Word Attention for Sequence to Sequence Text Understanding

Lijun Wu[1]*, **Fei Tian**[2], **Li Zhao**[2], **Jianhuang Lai**[1,3] and **Tie-Yan Liu**[2]
[1]School of Data and Computer Science, Sun Yat-sen University
[2]Microsoft Research
[3]Guangdong Key Laboratory of Information Security Technology
wulijun3@mail2.sysu.edu.cn; {fetia, lizo, tie-yan.liu}@microsoft.com; stsljh@mail.sysu.edu.cn

Abstract

Attention mechanism has been a key component in Recurrent Neural Networks (RNNs) based sequence to sequence learning framework, which has been adopted in many text understanding tasks, such as neural machine translation and abstractive summarization. In these tasks, the attention mechanism models how important each part of the source sentence is to generate a target side word. To compute such importance scores, the attention mechanism summarizes the source side information in the encoder RNN hidden states (i.e., h_t), and then builds a context vector for a target side word upon a subsequence representation of the source sentence, since h_t actually summarizes the information of the subsequence containing the first t-th words in the source sentence. We in this paper, show that an additional attention mechanism called *word attention*, that builds itself upon word level representations, significantly enhances the performance of sequence to sequence learning. Our word attention can enrich the source side contextual representation by directly promoting the clean word level information in each step. Furthermore, we propose to use contextual gates to dynamically combine the subsequence level and word level contextual information. Experimental results on abstractive summarization and neural machine translation show that word attention significantly improve over strong baselines. In particular, we achieve the state-of-the-art result on WMT'14 English-French translation task with $12M$ training data.

1 Introduction

Recurrent Neural Networks (RNNs) based encoder-decoder framework has been successfully applied to various sequence to sequence text understanding tasks, e.g. neural machine translation (Sutskever, Vinyals, and Le 2014; Bahdanau, Cho, and Bengio 2014), abstractive summarization (Nallapati et al. 2016a) and dialogue system (Asri, He, and Suleman 2016). The key component to the success of RNNs based sequence to sequence learning is the attention mechanism (Bahdanau, Cho, and Bengio 2014; Luong, Pham, and Manning 2015). The attention mechanism is able to allow the model automatically rely on different parts of the source sentence when generating a target word, and hence avoids using a fixed-size vector to represent the entire source sentence, and makes it possible to generate distinct and more relevant source sentence representation for each decoding step.

Typical attention mechanism models the dependency between source and target sentences through the interaction between encoder and decoder RNN hidden states, and summarizes such dependency into context vectors. Concretely, as the first step, the encoder RNN processes the source sentence into a memory consisting of all hidden state vectors. Such a memory is then queried by the target side RNN hidden state at each decoding step, generating a distribution over the memory itself. Based on such a distribution, the hidden states in the source side memory are linearly weighted into a *context vector* to represent source sentence, which is further used in generating the target side word in every step. It is therefore observed that the context vector acts as the key component of attention mechanism, since it dynamically represents all the source side information.

In current attention mechanism, the context vectors are built upon the RNN hidden state vectors, which act as representations of prefix substrings of the source sentence, given the sequential nature of RNN computation. Apart from such subsequence level context vectors, in this paper, we propose to leverage an additional context vector computed by a new attention mechanism named *word attention*, which allows the decoder to directly and selectively touch more raw source sentence information on word level without any complicated sequential operations. From the perspective of source side representation, our word attention enhanced model extracts the semantic information of the source sentence at different abstractive levels, combining previous subsequence level representation with low-level word representation. In this way, we are able to rebuild more adaptive and comprehensive context vectors to encode source sentence, which assist the attention model in every decoding step.

Concretely, our proposed *word attention* is a complementary attention that concentrates on specific source word information during target sentence generation. At each decoding step, the word attention will selectively pay attention to different source side words, and generate an attentive word context that is built directly upon word embeddings. The word context vector is then used to emit the target side word, together with previous context vector that arises from RNN hidden states that summarize source sentence at subse-

[*] This work was conducted at Microsoft Research Asia.

quence level.

Furthermore, in order to better balance the two context vectors, we add additional multiplicative *contextual gates* to control the information flow from conventional hidden state based context and the new word embedding based context at each decoding step. As we discussed before, in source side representation for attention model, word context complements conventional hidden state based context by bringing in more word level and direct information of the source sentence. Accordingly, contextual gates adaptively control how much each of the two contexts contributes to current decoding step. As revealed by our experiments, word context vectors are much better controlled with such gate units, playing its unique contribution in certain scenarios in which direct source side word information is needed.

Our contributions are summarized as follows:

- We propose to leverage the source side word level information to form a complementary attentive word context. Such a word level attention makes the source side contextual information to be more comprehensive.
- In order to better combine the hidden context and word context, we leverage contextual gates to dynamically select the amount of both contexts at each decoding step.
- Experiments on both abstractive summarization and neural machine translation clearly show that our method well improves model performance by a non-trivial margin. In particular, on WMT'14 English-French translation task with $12M$ training dataset, our method achieves the best performance among all the published results.

2 Background

In this section, we introduce the background of our work.

2.1 RNNs based Sequence to Sequence Modeling

Many sequence to sequence text understanding tasks rely on the Recurrent Neural Networks (RNNs) based encoder-decoder framework. In such a framework, the encoder RNN first maps an input source sentence $\mathbf{x} = (x_1, x_2, ..., x_{T_x})$ into hidden states $H = (h_1, h_2, ..., h_{T_x})$, and then the decoder RNN takes these state representations as input and generates the output sentence $\mathbf{y} = (y_1, y_2, ..., y_{T_y})$ word by word.

Specifically, given the source sentence $\mathbf{x}$ and previously generated target sequence $y_{<i}$, the conditional probability of generating word y_i at time step i is decided by:

$$\begin{aligned} p(y_i|y_{<i}, \mathbf{x}) &= g(y_{i-1}, s_i, c_i^\alpha) \\ s_i &= f(s_{i-1}, y_{i-1}, c_i^\alpha), \end{aligned}$$

where g is the softmax function, s_i is the hidden state of decoder RNN at time step i. In practice the choice of the recurrent component f is Long Short Term Memory (LSTM) unit (Hochreiter and Schmidhuber 1997) or Gated Recurrent Unit (GRU) (Cho et al. 2014). c_i^α is the source sentence representation (i.e., context vector), which can be the last hidden state h_{T_x} from encoder, mean of the encoder hidden states $\frac{1}{T_x}\sum_j^{T_x} h_j$, or calculated by an attention mechanism which we will review in the next subsection 2.2.

2.2 Attention Mechanism

The attention mechanism (Bahdanau, Cho, and Bengio 2014; Luong, Pham, and Manning 2015) allows fluent information flow by making each target side word directly and dynamically affected by subparts of the source sentence. It firstly leverages source hidden states H and target hidden state to obtain the attention weights, and then outputs an attentive context vector that is further used to generate the target word. Concretely, in generating the i-th target word, the attention mechanism weights each source hidden state h_j according to:

$$\alpha_{ij} = \frac{\exp(e_{ij}^\alpha)}{\sum_{k=1}^{T_x} \exp(e_{ik}^\alpha)}, \tag{1}$$

where the energy function

$$e_{ij}^\alpha = A_\alpha(s_{i-1}, h_j) \tag{2}$$

is the key alignment model, typically in the form of feedforward neural network. In our work we set it to be the same as (Bahdanau, Cho, and Bengio 2014):

$$A_\alpha(s_{i-1}, h_j) = v_a^T \tanh(W_a s_{i-1} + U_a h_j). \tag{3}$$

Then the attentive context vector c_i^α is calculated by:

$$c_i^\alpha = \sum_{j=1}^{T_x} \alpha_{ij} h_j. \tag{4}$$

Apparently, such context vector is based on hidden states H. We therefore in the rest of the paper, name such context vector c_i^α as *hidden context*.

3 Models

In this section, we will introduce our proposed word attention mechanism. The overall framework of our model is illustrated in Figure 1, including the word attention (the bottom blue parts in the figure), and the contextual gates (the yellow circle of the decoder in the figure).

3.1 Word Attention

From the Equation (2) to Equation (4), we observe that current attention mechanism heavily relies on the RNN hidden states as representation for source sentence. As discussed before, such attention mechanism enables the target word generation to be dependent on source subsequence level information, since each hidden state summarizes the information from the beginning of the sentence. Apart from that, we in this paper argue that the distinct and clean source word information would be beneficial for the target decoding, and those word level context vectors will act as qualified semantic supplementary to the conventional subsequence level representation H. To add such word level information, at each decoding step, we leverage source word embedding x_j together with target hidden state s_{i-1} to compute *word attention weight* β_{ij}, and by weighted averaging word embeddings using β_{ij}, we can obtain an extra context vector c_i^β, which we refer to as *word context.* [1]

[1] For the sake of clarity, we denote original hidden attention related variables with symbol α and our new word attention related variables with symbol β.

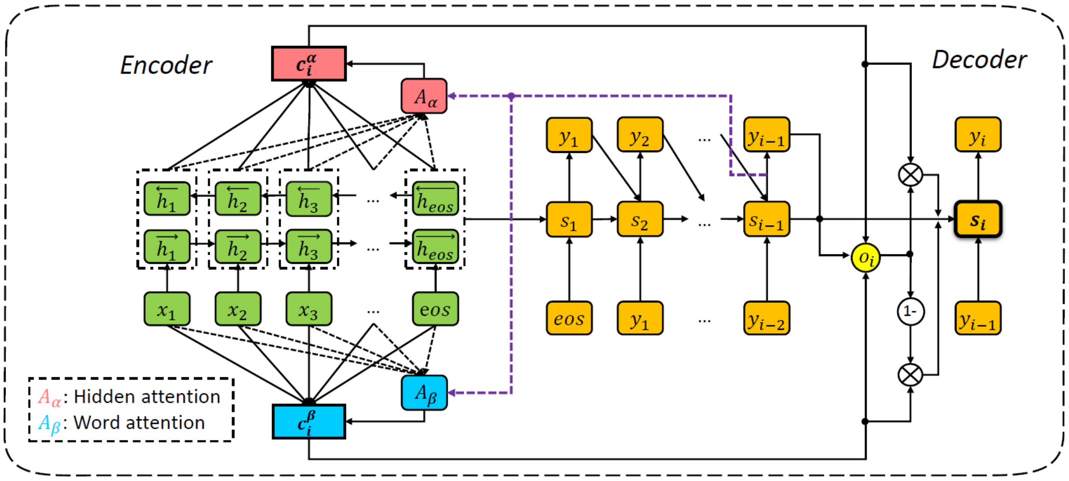


Figure 1: The overall architecture of our word attention model (better viewed in color mode). The top red parts A_α in the encoder denote the original attention mechanism computed with RNN hidden states together with its hidden context c_i^α. The bottom blue parts A_β in the encoder are our new word attention and its correspondingly attentive word context c_i^β. The yellow circle $\mathbf{o_i}$ represents the contextual gates to automatically balance the contribution of hidden context c_i^α and word context c_i^β when generating target hidden state s_i.

Specifically, when decoding a target word y_i at time step i, similar to Equation (1), we first calculate the attention weight β_{ij} as the softmax of energy function e_{ij}^β:

$$\beta_{ij} = \frac{\exp(e_{ij}^\beta)}{\sum_{k=1}^{T_x} \exp(e_{ik}^\beta)}, \qquad (5)$$

where the energy function e_{ij}^β is computed by j-th source word embedding x_j and previous target hidden state s_{i-1}:

$$e_{ij}^\beta = A_\beta(s_{i-1}, x_j) = v_b^T \tanh(W_b s_{i-1} + U_b x_j), \qquad (6)$$

where $v_b \in \mathbb{R}^m$, $W_b \in \mathbb{R}^{m\times n}$ and $U_b \in \mathbb{R}^{m\times m}$ are the weight matrices, with the dimensions of word embedding and decoder hidden units respectively denoted as m and n.

The above attention weight β_{ij} can be regarded as the probability that target word y_i is *directly* aligned to a specific source word x_j, without any extra information of previous words $x_{<j}$. After getting the attention weight β_{ij} for all source words, the resulting word context is the weighted sum of all source word embedding vectors:

$$c_i^\beta = \sum_{j=1}^{T_x} \beta_{ij} x_j. \qquad (7)$$

The word context c_i^β is then provided as an additional input to derive the current target hidden state s_i through

$$s_i = f(s_{i-1}, y_{i-1}, c_i^\alpha, \mathbf{c_i^\beta}). \qquad (8)$$

With the current decoder state s_i, the last generated word y_{i-1}, the two contexts c_i^α and c_i^β, the output probability $p(y_i|y_{<i}, \mathbf{x})$ of a target word y_i is correspondingly set as:

$$p(y_j|y_{<i}, \mathbf{x}) = g(y_{i-1}; s_i, c_i^\alpha, \mathbf{c_i^\beta}). \qquad (9)$$

To make a clearer introduction, by assuming f as GRU, we give the detailed mathematical form of Equation (8) and Equation (9):

$$\begin{aligned}
s_i &= (1 - z_i) \circ s_{i-1} + z_i \circ \tilde{s}_i \\
\tilde{s}_i &= \tanh(W y_{i-1} + U[r_i \circ s_{i-1}] + C^\alpha c_i^\alpha + \mathbf{C}^\beta \mathbf{c_i^\beta}) \\
z_i &= \sigma(W_z y_{i-1} + U_z s_{i-1} + C_z^\alpha c_i^\alpha + \mathbf{C_z^\beta c_i^\beta}) \\
r_i &= \sigma(W_r y_{i-1} + U_r s_{i-1} + C_r^\alpha c_i^\alpha + \mathbf{C_r^\beta c_i^\beta})
\end{aligned} \qquad (10)$$

$$p(y_i|y_{<i}, \mathbf{x}) = g(W_y y_{i-1} + W s_i + W^\alpha c_i^\alpha + \mathbf{W}^\beta \mathbf{c_i^\beta}).$$

In above equations, the bold parts represent what make our word attention model different from the conventional hidden attention model. All W, U, C are the weight matrices for our model. $\sigma(\cdot)$ denotes the sigmoid function and $\circ$ indicates element-wise product.

With our proposed word attention, we now have two different context vectors: c_i^α and c_i^β, respectively arising from source RNN hidden states and source word embeddings. In current mathematical modeling, i.e., Equation (10), they are directly summed together in target word decoding. However, treating them equally may be not optimal given that different target side words may rely on the source side information at different levels: some of them may prefer source side context information at subsequence level such as some compositional semantics contained in phrases, whereas the others might need more specific and clean word context to obtain more lower level information. Therefore, we propose to use gate units to dynamically control the amount of both contexts for each decoding step, which is introduced in the next subsection.

3.2 Contextual Gates to Combine Hidden Context and Word Context

Contextual gates can achieve better balance of both subsequence level and word level source side information via adaptively controlling the weights of hidden context c_i^{α} and word context c_i^{β}. To be specific, at the i-th decoding step, the multiplicative contextual gates examine both the two context vectors, the RNN hidden state s_{i-1} and last generated word y_{i-1}, and then output a gate vector $\mathbf{o_i}$ to decide the amount of information from the two contexts:

$$\mathbf{o_i} = \sigma(W_o y_{i-1} + U_o s_{i-1} + C_o^{\alpha} c_i^{\alpha} + C_o^{\beta} c_i^{\beta}), \qquad (11)$$

where $W_o \in \mathbb{R}^{n \times m}$, $U_o \in \mathbb{R}^{n \times n}$, $C_o^{\alpha} \in \mathbb{R}^{n \times n}$, $C_o^{\beta} \in \mathbb{R}^{n \times m}$ are weight matrices.

Afterwards, adding $\mathbf{o_i}$ into Equation (10), we recompute the hidden state $s_i = (1 - z_i) \circ s_{i-1} + z_i \circ \tilde{s_i}$ in GRU as

$$\begin{aligned}\tilde{s_i} = \tanh(W y_{i-1} + U[r_i \circ s_{i-1}] &+ \mathbf{o_i} \circ C^{\alpha} c_i^{\alpha} \\ &+ (\mathbf{1} - \mathbf{o_i}) \circ C^{\beta} c_i^{\beta}) \qquad (12)\end{aligned}$$

$$z_i = \sigma(W_z y_{i-1} + U_z s_{i-1} + \mathbf{o_i} \circ C_z^{\alpha} c_i^{\alpha} + (\mathbf{1} - \mathbf{o_i}) \circ C_z^{\beta} c_i^{\beta})$$

$$r_i = \sigma(W_r y_{i-1} + U_r s_{i-1} + \mathbf{o_i} \circ C_r^{\alpha} c_i^{\alpha} + (\mathbf{1} - \mathbf{o_i}) \circ C_r^{\beta} c_i^{\beta}).$$

The hidden context c_i^{α} and word context c_i^{β} are now weighted by the gate vector $\mathbf{o_i}$. Acting in this way, the decoding state s_i can dynamically benefit from both hidden context and word context with different ratios, and help to generate words under different situations. Such contextual gates are clearly shown in Figure 1 via the yellow circle and its related links.

4 Experiments

To evaluate our approach, we carried out experiments on two typical sequence to sequence text understanding tasks: abstractive summarization and neural machine translation.

4.1 Experimental Settings

Abstractive Summarization We first valid our approach on abstractive summarization task. We train on the Gigaword corpus (Graff and Cieri 2003) and pre-process it identically to (Rush, Chopra, and Weston 2015; Shen et al. 2016), resulting in $3.8M$ training article-headline pairs, $190k$ for validation and $2,000$ for test. Similar to (Shen et al. 2016), we use a source and target vocabulary consisting of $30k$ words.

Our model is developed based on one of the most widely used sequence to sequence framework RNNsearch (Bahdanau, Cho, and Bengio 2014) with LSTM as recurrent unit. The embedding size of our model is 620, and the LSTM hidden state size in both encoder and decoder is 1024. The initial values of all weight parameters are uniformly sampled between $(-0.05, 0.05)$. We train our word attention enhanced model by Adadelta (Zeiler 2012) with learning rate 1.0 and gradient clipping threshold 1.5 (Pascanu, Mikolov, and Bengio 2013). The mini-batch size is 64 and the learning rate is halved when the dev performance stops increasing.

Neural Machine Translation We conduct on two machine translation tasks, German-English (De-En for short) and English-French (En-Fr for short).

For De-En, we use data from the De-En machine translation track of the IWSLT 2014 evaluation campaign (Cettolo et al. 2014), which is popular used in machine translation community (Bahdanau et al. 2016; Ranzato et al. 2015; Wu et al. 2017). We follow the same pre-processing as described in above works. The training/dev/test data set respectively contains about $153k$/$7k$/$7k$ De-En sentences pairs, with $32,009$ German words and $22,822$ English words as the vocabulary, leaving the other words replaced by 'UNK'. In addition to the word level experiments, we also conduct experiments on sub-word units level where the corpus are pre-processed with byte pair encoding (BPE). BPE (Sennrich, Haddow, and Birch 2016) has been shown to be an effective approach to handle large vocabulary issue in NMT. We extract about $25k$ sub-word tokens as vocabulary.

For En-Fr, we use a widely adopted benchmark dataset (Jean et al. 2014; Zhou et al. 2016; Wang et al. 2017) which is the subset of WMT'14 En-Fr training corpus, consisting of $12M$ sentences pairs. *newstest 2012* and *newstest 2013* are concatenated as the dev set and *newstest 2014* acts as test set. Different with De-En, for En-Fr, we only run our experiments with sub-word units.

As for model, for De-En, we use a single-layer LSTM model with the dimension of both embedding and hidden state to be 256. Similar to summarization task, we also train the model by Adadelta with learning rate 1.0. The dropout rate is 0.15, the gradient is clipped by 2.5, and the batch size is 32. We automatically halve the learning rate according to validation performance and stop when the performance is not improved anymore. In addition, to further demonstrate the effect of word attention mechanism on top of more powerful baselines, we perform empirical studies based on a much stronger deep RNN model with 2 stacked LSTM layers. The dropout rate for such a deep model is 0.2 for all layers except the output layer before softmax, which is separately set as 0.5. All the hyperparameters such as dropout ratio and gradient clipping threshold are chosen via cross-validation on the dev set. For En-Fr, we directly set the RNNsearch baseline as a 4-layer encoder and 4-layer decoder model and run our model on top of it, with embedding size 512, hidden state size 1024, and the dropout ratio 0.1. The other experimental settings are the same as De-En.

All our models are implemented with Theano (Theano Development Team 2016) and trained on TITAN Xp GPU. For summarization task, it takes about 1 day on one GPU; for De-En 2-layer model, it takes about 4 hours on one GPU; for En-Fr 4-4 layer model, the training takes roughly 17 days on 4 GPUs to converge, with batch size on each GPU as 32 and gradients on each GPU summed together via Nvidia NCCL. For decoding, we use beam search (Sutskever, Vinyals, and Le 2014) with width 5 and 10 for translation and summarization respectively.

4.2 Main Results

As to evaluation measure, we use ROUGE (Lin 2004) F1 score for summarization task and tokenized case-sensitive

BLEU (Papineni et al. 2002) [2] score for translation task.

The results on abstractive summarization are reported in Table 1. Neural machine translation tasks are summarized in Table 2, Table 3 and Table 4. In these tables, "RNNsearch" refers to the baseline model proposed in (Bahdanau, Cho, and Bengio 2014), i.e., the sequence to sequence model with only hidden context in attention mechanism; "Word Attention" refers to our proposed word attention model on top of RNNsearch without gate units; "Contextual Gates" refers to our word attention enhanced model with gate units; "Word" and "BPE" in the first row respectively stand for the experiments conducted on word level and sub-word units level.

Abstractive Summarization For summarization task, the compared baseline models are: 1) *ABS* and *ABS+* (Rush, Chopra, and Weston 2015), both using attentive CNN encoder and NNLM decoder. ABS+ is further tuned by word-level n-gram features; 2) *RAS-Elman* (Chopra, Auli, and Rush 2016), which utilizes convolutional encoder and attention based ELman RNN decoder; 3) *Feats2s* (Nallapati et al. 2016b), which is an RNN sequence to sequence encoder-decoder model taking additional hand-crafted features as input; 4) *Luong-NMT* (Luong, Pham, and Manning 2015), a sequence to sequence model with 2-layer encoder and 2-layer decoder; 5) *Shen MLE* and *+MRT* (Shen et al. 2016), typical attention-based RNN models, the difference is that MRT further uses minimum risk training principle to directly optimize the evaluation measure, showing much better performance; 6) *RNNsearch* (Bahdanau, Cho, and Bengio 2014), which denotes the basic single-layer LSTM based sequence to sequence model implemented by ourselves.

From Table 1, we can see that our model has a significant improvement over most of baselines. In particular: 1) compared with RNNsearch, our model achieves 2.2, 1.3, 1.7 points improvement on unigram based ROUGE-1, bigram based ROUGE-2 and longest common subsequence based ROUGE-L F1 score respectively; 2) our word attention enhanced model even achieves the similar performance with MRT, which is specially designed to directly optimize the evaluation metric. We believe incorporating the MRT principle would further improve the performance of our method.

Neural Machine Translation For De-En translation task, from Table 2, compared with the single-layer RNNsearch baseline, our word attention enhanced model with gate units achieves an improvement of 0.74 and 0.87 BLEU points on word level and sub-word units level respectively. Table 3 summarizes the empirical verifications conducted with the 2-layer stacked LSTM model. By incorporating the word attention into such a stronger model, on word level translation, we obtain 0.90 BLEU points improvement over the original baseline 29.01, and we even outperform previously reported best result 29.16 from (Huang et al. 2017) by 0.75 points. Furthermore, on sub-word units level translation, again we achieve new state-of-the-art result for this task with the BLEU score of *31.90*, outperforming any other previous reported numbers by a non-trivial margin.

Model	RG-1	RG-2	RG-L
ABS	29.55	11.32	26.42
ABS+	29.76	11.88	26.96
RAS-Elman	33.78	15.97	31.15
Feats2s	32.67	15.59	30.64
Luong-NMT	33.10	14.45	30.71
Shen MLE	32.67	15.23	30.56
+MRT	36.54	16.59	33.44
RNNsearch	33.67	15.68	31.67
+Word Attention	35.64	16.64	33.03
+Contextual Gates	**35.93**	**16.99**	**33.41**

Table 1: ROUGE F1 scores on abstractive summarization test set. RG-N stands for N-gram based ROUGE F1 score, RG-L stands for longest common subsequence based ROUGE F1 score. Our work is significantly better than RNNsearch ($p < 0.01$).

We also report parameters comparison on 2-layer models in Table 3. As we can see, incorporating word attention only has a marginal increment of parameters compared with the 2-layer stacked LSTM baseline.

For En-Fr translation task, we compare our model with several strong baseline models, including LAU from (Wang et al. 2017), Deep Attention model (Deep-Att), Google's NMT system (GNMT), and 4-4 layer RNNsearch model implemented by ourselves. Again our word attention enhanced model achieves 39.10 BLEU score, which even surpasses previous widely acknowledged stack LSTM baselines which: 1) are trained with much larger dataset, i.e., the 38.95 BLEU is obtained on the full $36M$ WMT'14 En-Fr dataset for GNMT, while we are only using $12M$; 2) have much larger depths, e.g. $18-7$ and $8-8$ encoder-decoder stacked depth respectively for Deep-Att (Zhou et al. 2016) and GNMT (Wu et al. 2016). As far as we know, this (39.10) is the best reported result for WMT'14 En-Fr translation task obtained on the $12M$ training subset.

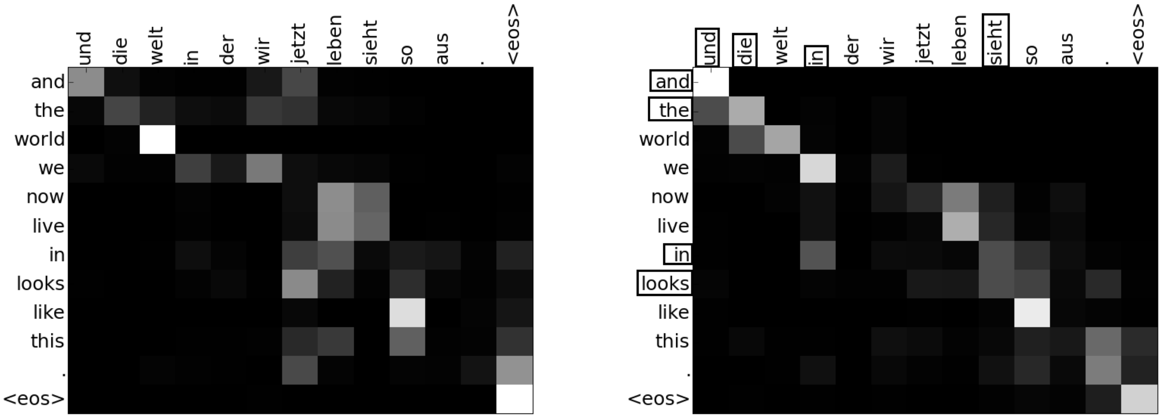
Model	Word	BPE
RNNsearch	26.98	27.83
+Word Attention	27.39	28.49
+Contextual Gates	**27.72**	**28.70**

Table 2: BLEU scores on De-En test set for single-layer models. Our work is significantly better than RNNsearch ($p < 0.01$).

4.3 Qualitative Analysis

To better understand the impact brought by the word attention in sequence to sequence learning, we provide qualitative analysis in this subsection. Concretely, through some example cases in De-En translation task, we visualize the effect of the two main parts of word attention, i.e., the attention weights and gate units.

[2] Calculated by the multi-bleu.pl script at `https://github.com/moses-smt/mosesdecoder/blob/master/scripts/generic/multi-bleu.perl`



(a) Attention weights from RNNsearch.

(b) Gated attention weights from our model.

Figure 2: Visualization of the attention weights for one De-En translation case. The x-axis and y-axis of each plot correspond to the words in the source sentence (German) and the target sentence (English). 2(a) is attention weights from RNNsearch, 2(b) is gated attention weights from our model. The different and important parts are emphasized with rectangle.

Model	Word	Params	BPE	Params
NPMT+LM	29.16	-	-	-
2-2 RNNsearch	29.01	24.3M	31.03	25.0M
+Word Attention	29.68	24.9M	31.71	25.6M
+Contextual Gates	**29.91**	25.6M	**31.90**	26.3M

Table 3: BLEU scores on De-En test set for 2-layer models. The BLEU number for baseline model "NPMT+LM" is reported in the original paper (Huang et al. 2017). Our work is significantly better than 2-2 RNNsearch ($p < 0.01$).

Model	Data	BLEU
LAU (Wang et al. 2017)	$12M$	35.10
Deep-Att (Zhou et al. 2016)	$12M$	35.90
Deep-Att (Zhou et al. 2016)	$36M$	37.70
Deep-Att+PosUNK (Zhou et al. 2016)	$36M$	39.20
GNMT (Wu et al. 2016)	$36M$	38.95
4-4 RNNsearch	$12M$	38.50
+Contextual Gates	$12M$	**39.10**

Table 4: BLEU scores on En-Fr test set. Our work is significantly better than 4-4 RNNsearch ($p < 0.05$).

Visualize Attention Weights Figure 2 shows the attention weights for one De-En translation example. The left subfigure 2(a) plots the alignment matrix from RNNsearch. As to our model, for each decoding step i, we first average the gate units value as $o_i = \bar{\mathbf{o}}_\mathbf{i}$, where $\mathbf{o_i}$ is the gate units vector at this step. After such average operation, we obtain the scalar o_i and use it to derive the visualized gated attention weight as $o_i * \alpha_{ij} + (1 - o_i) * \beta_{ij}$. The right subfigure 2(b) shows these gated attention weights.

From Figure 2(b), we can see that the word pair *"and the"* is aligned to source word pair *"und die"* with strong attention weights, while in RNNsearch, the two word pairs are relatively less correlated. As another observation, the target word *"in"* should be aligned to source word *"in"*, which is well achieved in our model. However, there is no such alignment in RNNsearch. Furthermore, target word *"looks"* is mostly aligned to the correct source side word *"sieht"* by our model. As a comparison, in RNNsearch *"looks"* is aligned to *"jetzt"* and has nearly no alignment with *"sieht"*. These observations clearly demonstrate the effectiveness of our proposal and prove the importance of clean word information in fixing the potential attention errors based on RNN hidden states.

Visualize Contextual Gates The contextual gates are used to dynamically control the amount of hidden context and word context. In order to demonstrate the effectiveness of such gate units, we visualize gate units value $1 - o_i$ for the same De-En translation pair shown in Figure 2, where o_i is the averaged gate units value as introduced before.

The visualization result is displayed in Figure 3. Form this figure, we can observe that the visualization result of gate units actually coincides with attention weights visualization in Figure 2: the target words *"and", "the", "in"* and *"looks"* are in deeper color than other words, which means at these steps, the decoder focuses more on word context than other steps. With more attention on clean word context, (i.e., larger $1 - o_i$), our model has more accurate alignments at these steps as shown in Figure 2(b), which makes it more likely to generate correct translations (see the first translation case in Table 5 in next subsection 4.4).

4.4 Generation Sentence Study

To further understand the advantages brought by our word attention enhanced model, we show two translation sentences in Table 5 from De-En translation task, and one summarized headline example from abstractive summarization task. Their major different parts are emphasized by bold fonts which lead to different sequence generation quality.

Src	und die welt in der wir jetzt leben **sieht so** aus .
Ref	and the world we now live in **looks like** this .
RNNs	and the world that we live in now is what we live .
Ours	and the world that we live in now **looks like** this .
Src	lassen sie uns nun **unseren** blick auf die rollstuhlfahrer richten , etwas , für das ich mich **besonders leidenschaftlich** einsetze .
Ref	now let &apos;s turn **our** heads towards the wheelchair users , something that i &apos;m **particularly passionate** about .
RNNs	so let &apos;s take a look at the UNK , something i &apos;m going to do something like that .
Ours	let &apos;s now take **our** view on the UNK , something that i &apos;m **particularly passionate** about .
Src	the chairman of china's national people's congress (npc), qiao shi, said today that **china** is **willing to expand** economic cooperation with italy on the basis of equality and mutual benefits.
Ref	**china willing to expand** economic ties with italy: npc leader
RNNs	chinese npc chairman meets italy on economic cooperation
Ours	**china willing to expand** economic cooperation with italy

Table 5: Cases-studies to demonstrate the translation and summarization quality improvement brought by our model. We provide two De-En translation examples and one summarization example, together with the source sentence (Src), ground-truth sentence (Ref), and two output sentences respectively provided by RNNsearch (RNNs) and our model (Ours).

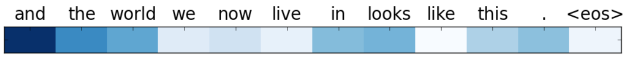


Figure 3: Visualization of the gate units on one De-En translation case. This figure shows the target sentence. The deeper blue color refers to larger value of $1 - o_i$, which means the decoder concentrates more on word context.

From the first translation case, as analysed before, one can see that *"sieht"* and *"so"* were not correctly translated in RNNsearch, while successfully translated into*"looks like"* after grasping the word context by our method. Similarly in the second case, *"unseren", "besonders"* and *"leidenschaftlich"* were missed during translation in RNNsearch, but our model correctly translated these words into *"our", "particularly"* and *"passionate"*. The third summarization case again demonstrates the effects of our proposal. One can see that *"china willing to expand"* from source article are remained in the ground-truth, and our model successfully generates these words in the summarized headline. These examples respectively prove that our word attention enhanced model improves the translation and summarization quality by making full use of word semantics.

5 Related Work

The attention mechanism for sequence to sequence text understanding is first used in neural machine translation task. (Bahdanau, Cho, and Bengio 2014) first introduce the "soft" attention mechanism by allowing a model to automatically align relevant parts of a source sentence to a target word at each decoding step with different weights. After that, plenty of works target at improving the performance of attention model. For example, (Luong, Pham, and Manning 2015) propose a local attention model that focuses on a subset of source hidden states, instead of the entire source hidden states. (Yang et al. 2016) use an LSTM to design an recurrent attention model. Recently there are also several works on self-attention (intra-attention) mechanism that attempt to enhance the coupling between different parts of the same sequence for the sake of better sequence representation (Cheng, Dong, and Lapata 2016; Lin et al. 2017; Paulus, Xiong, and Socher 2017; Vaswani et al. 2017).

All of above mentioned works calculate the attention weights only depend on source and target hidden states, without any word level information. Recently, (Wang et al. 2017; Gehring et al. 2017) try to directly incorporate word level information into attention computation. However, they only focus on attention computation in decoder side by adding target word embedding, e.g. (Wang et al. 2017) involve word embedding in target sentence, but still use RNN hidden states to represent source sentence. Our work is quite different from them, since our complementary word attention gets full use of specific source word embedding from encoder, and outputs an additional word context to enhance the source sentence representation.

6 Conclusion

In this paper, for sequence to sequence learning, we propose to compute attention weights by leveraging clean source word level information to enhance the semantic representation of source sentence. We also introduce contextual gates to dynamically select the contribution of hidden context and word context. Both of the two components contribute to make a better attention model. Our empirical study on two typical sequence to sequence text understanding tasks, abstractive summarization and neural machine translation, clearly shows that word attention with contextual gates significantly improve the performance of RNNs based sequence to sequence learning. For future work, we plan to apply our approach to more sequence to sequence text understanding tasks, like open-domain dialogue system, and incorporate our word attention into other sequence to sequence architectures such as ConvS2S (Gehring et al. 2017).

7 Acknowledgments

This work was partly supported by the NSFC (U1611461, 61573387).

References

Asri, L. E.; He, J.; and Suleman, K. 2016. A sequence-to-sequence model for user simulation in spoken dialogue systems. *arXiv preprint arXiv:1607.00070.*

Bahdanau, D.; Brakel, P.; Xu, K.; Goyal, A.; Lowe, R.; Pineau, J.; Courville, A.; and Bengio, Y. 2016. An actor-critic algorithm for sequence prediction. *arXiv preprint arXiv:1607.07086.*

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

Cettolo, M.; Niehues, J.; Stüker, S.; Bentivogli, L.; and Federico, M. 2014. Report on the 11th iwslt evaluation campaign, iwslt 2014.

Cheng, J.; Dong, L.; and Lapata, M. 2016. Long short-term memory-networks for machine reading. *arXiv preprint arXiv:1601.06733.*

Cho, K.; van Merrienboer, B.; Gulcehre, C.; Bahdanau, D.; Bougares, F.; Schwenk, H.; and Bengio, Y. 2014. Learning phrase representations using rnn encoder–decoder for statistical machine translation. In *EMNLP*.

Chopra, S.; Auli, M.; and Rush, A. M. 2016. Abstractive sentence summarization with attentive recurrent neural networks. In *NAACL*, 93–98.

Gehring, J.; Auli, M.; Grangier, D.; Yarats, D.; and Dauphin, Y. N. 2017. Convolutional sequence to sequence learning. *arXiv preprint arXiv:1705.03122.*

Graff, D., and Cieri, C. 2003. English gigaword, linguistic data consortium.

Hochreiter, S., and Schmidhuber, J. 1997. Long short-term memory. *Neural computation.*

Huang, P.; Wang, C.; Zhou, D.; and Deng, L. 2017. Neural phrase-based machine translation. *CoRR* abs/1706.05565.

Jean, S.; Cho, K.; Memisevic, R.; and Bengio, Y. 2014. On using very large target vocabulary for neural machine translation.

Lin, Z.; Feng, M.; Santos, C. N. d.; Yu, M.; Xiang, B.; Zhou, B.; and Bengio, Y. 2017. A structured self-attentive sentence embedding. *arXiv preprint arXiv:1703.03130.*

Lin, C.-Y. 2004. Rouge: A package for automatic evaluation of summaries. In *ACL-04 workshop*. Barcelona, Spain.

Luong, M.-T.; Pham, H.; and Manning, C. D. 2015. Effective approaches to attention-based neural machine translation. In *EMNLP*.

Nallapati, R.; Zhou, B.; Gulcehre, C.; Xiang, B.; et al. 2016a. Abstractive text summarization using sequence-to-sequence rnns and beyond. *arXiv preprint arXiv:1602.06023.*

Nallapati, R.; Zhou, B.; Gulcehre, C.; Xiang, B.; et al. 2016b. Abstractive text summarization using sequence-to-sequence rnns and beyond. *arXiv preprint arXiv:1602.06023.*

Papineni, K.; Roukos, S.; Ward, T.; and Zhu, W.-J. 2002. Bleu: a method for automatic evaluation of machine translation. In *ACL*. Association for Computational Linguistics.

Pascanu, R.; Mikolov, T.; and Bengio, Y. 2013. On the difficulty of training recurrent neural networks. In *ICML*, 1310–1318.

Paulus, R.; Xiong, C.; and Socher, R. 2017. A deep reinforced model for abstractive summarization. *arXiv preprint arXiv:1705.04304.*

Ranzato, M.; Chopra, S.; Auli, M.; and Zaremba, W. 2015. Sequence level training with recurrent neural networks. *arXiv preprint arXiv:1511.06732.*

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

Sennrich, R.; Haddow, B.; and Birch, A. 2016. Neural machine translation of rare words with subword units. In *ACL*.

Shen, S.; Zhao, Y.; Liu, Z.; Sun, M.; et al. 2016. Neural headline generation with sentence-wise optimization. *arXiv preprint arXiv:1604.01904.*

Sutskever, I.; Vinyals, O.; and Le, Q. V. 2014. Sequence to sequence learning with neural networks. In *NIPS*.

Theano Development Team. 2016. Theano: A Python framework for fast computation of mathematical expressions. *arXiv e-prints* abs/1605.02688.

Vaswani, A.; Shazeer, N.; Parmar, N.; Uszkoreit, J.; Jones, L.; Gomez, A. N.; Kaiser, L.; and Polosukhin, I. 2017. Attention is all you need. *CoRR* abs/1706.03762.

Wang, M.; Lu, Z.; Zhou, J.; and Liu, Q. 2017. Deep neural machine translation with linear associative unit. *arXiv preprint arXiv:1705.00861.*

Wu, Y.; Schuster, M.; Chen, Z.; Le, Q. V.; et al. 2016. Google's neural machine translation system: Bridging the gap between human and machine translation. *arXiv preprint arXiv:1609.08144.*

Wu, L.; Zhao, L.; Qin, T.; Lai, J.; and Liu, T.-Y. 2017. Sequence prediction with unlabeled data by reward function learning. In *IJCAI-17*, 3098–3104.

Yang, Z.; Hu, Z.; Deng, Y.; Dyer, C.; and Smola, A. 2016. Neural machine translation with recurrent attention modeling. *arXiv preprint arXiv:1607.05108.*

Zeiler, M. D. 2012. Adadelta: an adaptive learning rate method. *arXiv preprint arXiv:1212.5701.*

Zhou, J.; Cao, Y.; Wang, X.; Li, P.; and Xu, W. 2016. Deep recurrent models with fast-forward connections for neural machine translation. *arXiv preprint arXiv:1606.04199.*