Neural Machine Translation with Recurrent Attention Modeling

Zichao Yang, Zhiting Hu, Yuntian Deng, Chris Dyer, Alex Smola
Carnegie Mellon University
{zichaoy,zhitingh,yuntiand,cdyer}@cs.cmu.edu
alex@smola.org

Abstract

Knowing which words have been attended to in previous time steps while generating a translation is a rich source of information for predicting what words will be attended to in the future. We improve upon the attention model of Bahdanau et al. (2014) by explicitly modeling the relationship between previous and subsequent attention levels for each word using one recurrent network per input word. This architecture easily captures informative features, such as fertility and regularities in relative distortion. In experiments, we show our parameterization of attention improves translation quality.

1 Introduction

In contrast to earlier approaches to neural machine translation (NMT) that used a fixed vector representation of the input (Sutskever et al., 2014; Kalchbrenner and Blunsom, 2013), attention mechanisms provide an evolving view of the input sentence as the output is generated (Bahdanau et al., 2014). Although attention is an intuitively appealing concept and has been proven in practice, existing models of attention use content-based addressing and have made only limited use of the historical attention masks. However, lessons from better word alignment priors in latent variable translation model suggests value for modeling attention dependent of content.

A challenge in modeling dependencies between previous and subsequent attention decisions is that source sentences are of different lengths, so we need models that can deal with variable numbers of predictions across variable lengths. While other work has sought to address this problem (Cohn et al., 2016; Tu et al., 2016; Feng et al., 2016), these models either rely on explicitly engineered features (Cohn et al., 2016), resort to indirect modeling of the previous attention decisions as by looking at the content-based RNN states that generated them (Tu et al., 2016), or only model coverage rather than coverage together with ordering patterns (Feng et al., 2016). In contrast, we propose to model the sequences of attention levels for each word with an RNN, looking at a fixed window of previous alignment decisions. This enables us both to learn long range information about coverage constraints, and to deal with the fact that input sentences can be of varying sizes.

In this paper, we propose to explicitly model the dependence between attentions among target words. When generating a target word, we use a RNN to summarize the attention history of each source word. The resultant summary vector is concatenated with the context vectors to provide a representation which is able to capture the attention history. The attention of the current target word is determined based on the concatenated representation. Alternatively, in the viewpoint of the memory networks framework (Sukhbaatar et al., 2015), our model can be seen as augmenting the static encoding memory with dynamic memory which depends on preceding source word attentions. Our method improves over plain attentive neural models, which is demonstrated on two MT data sets.

2 Model

2.1 Neural Machine Translation

NMT directly models the condition probability $p(y|x)$ of target sequence $y = \{y_1, ..., y_T\}$ given source sequence $x = \{x_1, ..., x_S\}$, where x_i, y_j are tokens in source sequence and target sequence respectively. Sutskever et al. (2014) and Bahdanau et al. (2014) are slightly different in choosing the encoder and decoder network. Here we choose the

Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 2, Short Papers, pages 383–387,
Valencia, Spain, April 3-7, 2017.

RNNSearch model from (Bahdanau et al., 2014) as our baseline model. We make several modifications to the RNNSearch model as we find empirically that these modification lead to better results.

2.1.1 Encoder

We use bidirectional LSTMs to encode the source sentences. Given a source sentence $\{x_1, ..., x_S\}$, we embed the words into vectors through an embedding matrix W_S, the vector of i-th word is $W_S x_i$. We get the annotations of word i by summarizing the information of neighboring words using bidirectional LSTMs:

$$\overrightarrow{h}_i = \overrightarrow{\text{LSTM}}(\overrightarrow{h_{i-1}}, W_S x_i) \quad (1)$$

$$\overleftarrow{h}_i = \overleftarrow{\text{LSTM}}(\overleftarrow{h_{i+1}}, W_S x_i) \quad (2)$$

The forward and backward annotation are concatenated to get the annotation of word i as $h_i = [\overrightarrow{h}_i, \overleftarrow{h}_i]$. All the annotations of the source words form a context set, $C = \{h_1, ..., h_S\}$, conditioned on which we generate the target sentence. C can also be seen as memory vectors which encode all the information from the source sequences.

2.1.2 Attention based decoder

The decoder generates one target word per time step, hence, we can decompose the conditional probability as

$$\log p(y|x) = \sum_j p(y_j|y_{<j}, x). \quad (3)$$

For each step in the decoding process, the LSTM updates the hidden states as

$$s_j = \text{LSTM}(s_{j-1}, W_T y_{j-1}, c_{j-1}). \quad (4)$$

The attention mechanism is used to select the most relevant source context vector,

$$e_{ij} = v_a^T \tanh(W_a h_i + U_a s_j), \quad (5)$$

$$\alpha_{ij} = \frac{\exp(e_{ij})}{\sum_i \exp(e_{ij})}, \quad (6)$$

$$c_j = \sum_i \alpha_{ij} h_i. \quad (7)$$

This can also seen as memory addressing and reading process. Content based addressing is used to get weights α_{ij}. The decoder then reads the memory as weighted average of the vectors. c_j is combined with s_j to predict the j-th target word.

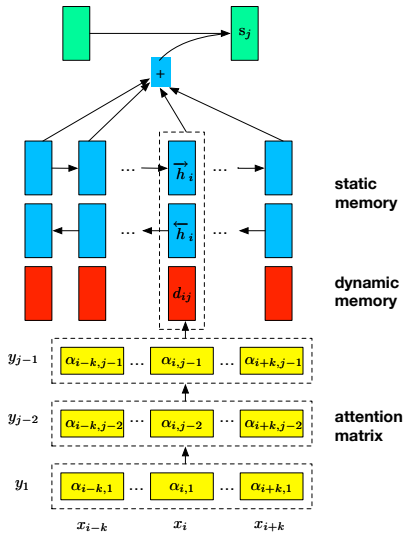

Figure 1: Model diagram

In our implementation we concatenate them and then use one layer MLP to predict the target word:

$$\tilde{s}_j = \tanh(W_1[s_j, c_j] + b_1) \quad (8)$$

$$p_j = \text{softmax}(W_2 \tilde{s}_j) \quad (9)$$

We feed c_j to the next step, this explains the c_{j-1} term in Eq. 4.

The above attention mechanism follows that of Vinyals et al. (2015). Similar approach has been used in (Luong et al., 2015a). This is slightly different from the attention mechanism used in (Bahdanau et al., 2014), we find empirically it works better.

One major limitation is that attention at each time step is not directly dependent of each other. However, in machine translation, the next word to attend to highly depends on previous steps, neighboring words are more likely to be selected in next time step. This above attention mechanism fails to capture these important characteristics. In the following, we attach a dynamic memory vector to the original static memory h_i, to keep track of how many times this word has been attended to and whether the neighboring words are selected at previous time steps, the information, together with h_i, is used to predict the next word to select.

2.2 Dynamic Memory

For each source word i, we attach a dynamic memory vector d_i to keep track of history attention maps. Let $\tilde{\alpha}_{ij} = [\alpha_{i-k,j}, ...\alpha_{i+k,j}]$ be a vector of length $2k+1$ that centers at position i, this vector keeps track of the attention maps status around

word i, the dynamic memory d_{ij} is updated as follows:

$$d_{ij} = \text{LSTM}(d_{i,j-1}, \tilde{\alpha}_{i,j-1}), \forall i \quad (10)$$

The model is shown in Fig. 1. We call the vector d_{ij} dynamic memory because at each decoding step, the memory is updated while h_i is static. d_{ij} is assumed to keep track of the history attention status around word i. We concatenate the d_{ij} with h_i in the addressing and the attention weight vector is calculated as:

$$e_{ij} = v_a^T \tanh(W_a[h_i, d_{ij}] + U_a s_j), \quad (11)$$

$$\alpha_{ij} = \frac{\exp(e_{ij})}{\sum_i \exp(e_{ij})}, \quad (12)$$

$$c_j = \sum_i \alpha_{ij} h_i. \quad (13)$$

Note that we only use dynamic memory d_{ij} in the addressing process, the actual memory c_j that we read does not include d_{ij}. We also tried to get the d_{ij} through a fully connected layer or a convolutional layer. We find empirically LSTM works best.

3 Experiments & Results

3.1 Data sets

We experiment with two data sets: WMT English-German and NIST Chinese-English.

- **English-German** The German-English data set contains Europarl, Common Crawl and News Commentary corpus. We remove the sentence pairs that are not German or English in a similar way as in (Jean et al., 2015). There are about 4.4 million sentence pairs after preprocessing. We use newstest2013 set as validation and newstest2014, newstest2015 as test.

- **Chinese-English** We use FIBS and LDC2004T08 Hong Kong News data set for Chinese-English translation. There are about 1.5 million sentences pairs. We use MT 02, 03 as validation and MT 05 as test.

For both data sets, we tokenize the text with `tokenizer.perl`. Translation quality is evaluated in terms of tokenized BLEU scores (Papineni et al., 2002) with `multi-bleu.perl`.

Model	test1	test2
RNNSearch	19.0	21.3
RNNSearch + UNK replace	21.6	24.3
RNNSearch + window 1	18.9	21.4
RNNSearch + window 11	19.5	22.0
RNNSearch + window 11 + UNK replace	**22.1**	**25.0**
(Jean et al., 2015)		
RNNSearch	16.5	-
RNNSearch + UNK replace	19.0	-
(Luong et al., 2015a)		
Four-layer LSTM + attention	19.0	-
Four-layer LSTM + attention + UNK replace	20.9	-
RNNSearch + character		
(Chung et al., 2016)	21.3	23.4
(Costa-jussà and Fonollosa, 2016)	-	20.2

Table 2: English-German results.

3.2 Experiments configuration

We exclude the sentences that are longer than 50 words in training. We set the vocabulary size to be 50k and 30k for English-German and Chinese-English. The words that do not appear in the vocabulary are replaced with UNK.

For both RNNSearch model and our model, we set the word embedding size and LSTM dimension size to be 1000, the dynamic memory vector d_{ij} size is 500. Following (Sutskever et al., 2014), we initialize all parameters uniformly within range [-0.1, 0.1]. We use plain SGD to train the model and set the batch size to be 128. We rescale the gradient whenever its norm is greater than 3. We use an initial learning rate of 0.7. For English-German, we start to halve the learning rate every epoch after training for 8 epochs. We train the model for a total of 12 epochs. For Chinese-English, we start to halve the learning rate every two epochs after training for 10 epochs. We train the model for a total of 18 epochs.

To investigate the effect of window size $2k + 1$, we report results for $k = 0, 5$, i.e., windows of size $1, 11$.

3.3 Results

The results of English-German and Chinese-English are shown in Table 2 and 3 respectively.

src	She was spotted three days later by a dog walker trapped in the quarry
ref	Drei Tage später wurde sie von einem Spaziergänger im Steinbruch in ihrer misslichen Lage entdeckt
baseline	Sie wurde drei Tage später von einem Hund entdeckt .
our model	Drei Tage später wurde sie von einem Hund im Steinbruch gefangen entdeckt .
src	At the Metropolitan Transportation Commission in the San Francisco Bay Area , officials say Congress could very simply deal with the bankrupt Highway Trust Fund by raising gas taxes .
ref	Bei der Metropolitan Transportation Commission für das Gebiet der San Francisco Bay erklärten Beamte , der Kongress könne das Problem des bankrotten Highway Trust Fund einfach durch Erhöhung der Kraftstoffsteuer lsen .
baseline	Bei der Metropolitan im San Francisco Bay Area sagen offizielle Vertreter des Kongresses ganz einfach den Konkurs Highway durch Steuererhöhungen .
our model	Bei der Metropolitan Transportation Commission in San Francisco Bay Area sagen Beamte , dass der Kongress durch Steuererhöhungen ganz einfach mit dem Konkurs Highway Trust Fund umgehen könnte .

Table 1: English-German translation examples

Model	MT 05
RNNSearch	27.3
RNNSearch + window 1	27.9
RNNSearch + window 11	28.8
RNNSearch + window 11 + UNK replace	**29.3**

Table 3: Chinese-English results.

We compare our results with our own baseline and with results from related works if the experimental setting are the same. From Table 2, we can see that adding dependency improves RNNSearch model by 0.5 and 0.7 on newstest2014 and newstest2015, which we denote as test1 and test2 respectively. Using window size of 1, in which coverage property is considered, does not improve much. Following (Jean et al., 2015; Luong et al., 2015b), we replace the UNK token with the most probable target words and get BLEU score of 22.1 and 25.0 on the two data sets respectively. We compare our results with related works, including (Luong et al., 2015a), which uses four layer LSTM and local attention mechanism, and (Costa-jussà and Fonollosa, 2016; Chung et al., 2016), which uses character based encoding, we can see that our model outperform the best of them by 0.8 and 1.6 BLEU score respectively. Table 1 shows some English-German translation examples. We can see the model with dependent attention can pick the right part to translate better and has better translation quality.

The improvement is more obvious for Chinese-English. Even only considering coverage property improves by 0.6. Using a window size of 11 improves by 1.5. Further using UNK replacement trick improves the BLEU score by 0.5, this improvement is not as significant as English-German data set, this is because English and German share lots of words while Chinese and English don't.

4 Conclusions & Future Work

In this paper, we propose a new attention mechanism that explicitly takes the attention history into consideration when generating the attention map. Our work is motivated by the intuition that in attention based NMT, the next word to attend is highly dependent on the previous steps. We use a recurrent neural network to summarize the preceding attentions which could impact the attention of the current decoding attention. The experiments on two MT data sets show that our method outperforms previous independent attentive models. We also find that using a larger context attention window would result in a better performance.

For future directions of our work, from the static-dynamic memory view, we would like to explore extending the model to a fully dynamic memory model where we directly update the representations for source words using the attention history when we generate each target word.

References

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2014. Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473*.

Junyoung Chung, Kyunghyun Cho, and Yoshua Bengio. 2016. A character-level decoder without explicit segmentation for neural machine translation. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1693–1703, Berlin, Germany, August. Association for Computational Linguistics.

Trevor Cohn, Cong Duy Vu Hoang, Ekaterina Vymolova, Kaisheng Yao, Chris Dyer, and Gholamreza Haffari. 2016. Incorporating structural alignment biases into an attentional neural translation model. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 876–885, San Diego, California, June. Association for Computational Linguistics.

Marta R. Costa-jussà and José A. R. Fonollosa. 2016. Character-based neural machine translation. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 357–361, Berlin, Germany, August. Association for Computational Linguistics.

Shi Feng, Shujie Liu, Nan Yang, Mu Li, Ming Zhou, and Kenny Q. Zhu. 2016. Improving attention modeling with implicit distortion and fertility for machine translation. In *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers*, pages 3082–3092, Osaka, Japan, December. The COLING 2016 Organizing Committee.

Sébastien Jean, Kyunghyun Cho, Roland Memisevic, and Yoshua Bengio. 2015. On using very large target vocabulary for neural machine translation. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 1–10, Beijing, China, July. Association for Computational Linguistics.

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

Minh-Thang Luong, Hieu Pham, and Christopher D Manning. 2015a. Effective approaches to attention-based neural machine translation. *arXiv preprint arXiv:1508.04025*.

Thang Luong, Ilya Sutskever, Quoc Le, Oriol Vinyals, and Wojciech Zaremba. 2015b. Addressing the rare word problem in neural machine translation. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 11–19, Beijing, China, July. Association for Computational Linguistics.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of 40th Annual Meeting of the Association for Computational Linguistics*, pages 311–318, Philadelphia, Pennsylvania, USA, July. Association for Computational Linguistics.

Sainbayar Sukhbaatar, Arthur Szlam, Jason Weston, and Rob Fergus. 2015. End-to-end memory networks. *arXiv preprint arXiv:1503.08895*.

Ilya Sutskever, Oriol Vinyals, and Quoc VV Le. 2014. Sequence to sequence learning with neural networks. In *Advances in neural information processing systems*, pages 3104–3112.

Zhaopeng Tu, Zhengdong Lu, Yang Liu, Xiaohua Liu, and Hang Li. 2016. Modeling coverage for neural machine translation. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 76–85, Berlin, Germany, August. Association for Computational Linguistics.

Oriol Vinyals, Łukasz Kaiser, Terry Koo, Slav Petrov, Ilya Sutskever, and Geoffrey Hinton. 2015. Grammar as a foreign language. In *Advances in Neural Information Processing Systems*, pages 2755–2763.