna Bonet, and Lluíz Màrquez. 2014. Document-level machine translation with word vector models. In *Proceedings of EACL*.

Jonas Gehring, Michael Auli, David Grangier, Denis Yarats, and Yann N Dauphin. 2017. Convolutional sequence to sequence learning. *CoRR*, abs/1705.03122.

Zhengxian Gong, Min Zhang, and Guodong Zhou. 2011. Cache-based document-level statistical machine translation. In *Proceedings of EMNLP*.

Christian Hardmeier, Joakim Nivre, and Jörg Tiedemann. 2012. Document-wide decoding for phrase-based statistical machine translation. In *Proceedings of EMNLP*.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016. Deep residual learning for image recognition. In *Proceedings of CVPR*.

Sébastien Jean, Kyunghyun Cho, Roland Memisevic, and Yoshua Bengio. 2015. On using very large target vocabulary for neural machine translation. In *Proceedings of ACL*.

Sebastien Jean, Stanislas Lauly, Orhan Firat, and Kyunghyun Cho. 2017. Does neural machine translation benefit from larger context? *CoRR*, abs/1704.05135.

Diederik P Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization.

Nikita Kitaev and Dan Klein. 2018. Constituency parsing with a self-attentive encoder.

Shaohui Kuang, Deyi Xiong, Weihua Luo, and Guodong Zhou. 2017. Cache-based document-level neural machine translation. *CoRR*, abs/1711.11221.

Sameen Maruf and Gholamreza Haffari. 2018. Document context neural machine translation with memory networks. In *Proceedings of ACL*.

Cettolo Mauro, Girardi Christian, and Federico Marcello. 2012. Wit3: Web inventory of transcribed and translated talks. In *Proceedings of EAMT*.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016. Neural machine translation of rare words with subword units. In *Proceedings of ACL*.

Shiqi Shen, Yong Cheng, Zhongjun He, Wei He, Hua Wu, Maosong Sun, and Yang Liu. 2016. Minimum risk training for neural machine translation. In *Proceedings of ACL*.

Ilya Sutskever, Oriol Vinyals, and Quoc V. Le. 2014. Sequence to sequence learning with neural networks. In *Proceedings of NIPS*.

Jörg Tiedemann and Yves Scherrer. 2017. Neural machine translation with extended context. In *Proceedings of the Third Workshop on Discourse in Machine Translation*.

Zhaopeng Tu, Yang Liu, Shuming Shi, and Tong Zhang. 2018. Learning to remember translation history with a continuous cache. *Transactions of the Association for Computational Linguistics*.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Proceedings of NIPS*.

Elena Voita, Pavel Serdyukov, Rico Sennrich, and Ivan Titov. 2018. Context-aware neural machine translation learns anaphora resolution. In *Proceedings of ACL*.

Longyue Wang, Zhaopeng Tu, Andy Way, and Liu Qun. 2017. Exploiting cross-sentence context for neural machine translation. In *Proceedings of EMNLP*.

Yonghui Wu, Mike Schuster, Zhifeng Chen, Quoc V Le, Mohammad Norouzi, Wolfgang Macherey, Maxim Krikun, et al. 2016. Google's neural machine translation system: Bridging the gap between human and machine translation. *arXiv preprint arXiv:1609.08144*.

Deyi Xiong, Guosheng Ben, Min Zhang, Yajuan Lv, and Qun Liu. 2013a. Modeling lexical cohesion for document-level machine translation. In *Proceedings of IJCAI*.

Deyi Xiong, Yang Ding, Min Zhang, and Chew Lim Tan. 2013b. Lexical chain based cohesion models for document-level statistical machine translation. In *Proceedings of EMNLP*.

Jiacheng Zhang, Yanzhuo Ding, Shiqi Shen, Yong Cheng, Maosong Sun, Huanbo Luan, and Yang Liu. 2017. Thumt: An open source toolkit for neural machine translation. *arXiv preprint arXiv:1706.06415*.

Barret Zoph, Deniz Yuret, Jonathan May, and Kevin Knight. 2016. Transfer learning for low-resource neural machine translation. In *Proceedings of EMNLP*.