XMU Neural Machine Translation Systems for CWMT 2017

Zhixing Tan, Boli Wang, Xiansong Ji, Bingyansen Wu, Jinming Hu,
Yidong Chen and Xiaodong Shi*

School of Information Science and Engineering, Xiamen University, Fujian, China

Abstract. This paper describes the Neural Machine Translation systems of Xiamen University for the shared translation tasks of CWMT 2017. Our systems are based on the Encoder-Decoder framework with attention. We participated in all six shared translation tasks. We experimented deep architectures, different segmentation models, synthetic training data and target-bidirectional translation models. Experiments show that all these methods can give substantial improvements.

1 Introduction

Neural Machine Translation (NMT) [1,2,12] has achieved great success in recent years and obtained state-of-the-art results on various language pairs [8,15,16]. This paper describes the NMT systems of Xiamen University (XMU) for the CWMT 2017 evaluation. We participated all six directions of shared tasks: Chinese-English News Translation (CE), English-Chinese News Translation (EC), Mongolian-Chinese Daily Expression Translation (MC), Tibetan-Chinese Government Document Translation (TC), Uyghur-Chinese News Translation (UC), and Japanese-Chinese Patent Domain Translation (JC). We use two different NMTs in these tasks:

- MININMT: A deep NMT system [14,15,16] with a simple architecture. The encoder is a stacked bidirectional Long Short-Term Memory (LSTM) [4] with 8 layers. The decoder is a stacked LSTMs with 8 layers. We apply this system to the CE task.
- DL4MT: Our reimplementation of dl4mt-tutorial[1] with minor changes. We also use a modified version of AmuNMT C++ decoder[2] for parallel decoding. We apply this system to the other five tasks.

We use both Byte Pair Encoding (BPE) [10] and mixed word/character segmentation [15] to achieve open-vocabulary translation. We apply back-translation method [9] to make use of monolingual data. We use target-bidirectional translation models to alleviate the label bias problem [6].

The remainder of this paper is organized as follows: Section 2 describes the architecture of MININMT. Section 3 describes the processing of the data. Section 4 describes

* Corresponding author.

[1] https://github.com/nyu-dl/dl4mt-tutorial

[2] https://github.com/emjotde/amunmt

all experimental features used in six shared translation tasks. Section 5 shows the results of our experiments. Finally, we conclude in section 6.

2 Model Description

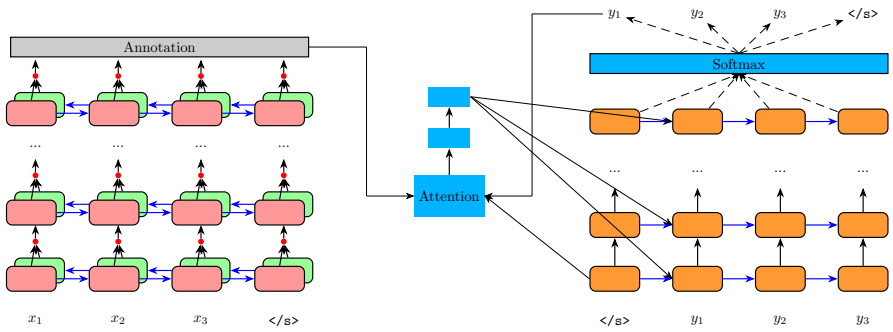

Fig. 1. The architecture of our deep NMT system, which is inspired by Deep-Att [16] and GNMT [15]. Both the encoder and decoder adopt LSTM as its main recurrent unit. We also use residual connections [3] to help training, but here we omit it for clarity. The black lines denote input connections while the blue lines denote recurrent connections.

NMT systems with deep architectures have recently shown promising results on various language pairs [14,15,16]. We also experimented a deep architecture as depicted in Figure 1. We use LSTM as the main recurrent unit and residual connections [3] to help training.

Given a source sentence $\boldsymbol{x} = \{x_1, \ldots, x_S\}$ and a target sentence $\boldsymbol{y} = \{y_1, \ldots, y_T\}$, the encoder maps the source sentence $\boldsymbol{x}$ into a sequence of annotation vectors $\{\mathbf{x}_i\}$. The decoder produces translation y_t given source annotation vectors $\{\mathbf{x}_i\}$ and target history $\boldsymbol{y}_{<t}$.

2.1 Encoder

To better exploit source representation, we adopt a stacked bidirectional encoder. As shown in Figure 1, all layers in the encoder are bidirectional. The calculation is described as follows:

$$\overrightarrow{\mathbf{x}}^i = \text{LSTM}_i^f(\mathbf{x}_t^{i-1}, \overrightarrow{\mathbf{s}}_{t-1}^i) \tag{1}$$

$$\overleftarrow{\mathbf{x}}^i = \text{LSTM}_i^b(\mathbf{x}_t^{i-1}, \overleftarrow{\mathbf{s}}_{t+1}^i) \tag{2}$$

$$\mathbf{x}^i = [\overrightarrow{\mathbf{x}}^{iT}; \overleftarrow{\mathbf{x}}^{iT}]^T \tag{3}$$

To reduce parameters, we reduce the dimension of hidden units from h to $h/2$ so that $\mathbf{x}^i \in \mathbb{R}^h$. The annotation vectors are taken from the output $\mathbf{x}^{L_{\text{enc}}}$ of top LSTM layer. In our experiments, L_{enc} is set to 8.

2.2 Decoder

The decoder network is similar to GNMT [15]. At each timestep t, let $\mathbf{y}_{t-1}^0 \in \mathbb{R}^e$ denotes the word embedding of y_{t-1} and $\mathbf{y}_{t-1}^1 \in \mathbb{R}^h$ denotes the output of bottom LSTM from previous timestep. The attention network calculates the context vector $\mathbf{a}_t$ as weighted sum of source annotation vectors:

$$\mathbf{a}_t = \sum_{i=1}^{S} \alpha_{t,i} \cdot \mathbf{x}_i \tag{4}$$

Different from GNMT [15], we use the concatenation of $\mathbf{y}_{t-1}^0$ and $\mathbf{y}_{t-1}^1$ as the query vector for attention network, as described follows:

$$\mathbf{h}_t = [{\mathbf{y}_{t-1}^0}^T; {\mathbf{y}_{t-1}^1}^T]^T \tag{5}$$

$$e_{t,i} = \mathbf{v}_a^T \tanh(\mathbf{W}_a \mathbf{h}_t + \mathbf{U}_a \mathbf{x}_i) \tag{6}$$

$$\alpha_{t,i} = \frac{\exp(e_{t,i})}{\sum_{j=1}^{S} \exp(e_{t,j})} \tag{7}$$

This approach is also used in [14]. The context vector $\mathbf{a}_t$ then feeds to all decoder LSTMs.

The probability of next word y_t is simply modeled using a softmax layer on the output of top LSTM:

$$p(y_t|\boldsymbol{x}, \boldsymbol{y}_{<t}) = \text{softmax}(y_t, \mathbf{y}_t^{L_{\text{dec}}}) \tag{8}$$

We set L_{dec} to 8 in all our experiments.

3 Data Processing

We use all training corpora provided by CWMT, and UN corpus and News Commentary corpus provided by WMT as well in CE and EC tasks.

In CE and EC tasks, the Chinese sentences are segmented using Stanford Word Segmenter[3]. The English sentences are tokenized and truecased using moses scripts[4].

In MC task, the Chinese sentences are segmented using our Chinese word segmenter `segtag`[5]. The Mongolian sentences are tokenized using our own multilingual tokenizer[6]. We first convert all Latin letters in the Mongolian sentences to their uppercase and then latinize Mongolian letters according to our own transliteration scheme [7]. Considering the homograph issue of Unicoded Traditional Mongolian, we adopt a normalization method to relieve the data sparseness problem [13].

In TC, UC and JC tasks, the Chinese sentences are segmented using `segtag`. In TC task, the Tibetan sentences are segmented using our Tibetan word segmenter

[3] `https://nlp.stanford.edu/software/segmenter.shtml`
[4] `http://statmt.org/moses/`
[5] `http://mandel.cloudtranslation.cc/segtag.html`
[6] `http://mandel.cloudtranslation.cc/mysoft/tokenize.exe`
[7] `http://mandel.cloudtranslation.cc/moncode.html`

tsc[8]. In UC task, we first tokenize the Uyghur sentences by simply inserting extra spaces between Uyghur letters and any other characters. Then, the Latin letters in Uyghur sentences are capitalized and the Uyghur letters are latinized using an open-sourced Uyghur script converter[9]. In JC task, the Japanese sentences are segmented using mecab[10].

For all tasks, we filter out bad sentence pairs according to ratio of length and alignment score obtained by fast-align toolkit[11] and remove duplications in the training data.

4 Experimental Features

4.1 Subword Segmentation

To enable open-vocabulary, we apply subword-based translation approaches. In our preliminary experiments, we found that BPE and mixed word/character segmentation works better than UNK replacement techniques.

In CE and EC tasks, we apply BPE[12] [10] with 50K operations to English sentences. In MC and UC tasks, we use BPE with 30K operations for Mongolian and Uyghur sides.

We apply mixed word/character model [15] to Chinese sentences for all tasks, except CE. We keep the most frequent Chinese words and split other words into characters. We keep 50K Chinese words in EC task and 30K Chinese words in other tasks. Unlike [15], we do not add any extra prefixes or suffixes to the segmented characters. In the post-processing step, we simply remove all the spaces. Similarly, in TC and JC tasks, we also use mixed word/character model with a shortlist of 30K words for both Tibetan and Japanese sides.

4.2 Synthetic Training Data

We use back-translation [9] method to utilize target language monolingual data. We use srilm [13] to train a 5-gram KN language model on the monolingual data and select monolingual sentences according to their perplexity. We sample 2.5M English sentences from the NewsCrawl2016[14] corpus for CE task[15] and 2.5M Chinese sentences from the XinhuaNet2011[16] corpus for EC, MC, TC and UC tasks. For JC patent domain task, we select 2.5M Chinese sentences from Lingosail-cn_for_lm-CWMT2017.

[8] http://mandel.cloudtranslation.cc/mysoft/tsc.rar
[9] https://github.com/neouyghur/Multiple-Uyghur-Script-Converter
[10] https://taku910.github.io/mecab/
[11] https://github.com/clab/fast_align
[12] https://github.com/rsennrich/subword-nmt
[13] http://www.speech.sri.com/projects/srilm/
[14] http://data.statmt.org/wmt17/translation-task/news.2016.en.shuffled.gz
[15] Here, we first filter the English sentences according the vocabulary of the development set.
[16] We split XinhuaNet2011 into sentences using our lbl tool (http://mandel.cloudtranslation.cc/mysoft/lbl.exe).

We train backward translation models on the parallel data and translate the selected monolingual sentences back to the source language.

In our preliminary experiments, we found that training or tuning on the synthetic training data alone could not improve the performance of NMT models. Therefore, in all six tasks, we mix up the synthetic data with a comparable amount of bilingual pairs randomly sampled from parallel data and train or fine-tune NMT models on the mixture data.

4.3 Target-bidirectional Translation

For Chinese-English translation, we use a target-bidirectional model [7,8] to re-score the hypotheses.

To train a target-bidirectional model, we reverse the target side of bilingual pairs from left-to-right (L2R) to right-to-left (R2L). We first output top 50 candidates from L2R models. Then we re-score candidates by interpolating L2R score and R2L score with uniform weights.

4.4 Training

For all our models, we adopt Adam [5] (β_1 = 0.9, β_2 = 0.999 and $\epsilon = 1\times 10^{-8}$) as the optimizer. The initial learning rate is set to 5×10^{-4}. We gradually halve the learning rate during the training process. As a common way to train RNNs, we clip the norm of gradient to a predefined value 5.0. The batch size is 128. We use dropout [11] to avoid over-fitting with a keep probability of 0.8.

5 Results

5.1 Results on Chinese↔English News Tasks

System	**CWMT2017 Test** (BLEU4-SBP)	**WMT17 Test** (BLEU4)
Single	–	23.4
Single + Synthetic	–	23.7
Single + R2L	23.7	–
Ensemble + Synthetic	25.2	25.3
Ensemble + R2L	**25.6**	–
Ensemble + Synthetic + R2L	–	**26.0**

Table 1. Chinese-English translation results on CWMT 2017 and WMT 17 test sets.

Table 1 shows the results of Chinese-English translation. The baseline is a single MININMT model trained on all parallel data. Tuning on synthetic data and re-ranking with R2L model are effective to improve the BLEU score. Moreover, we trained 4

System	CWMT2017 Test (BLEU5-SBP)	WMT17 Test (BLEU4)
Single	–	30.4
Single + Synthetic	21.5	34.3
Ensemble + Synthetic	22.0	35.8

Table 2. English-Chinese translation results on CWMT 2017 and WMT 17 test sets.

models with different random initialization and different data shuffling and apply the ensembling method proposed by [12] in decoding. The results show that the ensembling approach obtains further improvements on both two test sets.

We use our reimplementation of DL4MT to train English-Chinese models on CWMT and UN parallel corpus. The results are shown in Table 2. We obtained significant improvements by tuning on synthetic data and ensembling 4 models.

5.2 Results on Low-Resource Translation Tasks

	CWMT2017 Test (BLEU5-SBP)		
System	Mongolian-Chinese	Tibetan-Chinese	Uyghur-Chinese
Single	14.1	93.3*	32.4
Ensemble	–	93.7*	–
Single + Latinization	62.9* / 60.0	–	48.5*
Ensemble + Latinization	66.3* / 64.9	–	52.5*

Table 3. Mongolian, Tibetan, and Uyghur to Chinese translation results on CWMT 2017 test sets. Models trained on synthetic data are denoted by symbol *. We ensemble 3 models in MC and UC tasks and 4 models in TC task. Due to time constraints, the MC systems trained on synthetic data were not submitted to CWMT2017.

Table 3 shows the results of three low-resource translation tasks. We found that latinization methods, including normalization of Unicoded Traditional Mongolian, are effective to solve the data sparseness problems. We obtained further improvements by training on synthetic data and ensembling.

5.3 Results on Japanese-Chinese Patent Domain Translation

For patent domain task, the translation results are sensitive to word segmentation. We compared our `segtag` tool with an open-sourced Chinese word segmenter `jieba`[17]. The results are shown in Table 4. Our `segtag` outperforms `jieba` by +3.7 BLEU score. When tuning on synthetic data and ensembling of 5 models, we further gained +0.6 BLEU score.

[17] `https://github.com/fxsjy/jieba`

System	Segmenter	CWMT2017 Test (BLEU5-SBP)
Single	`mecab` + `jieba`	36.7
Single	`mecab` + `segtag`	40.4
Ensemble + Synthetic	`mecab` + `segtag`	41.0

Table 4. Japanese-Chinese translation results on CWMT 2017 set.

6 Conclusion

We describe XMU's neural machine translation systems for the CWMT 2017 shared translation tasks. Our models perform quite well on all tasks. Experiments also show the effectiveness of all features we used.

Acknowledgments

This work was supported by the Natural Science Foundation of China (Grant No. 61573294, 61303082, and 61672440), the Ph.D. Programs Foundation of Ministry of Education of China (Grant No. 20130121110040), the Foundation of the State Language Commission of China (Grant No. WT135-10) and the Natural Science Foundation of Fujian Province (Grant No. 2016J05161).

References

1. Bahdanau, D., Cho, K., Bengio, Y.: Neural machine translation by jointly learning to align and translate. In: Proceedings of ICLR (2015)
2. Cho, K., van Merrienboer, B., Gulcehre, C., Bahdanau, D., Bougares, F., Schwenk, H., Bengio, Y.: Learning phrase representations using rnn encoder–decoder for statistical machine translation. In: Proceedings of EMNLP. pp. 1724–1734 (2014)
3. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: Proceedings of CVPR. pp. 770–778 (2016)
4. Hochreiter, S., Schmidhuber, J.: Long short-term memory. Neural Computation pp. 1735–1780 (1997)
5. Kingma, D., Ba, J.: Adam: A method for stochastic optimization. In: Proceedings of ICLR (2015)
6. Lafferty, J., McCallum, A., Pereira, F., et al.: Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In: Proceedings of ICML. vol. 1, pp. 282–289 (2001)
7. Liu, L., Utiyama, M., Finch, A., Sumita, E.: Agreement on target-bidirectional neural machine translation. In: Proceedings of NAACL-HLT. pp. 411–416 (2016)
8. Sennrich, R., Haddow, B., Birch, A.: Edinburgh neural machine translation systems for wmt 16. arXiv preprint arXiv:1606.02891 (2016)
9. Sennrich, R., Haddow, B., Birch, A.: Improving neural machine translation models with monolingual data. In: Proceddings of ACL. pp. 86–96 (2016)
10. Sennrich, R., Haddow, B., Birch, A.: Neural machine translation of rare words with subword units. In: Proceedings of ACL. pp. 1715–1725 (2016)

11. Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., Salakhutdinov, R.: Dropout: A simple way to prevent neural networks from overfitting. The Journal of Machine Learning Research 15(1), 1929–1958 (2014)
12. Sutskever, I., Vinyals, O., Le, Q.V.: Sequence to sequence learning with neural networks. In: Advances in neural information processing systems. pp. 3104–3112 (2014)
13. Wang, B., Shi, X., Chen, Y.: Coping with problems of unicoded traditional mongolian. In: Proceddings of China National Conference on Chinese Computational Linguistics. pp. 125–131 (2016)
14. Wang, M., Lu, Z., Zhou, J., Liu, Q.: Deep Neural Machine Translation with Linear Associative Unit. arXiv preprint arXiv:1705.00861 (may 2017)
15. Wu, Y., Schuster, M., Chen, Z., Le, Q.V., Norouzi, M., Macherey, W., Krikun, M., Cao, Y., Gao, Q., Macherey, K., et al.: Google's neural machine translation system: Bridging the gap between human and machine translation. arXiv preprint arXiv:1609.08144 (2016)
16. Zhou, J., Cao, Y., Wang, X., Li, P., Xu, W.: Deep recurrent models with fast-forward connections for neural machine translation. Transactions of the Association for Computational Linguistics 4, 371–383 (2016)