Learning to Parse and Translate Improves Neural Machine Translation

Akiko Eriguchi[†], **Yoshimasa Tsuruoka**[†], and **Kyunghyun Cho**[‡]
[†]The University of Tokyo, 7-3-1 Hongo, Bunkyo-ku, Tokyo, Japan
{eriguchi, tsuruoka}@logos.t.u-tokyo.ac.jp
[‡]New York University, New York, NY 10012, USA
kyunghyun.cho@nyu.edu

Abstract

There has been relatively little attention to incorporating linguistic prior to neural machine translation. Much of the previous work was further constrained to considering linguistic prior on the source side. In this paper, we propose a hybrid model, called NMT+RG, that learns to parse and translate by combining the recurrent neural network grammar into the attention-based neural machine translation. Our approach encourages the neural machine translation model to incorporate linguistic prior during training, and lets it translate on its own afterward. Extensive experiments with four language pairs show the effectiveness of the proposed NMT+RG.

1 Introduction

Neural Machine Translation (NMT) has enjoyed impressive success without relying on much, if any, prior linguistic knowledge. Some of the most recent studies have for instance demonstrated that NMT systems work comparably to other systems even when the source and target sentences are given simply as flat sequences of characters (Lee et al., 2016; Chung et al., 2016) or statistically, not linguistically, motivated subword units (Sennrich et al., 2016; Wu et al., 2016). Shi et al. (2016) recently made an observation that the encoder of NMT captures syntactic properties of a source sentence automatically, indirectly suggesting that explicit linguistic prior may not be necessary.

On the other hand, there have only been a couple of recent studies showing the potential benefit of explicitly encoding the linguistic prior into NMT. Sennrich and Haddow (2016) for instance proposed to augment each source word with its corresponding part-of-speech tag, lemmatized form and dependency label. Eriguchi et al. (2016) instead replaced the sequential encoder with a tree-based encoder which computes the representation of the source sentence following its parse tree. Stahlberg et al. (2016) let the lattice from a hierarchical phrase-based system guide the decoding process of neural machine translation, which results in two separate models rather than a single end-to-end one. Despite the promising improvements, these explicit approaches are limited in that the trained translation model strictly requires the availability of external tools during inference time.

We instead propose to *implicitly* incorporate linguistic prior based on the idea of multi-task learning (Caruana, 1998; Collobert et al., 2011). More specifically, we design a hybrid decoder for NMT, called NMT+RG, that combines a usual conditional language model and a recently proposed recurrent neural network grammars (RNNGs, Dyer et al., 2016). This is done by plugging in the conventional language model decoder in the place of the buffer in RNNG, while sharing a subset of parameters, such as word vectors, between the language model and RNNG. We train this hybrid model to maximize both the log-probability of a target sentence and the log-probability of a parse action sequence. We use an external parser (Andor et al., 2016) to generate target parse actions, but unlike the previous explicit approaches, we do not need it during test time.

We evaluate the proposed NMT+RG on four language pairs ({Cs, De, Ru, Jp}-En). We observe significant improvements in terms of BLEU scores on three out of four language pairs and RIBES scores on all the language pairs.

2 Neural Machine Translation

Neural machine translation is a recently proposed framework for building a machine translation sys-

tem based purely on neural networks. It is often built as an attention-based encoder-decoder network (Cho et al., 2015) with two recurrent networks—encoder and decoder—and an attention model. The encoder, which is often implemented as a bidirectional recurrent network with long short-term memory units (LSTM, Hochreiter and Schmidhuber, 1997) or gated recurrent units (GRU, Cho et al., 2014), first reads a source sentence represented as a sequence of words $\boldsymbol{x} = (x_1, x_2, \ldots, x_N)$. The encoder returns a sequence of hidden states $\boldsymbol{h} = (h_1, h_2, \ldots, h_N)$. Each hidden state h_i is a concatenation of those from the forward and backward recurrent network: $h_i = \left[\overrightarrow{h}_i; \overleftarrow{h}_i\right]$, where

$$\overrightarrow{h}_i = \overrightarrow{f}_{\text{enc}}(\overrightarrow{h}_{i-1}, V_x(x_i)),$$
$$\overleftarrow{h}_i = \overleftarrow{f}_{\text{enc}}(\overleftarrow{h}_{i+1}, V_x(x_i)).$$

$V_x(x_i)$ refers to the word vector of the i-th source word.

The decoder is implemented as a conditional recurrent language model which models the target sentence, or translation, as

$$\log p(\boldsymbol{y}|\boldsymbol{x}) = \sum_j \log p(y_j|\boldsymbol{y}_{<j}, \boldsymbol{x}),$$

where $\boldsymbol{y} = (y_1, \ldots, y_M)$. Each of the conditional probabilities in the r.h.s is computed by

$$p(y_j = y|\boldsymbol{y}_{<j}, \boldsymbol{x}) = softmax(W_y^\top \tilde{s}_j), \quad (1)$$
$$\tilde{s}_j = \tanh(\boldsymbol{W}_c[s_j; c_j]), \quad (2)$$
$$s_j = f_{\text{dec}}(s_{j-1}, [V_y(y_{j-1}); \tilde{s}_{j-1}]), \quad (3)$$

where f_{dec} is a recurrent activation function, such as LSTM or GRU, and W_y is the output word vector of the word y.

c_j is a time-dependent context vector that is computed by the attention model using the sequence $\boldsymbol{h}$ of hidden states from the encoder. The attention model first compares the current hidden state s_j against each of the hidden states and assigns a scalar score: $\beta_{i,j} = \exp(h_i^\top \boldsymbol{W}_d s_j)$ (Luong et al., 2015). These scores are then normalized across the hidden states to sum to 1, that is $\alpha_{i,j} = \frac{\beta_{i,j}}{\sum_i \beta_{i,j}}$. The time-dependent context vector is then a weighted-sum of the hidden states with these attention weights: $c_j = \sum_i \alpha_{i,j} h_i$.

3 Recurrent Neural Network Grammars

A recurrent neural network grammar (RNNG, Dyer et al., 2016) is a probabilistic syntax-based language model. Unlike a usual recurrent language model (see, e.g., Mikolov et al., 2010), an RNNG simultaneously models both tokens and their tree-based composition. This is done by having a (output) buffer, stack and action history, each of which is implemented as a stack LSTM (sLSTM, Dyer et al., 2015). At each time step, the action sLSTM predicts the next action based on the (current) hidden states of the buffer, stack and action sLSTM. That is,

$$p(a_t = a|\boldsymbol{a}_{<t}) \propto e^{W_a^\top f_{\text{action}}(h_t^{\text{buffer}}, h_t^{\text{stack}}, h_t^{\text{action}})}, \quad (4)$$

where W_a is the vector of the action a. If the selected action is *shift*, the word at the beginning of the buffer is moved to the stack. When the *reduce* action is selected, the top-two words in the stack are reduced to build a partial tree. Additionally, the action may be one of many possible non-terminal symbols, in which case the predicted non-terminal symbol is pushed to the stack.

The hidden states of the buffer, stack and action sLSTM are correspondingly updated by

$$h_t^{\text{buffer}} = \text{StackLSTM}(h_{\text{top}}^{\text{buffer}}, V_y(y_{t-1})), \quad (5)$$
$$h_t^{\text{stack}} = \text{StackLSTM}(h_{\text{top}}^{\text{stack}}, r_t),$$
$$h_t^{\text{action}} = \text{StackLSTM}(h_{\text{top}}^{\text{action}}, V_a(a_{t-1})),$$

where V_y and V_a are functions returning the target word and action vectors. The input vector r_t of the stack sLSTM is computed recursively by

$$r_t = \tanh(\boldsymbol{W}_r[r^d; r^p; V_a(a_t)]),$$

where r^d and r^p are the corresponding vectors of the parent and dependent phrases, respectively (Dyer et al., 2015). This process is iteratively done until the complete parse tree is built.

When the complete sentence is provided, the buffer simply summarizes the shifted words. When the RNNG is used as a generator, the buffer further generates the next word when the selected action is shift. The latter can be done by replacing the buffer with a recurrent language model, which is the idea on which our proposal is based.

4 Learning to Parse and Translate

4.1 NMT+RG

Our main proposal in this paper is to hybridize the decoder of the neural machine translation and the RNNG. We continue from the earlier observation

that we can replace the buffer of RNNG to a recurrent language model that simultaneously summarizes the shifted words as well as generates future words. We replace the RNNG's buffer with the neural translation model's decoder in two steps.

Construction First, we replace the hidden state of the buffer h^{buffer} (in Eq. (5)) with the hidden state of the decoder of the attention-based neural machine translation from Eq. (3). As is clear from those two equations, both the buffer sLSTM and the translation decoder take as input the previous hidden state ($h^{\text{buffer}}_{\text{top}}$ and s_{j-1}, respectively) and the previously decoded word (or the previously shifted word in the case of the RNNG's buffer), and returns its summary state. The only difference is that the translation decoder additionally considers the state $\tilde{s}_{j-1}$. Second, we let the next word prediction of the translation decoder as a generator of RNNG. In other words, the generator of RNNG will output a word, when asked by the shift action, according to the conditional distribution defined by the translation decoder in Eq. (1). Once the buffer sLSTM is replaced with the neural translation decoder, the action sLSTM naturally takes as input the translation decoder's hidden state when computing the action conditional distribution in Eq. (4). We call this hybrid model *NMT+RG*.

Learning and Inference After this integration, our hybrid NMT+RG models the conditional distribution over all possible pairs of translation and its parse given a source sentence, i.e., $p(\boldsymbol{y}, \boldsymbol{a}|\boldsymbol{x})$. Assuming the availability of parse annotation in the target-side of a parallel corpus, we train the whole model jointly to maximize $\mathbb{E}_{(\boldsymbol{x},\boldsymbol{y},\boldsymbol{a})\sim\text{data}}[\log p(\boldsymbol{y}, \boldsymbol{a}|\boldsymbol{x})]$. In doing so, we notice that there are two separate paths through which the neural translation decoder receives error signal. First, the decoder is updated in order to maximize the conditional probability of the correct next word, which has already existed in the original neural machine translation. Second, the decoder is updated also to maximize the conditional probability of the correct parsing action, which is a novel learning signal introduced by the proposed hybridization. Furthermore, the second learning signal affects the encoder as well, encouraging the whole neural translation model to be aware of the syntactic structure of the target language. Later in the experiments, we show that this additional learning signal is useful for translation, even though we discard the RNNG (the stack and action sLSTMs) in the inference time.

	Train.	Dev.	Test	Voc. (*src*, *tgt*, *act*)
Cs-En	134,453	2,656	2,999	(33,867, 27,347, 82)
De-En	166,313	2,169	2,999	(33,820, 30,684, 80)
Ru-En	131,492	2,818	2,998	(32,442, 27,979, 82)
Jp-En	100,000	1,790	1,812	(23,509, 28,591, 80)

Table 1: Statistics of parallel corpora.

4.2 Knowledge Distillation for Parsing

A major challenge in training the proposed hybrid model is that there is not a parallel corpus augmented with gold-standard target-side parse, and vice versa. In other words, we must either parse the target-side sentences of an existing parallel corpus or translate sentences with existing gold-standard parses. As the target task of the proposed model is translation, we start with a parallel corpus and annotate the target-side sentences. It is however costly to manually annotate any corpus of reasonable size (Table 6 in Alonso et al., 2016).

We instead resort to noisy, but automated annotation using an existing parser. This approach of automated annotation can be considered along the line of recently proposed techniques of knowledge distillation (Hinton et al., 2015) and distant supervision (Mintz et al., 2009). In knowledge distillation, a teacher network is trained purely on a training set with ground-truth annotations, and the annotations predicted by this teacher are used to train a student network, which is similar to our approach where the external parser could be thought of as a teacher and the proposed hybrid network's RNNG as a student. On the other hand, what we propose here is a special case of distant supervision in that the external parser provides noisy annotations to otherwise an unlabeled training set.

Specifically, we use SyntaxNet, released by Andor et al. (2016), on a target sentence. We convert a parse tree into a sequence of one of three transition actions (SHIFT, REDUCE-L, REDUCE-R). We label each REDUCE action with a corresponding dependency label and treat it as a more fine-grained action.

5 Experiments

5.1 Language Paris and Corpora

We compare the proposed NMT+RG against the baseline model on four different language pairs– Jp-En, Cs-En, De-En and Ru-En. The basic statistics of the training data are presented in Table 1.

Ja We use the ASPEC corpus ("train1.txt") from the WAT'16 Jp-En translation task. We tokenize each Japanese sentence with *KyTea* (Neubig et al., 2011) and preprocess according to the recommendations from WAT'16 (WAT, 2016). We use the first 100K sentence pairs of length shorter than 50 for training. The vocabulary is constructed with all the unique tokens that appear at least twice in the training corpus. We use "dev.txt" and "test.txt" provided by WAT'16 respectively as development and test sets.

Cs, De and Ru We use News Commentary v8 from WMT'16. We use the tokenizer from Moses (Koehn et al., 2007), and build a vocabulary of each language using unique tokens that appear at least 6, 6 and 5 times respectively for Cs, Ru and De. The target-side (English) vocabulary was constructed with all the unique tokens appearing more than three times in each corpus. We only use sentence pairs of length 50 or less for training. We use "newstest2015" and "newstest2016" as development and test sets respectively.

5.2 Models, Learning and Inference

In all our experiments, each recurrent network has a single layer of LSTM units of 256 dimensions, and the word vectors and the action vectors are of 256 and 128 dimensions, respectively. To reduce computational overhead, we use BlackOut (Ji et al., 2015) with 2000 negative samples and $\alpha = 0.4$. For the proposed NMT+RG, we share the target word vectors between the decoder (buffer) and the stack sLSTM. We use stochastic gradient descent with minibatches of 128 examples. The learning rate starts from 1.0, and is halved each time the perplexity on the development set increases. We clip the norm of the gradient (Pascanu et al., 2012) with the threshold set to 3.0 (2.0 for the baseline models on Ru-En and Cs-En to avoid NaN and Inf). We use beam search in the inference time, with the beam width selected based on the development set performance.

The more details are described in the supplementary material.

5.3 Results and Analysis

In Table 2, we report the translation qualities of the tested models on all the four language pairs. We report both BLEU (Papineni et al., 2002) and RIBES (Isozaki et al., 2010). Except for De-En, measured in BLEU, we observe the statistically significant improvement by the proposed NMT+RG over the baseline model. It is worthwhile to note that these significant improvements have been achieved *without* any additional parameters nor computational overhead in the inference time.

	De-En	Ru-En	Cs-En	Jp-En
BLEU				
NMT	16.61	12.03	11.22	17.88
NMT+RG	16.41	**12.46**†	**12.06**†	**18.84**†
RIBES				
NMT	73.75	69.56	69.59	71.27
NMT+RG	**75.03**†	**71.04**†	**70.39**†	**72.25**†

Table 2: BLEU and RIBES scores by the baseline and proposed models on the test set. We use the bootstrap resampling method from Koehn (2004) to compute the statistical significance. We use † to mark those significant cases with $p < 0.005$.

Ablation Since each component in RNNG may be omitted, we ablate each component in the proposed NMT+RG to verify their necessity.[1] As shown in Table 3, we see that the best performance could only be achieved when all the three components were present. Removing the stack had the most adverse effect, which was found to be the case for parsing as well by Kuncoro et al. (2017).

Jp-En (Dev)	BLEU
NMT+RG	18.60
w/o Buffer	18.02
w/o Action	17.94
w/o Stack	17.58
NMT	17.75

Table 3: Effect of each component in RNNG.

6 Conclusion

We propose a hybrid model, to which we refer as NMT+RG, that combines the decoder of an attention-based neural translation model with the RNNG. This model learns to parse and translate simultaneously, and training it encourages both the encoder and decoder to better incorporate linguistic priors. Our experiments confirmed its effectiveness on four language pairs. The RNNG can in principle be trained without ground-truth parses, and this would eliminate the need of external parsers completely. We leave the investigation

[1] Since the buffer is the decoder, it is not possible to completely remove it. Instead we simply remove the dependency of the action distribution on it.

into this possibility for future research.

Acknowledgments

We thank Yuchen Qiao and Kenjiro Taura for their help to speed up the implementations of training and also Kazuma Hashimoto for his valuable comments and discussions. This work was supported by CREST, JST, and JSPS KAKENHI Grant Number 15J12597. KC thanks support by Facebook, Google and NVIDIA.

References

Héctor Martínez Alonso, Djamé Seddah, and Benoît Sagot. 2016. From noisy questions to minecraft texts: Annotation challenges in extreme syntax scenario. In *Proceedings of the 2nd Workshop on Noisy User-generated Text (WNUT)*. The COLING 2016 Organizing Committee, pages 13–23.

Daniel Andor, Chris Alberti, David Weiss, Aliaksei Severyn, Alessandro Presta, Kuzman Ganchev, Slav Petrov, and Michael Collins. 2016. Globally normalized transition-based neural networks. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics*. Association for Computational Linguistics, pages 2442–2452.

Rich Caruana. 1998. Multitask learning. In *Learning to learn*, Springer, pages 95–133.

Kyunghyun Cho, Aaron Courville, and Yoshua Bengio. 2015. Describing multimedia content using attention-based encoder-decoder networks. *IEEE Transactions on Multimedia* 17(11):1875–1886.

Kyunghyun Cho, Bart van Merrienboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. 2014. Learning phrase representations using rnn encoder–decoder for statistical machine translation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, pages 1724–1734.

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*. Association for Computational Linguistics, pages 1693–1703.

Ronan Collobert, Jason Weston, Léon Bottou, Michael Karlen, Koray Kavukcuoglu, and Pavel Kuksa. 2011. Natural language processing (almost) from scratch. *Journal of Machine Learning Research* 12:2493–2537.

Chris Dyer, Miguel Ballesteros, Wang Ling, Austin Matthews, and A. Noah Smith. 2015. Transition-based dependency parsing with stack long short-term memory. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing*. Association for Computational Linguistics, pages 334–343.

Chris Dyer, Adhiguna Kuncoro, Miguel Ballesteros, and A. Noah Smith. 2016. Recurrent neural network grammars. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. Association for Computational Linguistics, pages 199–209.

Akiko Eriguchi, Kazuma Hashimoto, and Yoshimasa Tsuruoka. 2016. Tree-to-sequence attentional neural machine translation. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics*. Association for Computational Linguistics, pages 823–833.

Kazuma Hashimoto and Yoshimasa Tsuruoka. 2017. Neural Machine Translation with Source-Side Latent Graph Parsing. *arXiv preprint arXiv:1702.02265* .

Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. 2015. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531* .

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural Comput.* 9(8):1735–1780.

Hideki Isozaki, Tsutomu Hirao, Kevin Duh, Katsuhito Sudoh, and Hajime Tsukada. 2010. Automatic evaluation of translation quality for distant language pairs. In *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, pages 944–952.

Shihao Ji, S. V. N. Vishwanathan, Nadathur Satish, Michael J. Anderson, and Pradeep Dubey. 2015. Blackout: Speeding up recurrent neural network language models with very large vocabularies. *Proceedings of International Conference on Learning Representations 2015* .

Rafal Józefowicz, Wojciech Zaremba, and Ilya Sutskever. 2015. An empirical exploration of recurrent network architectures. In *Proceedings of the 32nd International Conference on Machine Learning*. pages 2342–2350.

Philipp Koehn. 2004. Statistical significance tests for machine translation evaluation. In *Proceedings of the 2004 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, pages 388–395.

Philipp Koehn, Hieu Hoang, Alexandra Birch, Chris Callison-Burch, Marcello Federico, Nicola Bertoldi, Brooke Cowan, Wade Shen, Christine Moran, Richard Zens, Chris Dyer, Ondrej Bojar, Alexandra Constantin, and Evan Herbst. 2007. Moses: Open source toolkit for statistical machine translation. In

Proceedings of the 45th Annual Meeting of the Association for Computational Linguistics Companion Volume Proceedings of the Demo and Poster Sessions. Association for Computational Linguistics, pages 177–180.

Adhiguna Kuncoro, Miguel Ballesteros, Lingpeng Kong, Chris Dyer, Graham Neubig, and Noah A Smith. 2017. What do recurrent neural network grammars learn about syntax? In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics (to appear)*.

Jason Lee, Kyunghyun Cho, and Thomas Hofmann. 2016. Fully character-level neural machine translation without explicit segmentation. *arXiv preprint arXiv:1610.03017* .

Thang Luong, Hieu Pham, and Christopher D. Manning. 2015. Effective approaches to attention-based neural machine translation. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, pages 1412–1421.

Tomáš Mikolov, Martin Karafiát, Lukáš Burget, Jan Černocký, and Sanjeev Khudanpur. 2010. Recurrent neural network based language model. In *Proceedings of the 11th Annual Conference of the International Speech Communication Association (INTERSPEECH 2010)*. International Speech Communication Association, pages 1045–1048.

Mike Mintz, Steven Bills, Rion Snow, and Dan Jurafsky. 2009. Distant supervision for relation extraction without labeled data. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP*. Association for Computational Linguistics, pages 1003–1011.

Graham Neubig, Yosuke Nakata, and Shinsuke Mori. 2011. Pointwise prediction for robust, adaptable japanese morphological analysis. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*. Association for Computational Linguistics, Portland, Oregon, USA, pages 529–533. http://www.aclweb.org/anthology/P11-2093.

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

Razvan Pascanu, Tomas Mikolov, and Yoshua Bengio. 2012. Understanding the exploding gradient problem. *arXiv preprint arXiv:1211.5063* abs/1211.5063.

Rico Sennrich and Barry Haddow. 2016. Linguistic input features improve neural machine translation. In *Proceedings of the First Conference on Machine Translation*. Association for Computational Linguistics, pages 83–91.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016. Neural machine translation of rare words with subword units. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics*. Association for Computational Linguistics, pages 1715–1725.

Xing Shi, Inkit Padhi, and Kevin Knight. 2016. Does string-based neural mt learn source syntax? In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, pages 1526–1534.

Felix Stahlberg, Eva Hasler, Aurelien Waite, and Bill Byrne. 2016. Syntactically guided neural machine translation. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics*. Association for Computational Linguistics, pages 299–305.

WAT. 2016. http://lotus.kuee.kyoto-u.ac.jp/WAT/baseline/dataPreparationJE.html.

Yonghui Wu, Mike Schuster, Zhifeng Chen, Quoc V. Le, Mohammad Norouzi, Wolfgang Macherey, Maxim Krikun, Yuan Cao, Qin Gao, Klaus Macherey, Jeff Klingner, Apurva Shah, Melvin Johnson, Xiaobing Liu, Lukasz Kaiser, Stephan Gouws, Yoshikiyo Kato, Taku Kudo, Hideto Kazawa, Keith Stevens, George Kurian, Nishant Patil, Wei Wang, Cliff Young, Jason Smith, Jason Riesa, Alex Rudnick, Oriol Vinyals, Greg Corrado, Macduff Hughes, and Jeffrey Dean. 2016. Google's neural machine translation system: Bridging the gap between human and machine translation. *arXiv preprint arXiv:1609.08144* .

Barret Zoph, Ashish Vaswani, Jonathan May, and Kevin Knight. 2016. Simple, Fast Noise-Contrastive Estimation for Large RNN Vocabularies. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. pages 1217–1222.

Supplemental Material

7 Dataset Preparation

We mapped all the low-frequency words to the unique symbol "UNK" and inserted a special symbol "EOS" at the end of both source and target sentences.

News Commentary v8 corpus ({Cs, De, Ru}-EN) We removed noisy metacharacters. We also excluded the sentence pairs which include empty lines in either a source sentence or a target sentence.

8 Models, Learning and Inference

The RNNG's stack computes the vector of a dependency parse tree which consists of the generated target words by the buffer. Since the complete parse tree has a "ROOT" node, the special token of the end of a sentence ("EOS") is considered as the ROOT.

Each weight is initialized from the uniform distribution $[-0.1, 0.1]$. The bias vectors and the weights of the softmax and BlackOut are initialized to be zero. When employing BlackOut, we shared the negative samples of each target word in a sentence in training time (Hashimoto and Tsuruoka, 2017), which is similar to the previous work (Zoph et al., 2016). The forget gate biases of LSTMs and Stack-LSTMs are initialized to 1 as recommended in Józefowicz et al. (2015). When the perplexity of development data increased in training time, we halved the learning rate of stochastic gradient descent and reloaded the previous model.