Neural Machine Translation with Memory Network Based Attention

Étienne Simon*
ENS Paris-Saclay
61 Avenue du Président Wilson,
94230 Cachan
esimon@esimon.eu

Holger Schwenk
Facebook AI Research
75002 Paris, FRANCE
schwenk@fb.com

Abstract

We investigate whether memory networks can be used to improve the attention model of a neural MT system. We will show that using multiple hops on the source sentence yield improvements of up to 1.4 BLEU on the IWSLT De/En task. We then integrate a second memory of all preceding target words. This brings an additional 0.4 gain in the BLEU score. Finally, we completely remove the decoder LSTM and show that a memory network can jointly handle the attention mechanism and the generation of the target sentence.

1 Introduction

The main idea of neural machine translation (NMT) is to first encode the source sentence into a higher-level representation, and then to create the output sequence with a decoder (Kalchbrenner and Blunsom, 2013; Cho et al., 2014; Sutskever et al., 2014). Usually, both the encoder and decoder are RNNs. These basic sequence-to-sequence models have achieved good results (Sutskever et al., 2014), but the performances tends to decrease with the sentence length. The current state-of-the-art in NMT is to use an attention mechanism (Bahdanau et al., 2015). Before generating the next word, the decoder uses a soft alignment to decide on which part of the source sentence to focus on. By these means, longer sentences can be easily handled.

In this paper, we investigate whether memory networks can be used in neural machine translation. A memory network with one hop performs an operation which is quite similar to the attention mechanism proposed in Luong et al. (2015). We will show that using multiple hops on the source sentence yield improvements of up to 1.4 BLEU. We then integrate a second memory of all preceding target words. This brings an additional 0.4 gain in the BLEU score. Finally, we show that a memory network can jointly handle the attention mechanism and the generation of the target sentence.

2 Architecture

Let us use the following notation to fix ideas:

- x_j: source sequence, $j = 1..n$
- h_j^{enc}: hidden states of the encoder
- y_i: target sequence (output of the decoder), $i = 1..m$
- h_i^{dec}: hidden states of the decoder
- c_i: context vector at step i

The context vector is a weighted sum of all the hidden states of the encoder:

$$c_i = \sum_j \alpha_{ij} h_j^{\text{enc}} \quad (1)$$

$$\text{with } \alpha_{ij} = \frac{exp(e_{ij})}{\sum_k exp(e_{ik})} \quad (2)$$

$$\text{and } e_{ij} = f\left(y_{i-1}, h_{i-1}^{\text{dec}}, h_j^{\text{enc}}\right) \quad (3)$$

The energies e_{ij} are proportional to the importance of the source word x_j in the generation of the target word y_i. In the original paper (Bahdanau et al., 2015), the encoder is a bidirectional GRU, the decoder an GRU and f is an MLP.

Different variants of the attention model have been explored in (Luong et al., 2015), in particular using the dot product between h_{i-1}^{dec} and h_j^{enc} to compute e_{ij}. This approach has a nice analogy with end-to-end memory networks (Sukhbaatar et al., 2015). Given key and value memories k and

*This work was performed while É. Simon performed an internship at Facebook AI Research, summer 2016

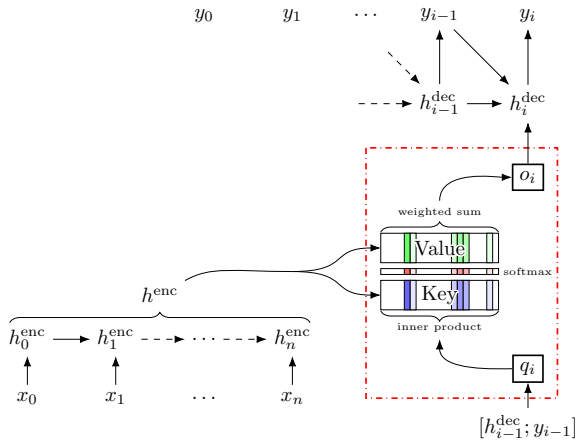


Figure 1: Neural MT system. When a dot product is used to calculate the weights e_{ij} then the attention mechanism performs an operation very similar to a memory network.

v and a query q, a memory network compute its output u as follow:

$$p = \text{softmax}(q^T \cdot k) \quad (4)$$

$$o = \sum_j p_j v_j \quad (5)$$

$$u = o + q \quad (6)$$

The similarity with the attention mechanism is direct: the memories are composed of the hidden states of the encoder $k = v = h^{\text{enc}}$ and the query q is h^{dec}_{i-1}. The sole notable difference being that when computing an attention the query is not usually summed with the output, so $u = o$. This relationship is depicted in Figure 1.

In this paper, we extend the standard attention mechanism in NMT in three ways:

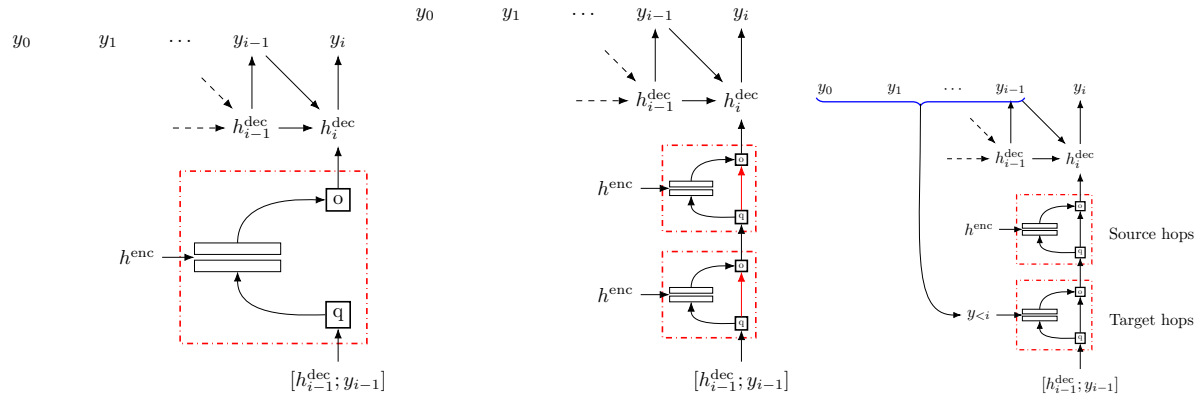
1. Use a memory network with multiple hops on the source to calculate the attention vector. That is, we stack several memory networks on top of each other by setting $q^k = u^{k-1}$ at layer k as illustrated by Figure 2b. By these means we hope to achieve better, incrementally refined alignments;

2. Perform hops on the source and the preceding target words. That is, we use a memory network where the memories are filled with embeddings of the preceding words of the generated sentence. Including a memory on the targets could help to selectively focus on some past words.

3. Replace the decoder LSTM with a memory network which simultaneously performs the attention and sentence generation. In this case the output of the memory network is directly used to generate the target word $y_i = \text{softmax}(Wu)$.

3 Experimental results

We have used the IWSLT 2014 task (Cettolo et al., 2012) to perform all our experiments, translating from German into English. We have chosen this task since the translation of German is challenging for the attention model (the word order may differ a lot from the English one, frequent long distance dependencies, etc). The limited amount of resources makes it also difficult to learn a good attention model. The training corpus consists of about 170k sentence (3.2M English words). 7k sentences (roughly 120k words) were randomly extracted for development and test respectively.



(a) Standard attention (with dot-product) (b) Extension to multiple hops on the source sentence (c) Hops on the target and source sentence.

Figure 2: Different attention mechanisms used in our NMT system.

	S1	S2	S3	S4	S5	S6	S7
T0	*23.0*	23.5	23.9	24.1	24.4	24.1	23.9
T1	22.0	23.9	24.0	24.2	**24.8**	24.1	23.9
T2	21.8	23.8	24.0	24.2	24.3	23.8	23.7
T3	21.5	23.5	23.8	23.8	23.9	23.9	23.6
T4	21.9	23.5	24.1	24.0	24.3	23.8	23.6
T5	21.4	22.5	23.9	23.7	24.2	23.7	23.2

Table 1: BLEU scores on the test set in function of the number of hops on the source and target sentence. Best results are obtained with one hop on the target (T1) and five hops on the source sentence (S5). The standard attention model achieves a BLEU score of 23.0 (entry T0S1).

We used a BLSTM for the encoder and an LSTM for the decoder, both with one layer only. We performed a grid search to find the best setting of the hyper-parameters: the dimension of the word embeddings and the LSTM hidden states are set to 256. We trained our networks using standard SGD with a batch size of 50, a learning rate of 0.1 and slow decay: at the end of each epoch, if the BLEU score did not improved on the validation set, we scale down the learning rate by 0.6. These settings are identical for all our experiments.

We trained our memory networks in a similar way to their application to language modeling (Sukhbaatar et al., 2015). First of all, the parameters of different hops are shared, so for all hops i we have $v^i = v^0$ and $k^i = k^0$. Furthermore, we use temporal encoding in the decoder memory network, namely, each word is embedded alongside its relative position to the current decoding position. Since word order is already encoded by the BLSTM, we do not use temporal encoding in the encoder memory network. When performing a single hop, we do not sum the query with the memory output as this has shown to worsen performance, so $u = o$. On the other hand, as soon as the decoder performs 2 hops, it becomes necessary to use this additional connection with a ReLU activation function, so $u = o + \text{ReLU}(q)$. When attending the source sentence, the memories are filled with the same vectors: $k_j = v_j = h_j^{\text{enc}}$. On the other hand, we observed that using the same memory vector to predict the alignment matrix and the context vector performed poorly on the target sentence. Therefore, target words y_i are embedded twice: once for the key memory k_i and once for the value memory v_i.

We train each model 5 times with different random initializations and report the score on the test set of the one with smallest validation cost. First of all, we observed that models which attend the source sentence before attending the target sentence performed poorly. Furthermore, alternating between source and target hops did not bring any improvement. Therefore, we report results on model performing target hops followed by source hops (see Table 1). It can be clearly seen that performing multiple hops on the source sentence improves the BLEU score by as much as 1.4 points with 5 hops. Furthermore, introducing a single hop on the target sentence gives best results with a BLEU score of 24.8.

Figure 3 shows the alignment matrices of a model with five hops on the source sentence, that is the output of the softmax at each memory hop. A similar pattern is observed for most sentences: the model begins to predict a fuzzy alignment which is then refined by subsequent hops. This is further confirmed by truncating the memory net-

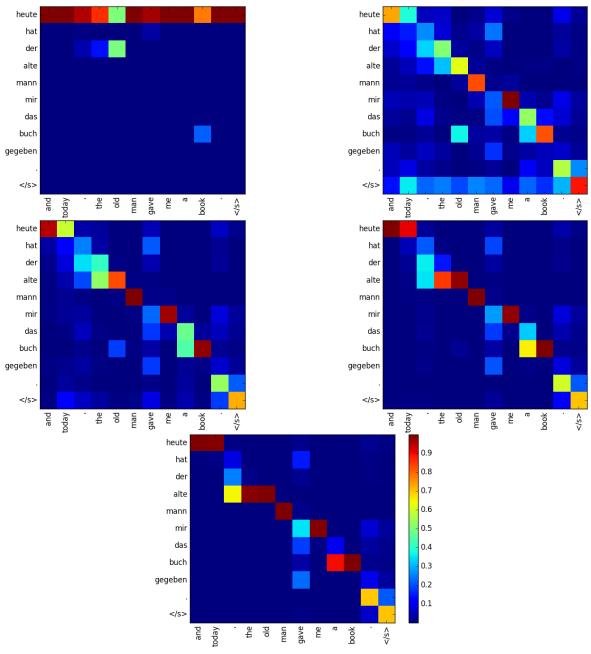


Figure 3: Example of the consecutive alignment matrices when using five hops on the source sentence. At the first hop (top left), the attention only considers the first source word. Hop after hop (left to right, top to bottom), the final alignment is developed. One can clearly see that the alignment of the English words *"gave me"* needs all the hops to find the relation with the German words *"hat ... mir ... gegeben"*.

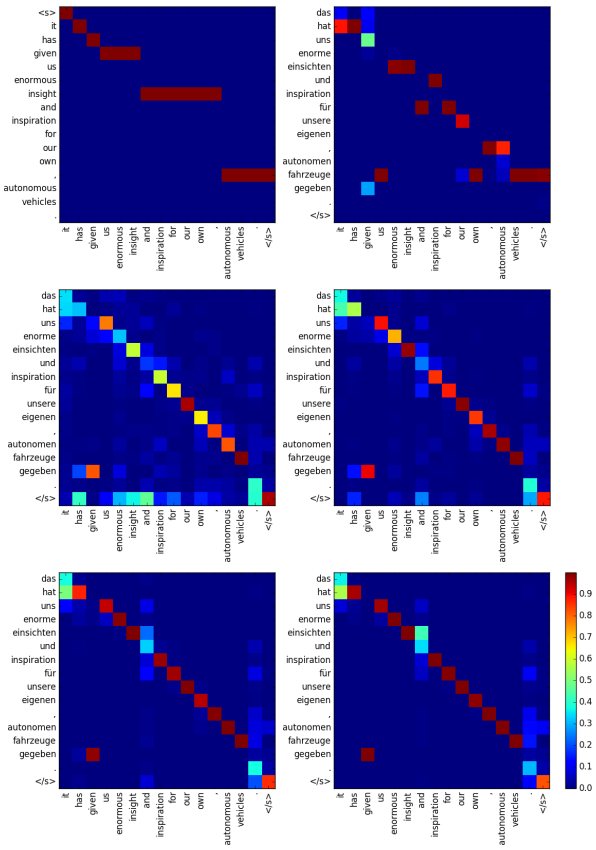


Figure 4: Example of the consecutive alignment matrices when using first one hop on the target sentence (top left) and then five hops on the source sentence (left to right, top to bottom). The target alignments a very sharp and concentrate on one word only at each step. The memory net first looks at the preceding target word and then anchors on the same target word in the past while generating several new words. In comparison to Figure 3, the source alignments are quite precise from the first hops and need less refinement.

work: when we train a model with 5 source hops and then test it with only 3 source hops, we observe that the theme of the translated sentences is somewhat preserved while the exact meaning is lost. The consecutive alignments matrices of our best performing model are shown in Figure 4.

In the past, memory networks had been successfully applied to language modeling (LM) (Sukhbaatar et al., 2015). The decoder LSTM of an NMT system is basically an LM which is conditioned on the source sentence. In our best configuration (1 target and 5 source hops), all the preceding target words are already in memory. In other words, our memory network has already all the necessary information to generate the next word and one may wonder whether the LSTM in the decoder is still necessary. To verify this, we completely removed the decoder LSTM. The output of the memory network is directly fed to a softmax output layer to generate the next word. In this case, more hops on the target sentence are needed (as it was observed in language modeling with memory networks). We were able to achieve a BLEU score of 22.9 using 3 hops on the target and 7 hops on the source. This is not as good as our best configuration with an LSTM (BLEU of 24.8), but it matches the baseline NMT system with a standard attention mechanism (BLEU of 23.0). To the best of our knowledge, this is the first time that memory networks are used to generate sequences, conditioned on an input.

4 Related work

Luong et al. (2015) proposed different variants of the original attention model. As mentionned before, their use of the dot-product makes their approach very close to a memory network with a single hop. Several works address the problem of coverage, i.e. how to guarantee that all the source words are considered for translation and that no word is translated multiple times (Tu et al., 2016; Mi et al., 2016; Sankaran et al., 2016; Yang et al., 2016). In most of these approaches some form of memory is introduced (of the preceding soft alignments), but none uses memory networks with multiple hops. We don't explictely handle coverage. Other approaches use a phrase memory (Tang et al., 2016), or *"interactive attention"* (Meng et al., 2016).

5 Conclusion

It has been previously pointed out that the attention mechanism of NMT systems is related to memory networks. An important aspect of memory networks is the ability to perform multiple hops in the memory, and it has been shown in the past that this is the key technique to achieve good performance in tasks with nested relationships.

As far as we know, this work is the first time that an attention mechanism based memory networks with multiple hops has been shown to significantly improve the BLEU score of an NMT system. We then extended our approach by performing hops on the source and target sentence. Finally, we showed that the memory network can be used to jointly align and generate the target sentence, without the need of an LSTM.

References

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2015. Neural machine translation by jointly learning to align and translate. In *ICLR*.

Mauro Cettolo, Christian Girardi, and Marcello Federico. 2012. Wit^3: Web inventory of transcribed and translated talks. In *EAMT*, pages 261–268.

Kyunghyun Cho, Bart van Merrienboer, Caglar Gulcehre, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. 2014. Learning phrase representations using rnn encoder-decoder for statistical machine translation. In *EMNLP*.

Nal Kalchbrenner and Phil Blunsom. 2013. Recurrent continuous translation models. In *EMNLP*, pages 1700–1709.

Minh-Thang Luong, Hieu Pham, and Christophe D. manning. 2015. Effective approaches to attention-based neural machine translation. In *EMNLP*, pages 1412–1421.

Fandong Meng, Zhengdong Lu, Hang Li, and Qun Liu. 2016. Interactive attention for neural machine translation. In *https://arxiv.org/abs/1610.05011*.

Haitao Mi, Baskaran Sankaran, Zhiguo Wang, and Abe Ittycheriah. 2016. Coverage embedding models for neural machine translation. In *https://arxiv.org/abs/1605.03148*.

Baskaran Sankaran, Haitao Mi, Yaser Al-Onaizan, and Abe Ittycheriah. 2016. Temporal attention model for neural machine translation. In *https://arxiv.org/abs/1608.02927*.

Sainbayar Sukhbaatar, Arthur Szlam, Jason Weston, and Rob Fergus. 2015. End-to-end memory networks. In *NIPS*, pages 2440–2448.

Ilya Sutskever, Oriol Vinyals, and Quoc V. Le. 2014. Sequence to sequence learning with neural networks. In *NIPS*, pages 3104–3112.

Yaohua Tang, Fandong Meng, Zhengdong Lu, Hang Li, and Philip L.H. Yu. 2016. Neural machine translation with external phrase memory. In *https://arxiv.org/abs/1606.01792*.

Zhaopeng Tu, Zhengdong Lu, Yang Liu, Xiaohua Liu, and Hang Li. 2016. Modeling coverage for neural machine translation. In *ACL*, pages 76–85.

Zichao Yang, Zhiting Hu, Yuntian Deng, Chris Dyer, and Alex Smola. 2016. Neural machine translation with recurrent attention modeling. In *https://arxiv.org/abs/1607.05108*.