Character-based Decoding in Tree-to-Sequence Attention-based Neural Machine Translation

Akiko Eriguchi, Kazuma Hashimoto, and **Yoshimasa Tsuruoka**
The University of Tokyo, 7-3-1 Hongo, Bunkyo-ku, Tokyo, Japan
{eriguchi, hassy, tsuruoka}@logos.t.u-tokyo.ac.jp

Abstract

This paper reports our systems (UT-AKY) submitted in the 3rd Workshop of Asian Translation 2016 (WAT'16) and their results in the English-to-Japanese translation task. Our model is based on the tree-to-sequence Attention-based NMT (ANMT) model proposed by Eriguchi et al. (2016). We submitted two ANMT systems: one with a word-based decoder and the other with a character-based decoder. Experimenting on the English-to-Japanese translation task, we have confirmed that the character-based decoder can cover almost the full vocabulary in the target language and generate translations much faster than the word-based model.

1 Introduction

End-to-end Neural Machine Translation (NMT) models have recently achieved state-of-the-art results in several translation tasks (Luong et al., 2015a; Luong et al., 2015b). Those NMT models are based on the idea of sequence-to-sequence learning (Sutskever et al., 2014), where both of the source and the target sentences are considered as a sequence of symbols (e.g. words or characters) and they are directly converted via a vector space. The sequence of symbols on the source side is input into a vector space, and the sequence of symbols on the target side is output from the vector space. In the end-to-end NMT models, the above architectures are embodied by a single neural network.

The optimal unit for NMT is an important research question discussed in the community. Early NMT models employ a word as a unit of the sequence (Cho et al., 2014b; Sutskever et al., 2014). Sennrich et al. (2016) have used a Byte-Pair Encoding (BPE) method to create a sub-word level vocabulary according to the frequencies of sub-word appearance in the corpus. They successfully replaced a large word vocabulary in German and Russian with a much smaller sub-word vocabulary. They have also shown that their sub-word-based NMT model gives better translations than the word-based NMT models.

The smallest unit of a sequence of text data is a character. The character-based approach has attracted much attention in the field of NMT, because it enables an NMT model to handle all of the tokens in the corpus (Costa-jussà and Fonollosa, 2016; Chung et al., 2016). A hybrid model of the word-based and the character-based model has also been proposed by Luong and Manning (2016). These studies reported the success and effectiveness in translating the out-of-vocabulary words.

In this paper, we apply character-based decoding to a tree-based NMT model (Eriguchi et al., 2016). The existing character-based models focus only on the sequence-based NMT models. The objective of this paper is to analyze the results of the character-based decoding in the tree-based NMT model. We also enrich the tree-based encoder with syntactic features. Figure 1 shows an overview of our system. We conducted the English-to-Japanese translation task on the WAT'16 dataset. The results of our character-based decoder model show that its translation accuracy is lower than that of the word-based decoder model by 1.34 BLEU scores, but the character-based decoder model needed much less time to generate a sentence.

2 Neural Machine Translation

End-to-end NMT models have recently outperformed phrase-based statistical machine translation (SMT) models in several languages (Luong et al., 2015a; Eriguchi et al., 2016). Those NMT models are ba-

Proceedings of the 3rd Workshop on Asian Translation,
pages 175–183, Osaka, Japan, December 11-17 2016.

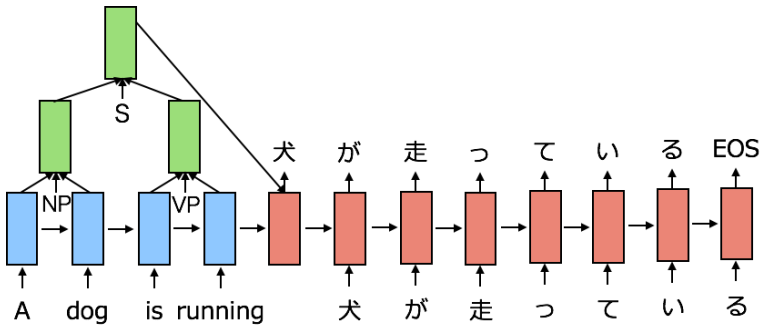

Figure 1: WAT'16 Our system: tree-to-character Neural Machine Translation model.

sically composed of two processes called an *encoder* and a *decoder*. We feed a sequence of words $\boldsymbol{x} = (x_1, x_2, \cdots, x_n)$ in the source language into the encoder, and the encoder converts the input data into a vector space until the last n-th word in the sentence is input. Recurrent Neural Networks (RNNs) are used to obtain the vectors of the sequence of data in the recent NMT models. The i-th hidden state $\boldsymbol{h}_i \in \mathbb{R}^{d\times 1}$ in the RNN holds a vector computed by the current input x_i and the previous hidden state $\boldsymbol{h}_{i-1} \in \mathbb{R}^{d\times 1}$:

$$\boldsymbol{h}_i \quad = \quad RNN_{encoder}(Embed(x_i), \boldsymbol{h}_{i-1}), \tag{1}$$

where $Embed(x_i)$ is the word embedding vector of the i-th source word x_i. $\boldsymbol{h}_0$ is set to $\mathbf{0}$.

Another RNN is used as the decoder to obtain the vectors for predicting the words on the target side. The j-th hidden state $\boldsymbol{s}_j \in \mathbb{R}^{d\times 1}$ of the RNN is computed from the previous hidden state $\boldsymbol{s}_{j-1} \in \mathbb{R}^{d\times 1}$ and the previous output word y_{j-1} as follows:

$$\boldsymbol{s}_j \quad = \quad RNN_{decoder}(Embed(y_{j-1}), \boldsymbol{s}_{j-1}), \tag{2}$$

where $Embed(y_{j-1})$ is the word embedding vector of the $(j-1)$-th target word y_{j-1}. The first decoder $\boldsymbol{s}_1$ is initialized with the last hidden state of the encoder $\boldsymbol{h}_n$.

The NMT models that simply connect the above two types of RNNs cannot capture strong relations between the encoder units and the decoder unit, and they often fail to translate a long sentence. An attention mechanism has been introduced to solve the problem by creating an attention path so that the hidden states of the decoder can access each hidden state of the encoder (Bahdanau et al., 2015). Luong et al. (2015a) have refined the calculation of the attention mechanism. In the decoder process, the attention score $\boldsymbol{\alpha}_j(i)$ is computed by the j-th hidden state of the decoder $\boldsymbol{s}_j$ and each hidden state of the encoder $\boldsymbol{h}_i$ as follows:

$$\boldsymbol{\alpha}_j(i) \quad = \quad \frac{\exp(\boldsymbol{h}_i \cdot \boldsymbol{s}_j)}{\sum_{k=1}^{n} \exp(\boldsymbol{h}_k \cdot \boldsymbol{s}_j)}, \tag{3}$$

where $\cdot$ represents the inner product, and its value of $\boldsymbol{h}_i \cdot \boldsymbol{s}_j$ is the similarity score between $\boldsymbol{h}_i$ and $\boldsymbol{s}_j$. The j-th context vector $\boldsymbol{d}_j \in \mathbb{R}^{d\times 1}$ are computed as the summation of the hidden states:

$$\boldsymbol{d}_j \quad = \quad \sum_{i=1}^{n} \boldsymbol{\alpha}_j(i)\boldsymbol{h}_i, \tag{4}$$

where each of the hidden states is weighted by $\boldsymbol{\alpha}_j(i)$. We compute the j-th final decoder $\tilde{\boldsymbol{s}}_j \in \mathbb{R}^{d\times 1}$ as follows:

$$\tilde{\boldsymbol{s}}_j = \tanh(\boldsymbol{W}_d[\boldsymbol{s}_j;\ \boldsymbol{d}_j] + \boldsymbol{b}_d), \tag{5}$$

where $[\boldsymbol{s}_j; \boldsymbol{d}_j] \in \mathbb{R}^{2d\times 1}$ denotes the concatenation of $\boldsymbol{s}_j$ and $\boldsymbol{d}_j$. $\boldsymbol{W}_d \in \mathbb{R}^{d\times 2d}$ is a weight matrix. $\boldsymbol{b}_d \in \mathbb{R}^{d\times 1}$ is a bias vector. The conditional probability of predicting an output is defined as follows:

$$p(y_j|\boldsymbol{x}, \boldsymbol{y}_{<j}) \quad = \quad \text{softmax}(\boldsymbol{W}_s\tilde{\boldsymbol{s}}_j + \boldsymbol{b}_s), \tag{6}$$

where $\boldsymbol{W}_s \in \mathbb{R}^{d \times d}$ is a matrix and $\boldsymbol{b}_s \in \mathbb{R}^{d \times 1}$ is a bias.

The objective function to train the NMT models is defined as the sum of the log-likelihoods of the translation pairs in the training data:

$$J(\boldsymbol{\theta}) = \frac{1}{|\mathcal{D}|} \sum_{(\boldsymbol{x},\boldsymbol{y}) \in \mathcal{D}} \log p(\boldsymbol{y}|\boldsymbol{x}), \tag{7}$$

where $\mathcal{D}$ denotes the set of parallel sentence pairs. When training the model, the parameters $\boldsymbol{\theta}$ are updated by Stochastic Gradient Descent (SGD).

3 Our systems: tree-to-character attention-based NMT model

Our system is mostly based on the tree-to-sequence Attention-based NMT (ANMT) model described in Eriguchi et al. (2016) which has a tree-based encoder and a sequence-based decoder. They employed Long Short-Term Memory (LSTM) as the units of RNNs (Hochreiter and Schmidhuber, 1997; Gers et al., 2000). In their proposed tree-based encoder, the phrase vectors are computed from their child states by using Tree-LSTM units (Tai et al., 2015), following the phrase structure of a sentence. We incorporate syntactic features into the tree-based encoder, and the k-th phrase vector $\boldsymbol{h}_k^{(p)} \in \mathbb{R}^{d \times 1}$ in our system is computed as follows:

$$\begin{aligned}
\boldsymbol{i}_k &= \sigma(\boldsymbol{U}_l^{(i)} \boldsymbol{h}_k^l + \boldsymbol{U}_r^{(i)} \boldsymbol{h}_k^r + \boldsymbol{W}^{(i)} \boldsymbol{z}_k + \boldsymbol{b}^{(i)}), & \boldsymbol{f}_k^l &= \sigma(\boldsymbol{U}_l^{(f_l)} \boldsymbol{h}_k^l + \boldsymbol{U}_r^{(f_l)} \boldsymbol{h}_k^r + \boldsymbol{W}^{(f_l)} \boldsymbol{z}_k + \boldsymbol{b}^{(f_l)}), \\
\boldsymbol{f}_k^r &= \sigma(\boldsymbol{U}_l^{(f_r)} \boldsymbol{h}_k^l + \boldsymbol{U}_r^{(f_r)} \boldsymbol{h}_k^r + \boldsymbol{W}^{(f_r)} \boldsymbol{z}_k + \boldsymbol{b}^{(f_r)}), & \boldsymbol{o}_k &= \sigma(\boldsymbol{U}_l^{(o)} \boldsymbol{h}_k^l + \boldsymbol{U}_r^{(o)} \boldsymbol{h}_k^r + \boldsymbol{W}^{(o)} \boldsymbol{z}_k + \boldsymbol{b}^{(o)}), \\
\tilde{\boldsymbol{c}}_k &= \tanh(\boldsymbol{U}_l^{(\tilde{c})} \boldsymbol{h}_k^l + \boldsymbol{U}_r^{(\tilde{c})} \boldsymbol{h}_k^r + \boldsymbol{W}^{(o)} \boldsymbol{z}_k + \boldsymbol{b}^{(\tilde{c})}), & \boldsymbol{c}_k &= \boldsymbol{i}_k \odot \tilde{\boldsymbol{c}}_k + \boldsymbol{f}_k^l \odot \boldsymbol{c}_k^l + \boldsymbol{f}_k^r \odot \boldsymbol{c}_k^r, \\
\boldsymbol{h}_k^{(p)} &= \boldsymbol{o}_k \odot \tanh(\boldsymbol{c}_k),
\end{aligned} \tag{8}$$

where each of $\boldsymbol{i}_k$, $\boldsymbol{o}_k$, $\tilde{\boldsymbol{c}}_k$, $\boldsymbol{c}_k$, $\boldsymbol{c}_k^l$, $\boldsymbol{c}_k^r$, $\boldsymbol{f}_k^l$, and $\boldsymbol{f}_k^r \in \mathbb{R}^{d \times 1}$ denotes an input gate, an output gate, a state for updating the memory cell, a memory cell, the memory cells of the left child node and the right child node, the forget gates for the left child and for the right child, respectively. $\boldsymbol{W}^{(\cdot)} \in \mathbb{R}^{d \times d}$ and $\boldsymbol{U}^{(\cdot)} \in \mathbb{R}^{d \times m}$ are weight matrices, and $\boldsymbol{b}^{(\cdot)} \in \mathbb{R}^{d \times 1}$ is a bias vector. $\boldsymbol{z}_k \in \mathbb{R}^{m \times 1}$ is an embedding vector of the phrase category label of the k-th node. $\sigma(\cdot)$ denotes the logistic function. The operator $\odot$ is element-wise multiplication.

The decoder in our system outputs characters one by one. Note that the number of characters in a language is far smaller than the vocabulary size of the words in the language. The character-based approaches thus enable us to significantly speed up the softmax computation for generating a symbol, and we can train the NMT model and generate translations much faster. Moreover, all of the raw text data are directly covered by the character units, and therefore the decoder in our system requires few preprocessing steps such as segmentation and tokenization.

We also use the *input-feeding* technique (Luong et al., 2015a) to improve translation accuracy. The j-th hidden state of the decoder is computed in our systems as follows:

$$\boldsymbol{s}_j = RNN_{decoder}(Embed(y_{j-1}), [\boldsymbol{s}_{j-1};\ \tilde{\boldsymbol{s}}_{j-1}]), \tag{9}$$

where $[\boldsymbol{s}_{j-1};\ \tilde{\boldsymbol{s}}_{j-1}] \in \mathbb{R}^{2d \times 1}$ denotes the concatenation of $\boldsymbol{s}_{j-1}$ and $\tilde{s}_{j-1}$.

4 Experiment in WAT'16 task

4.1 Experimental Setting

We conducted experiments for our system using the 3rd Workshop of Asian Translation 2016 (WAT'16)[1] English-to-Japanese translation task (Nakazawa et al., 2016a). The data set is the Asian Scientific Paper Excerpt Corpus (ASPEC) (Nakazawa et al., 2016b). The data setting followed the ones in Zhu (2015) and Eriguchi et al. (2016). We collected 1.5 million pairs of training sentences from *train-1.txt* and the first

[1] http://lotus.kuee.kyoto-u.ac.jp/WAT/

	Sentences	Parsed sentences
Train dataset	1,346,946	1,346,946
Dev. dataset	1,790	1,789
Test dataset	1,812	1,811

Table 1: The details of dataset in the ASPEC corpus.

	Vocabulary size
$\|V_{word}\|$ in English	87,796
$\|V_{word}\|$ in Japanese	65,680
$\|V_{character}\|$ in Japanese	3,004

Table 2: Vocabulary sizes in the training models.

half of *train-2.txt*. We removed the sentences whose lengths are greater than 50 words. In the tree-based encoder, binary trees of the source sentences were obtained by Enju (Miyao and Tsujii, 2008), which is a probabilistic HPSG parser. We used phrase category labels as the syntactic features in the proposed tree-based encoder. There are 19 types of phrase category labels given by Enju. In the word-based decoder model, we employed KyTea (Neubig et al., 2011) as the segmentation tool for the Japanese sentences. We performed the preprocessing steps of the data as recommended in WAT'16.[2] Table 1 and Table 2 show the details of the final dataset and the vocabulary sizes in our experiments. Each vocabulary includes the words and the characters whose frequencies exceed five or two in the training data, respectively. The out-of-vocabulary words are mapped into a special token i.e. "UNK". As a result, the vocabulary size of the characters in Japanese is about 22 times smaller than that of the words.

NMT models are often trained on a limited vocabulary, because the high computational cost of the softmax layer for target word generation is usually the bottleneck when training an NMT model. In the word-based models, we use the BlackOut sampling method (Ji et al., 2016) to approximately compute the softmax layer. The parameter setting of BlackOut follows Eriguchi et al. (2016). In the character-based models, we use the original softmax in the softmax layer. All of the models are trained on CPUs.[3] We employed multi-threading programming to update the parameters in a mini-batch in parallel. The training times of the single word-based model and the single character-based model were about 11 days and 7 days, respectively.

We set the dimension size of the hidden states to 512 in both of the LSTMs and the Tree LSTMs. The dimension size of embedding vectors is set to 512 for the words and to 256 for the characters. In our proposed tree-based encoder, we use 64 and 128 for the dimension size of the phrase label embedding. The model parameters are uniformly initialized in $[-0.1, 0.1]$, except that the forget biases are filled with 1.0 as recommended in Józefowicz et al. (2015). Biases, softmax weights and BlackOut weights are filled with 0. We shuffle the training data randomly per each epoch. All of the parameters are updated by the plain SGD algorithm with a mini-batch size of 128. The learning rate of SGD is set to 1.0, and we halve it when the perplexity of development data becomes worse. The value of gradient norm clipping (Pascanu et al., 2012) is set to 3.0.

We use a beam search in order to obtain a proper translation sentence with the size of 20 and 5 in the word-based decoder and the character-based decoder, respectively. The maximum length of a generated sentence is set to 100 in the word-based decoder and to 300 in the character-based decoder. Cho et al. (2014a) reported that an RNN-based decoder generates a shorter sentence when using the original beam search. We used the beam search method proposed in Eriguchi et al. (2016) in order to output longer translations. We evaluated the models by the BLEU score (Papineni et al., 2002) and the RIBES score (Isozaki et al., 2010) employed as the official evaluation metrics in WAT'16.

4.2 Experimental Results

Table 3 shows the experimental results of the character-based models, the word-based models and the baseline SMT systems. BP denotes the brevity penalty in the BLEU score. First, we can see small improvements in the RIBES score of the single tree-to-sequence ANMT models with the character-based decoder using syntactic features, compared to our proposed baseline system. System 1 is one of our submitted systems. The translations are output by the ensemble of the three models, and we used a simple

[2] http://lotus.kuee.kyoto-u.ac.jp/WAT/baseline/dataPreparationJE.html

[3] 16 threads on Intel(R) Xeon(R) CPU E5-2667 v3 @ 3.20GHz

Model	BLEU (BP)	RIBES
Character-based decoder		
Our proposed baseline: tree-to-seq ANMT model	31.52 (0.96)	79.39
+ phrase label input ($m = 64$)	31.49 (0.95)	79.51
+ phrase label input ($m = 128$)	31.46 (0.95)	79.48
System1: Ensemble of the above three models w/ the original beam search	33.21 (0.86)	81.45
Word-based decoder		
seq-to-seq ANMT model (Luong et al., 2015a)	34.64 (0.93)	81.60
tree-to-seq ANMT model ($d = 512$)	34.91 (0.92)	81.66
System2: Ensemble of the tree-to-seq ANMT models (Eriguchi et al., 2016)	36.95 (0.92)	82.45
Baseline system		
Baseline 1: Phrase-based SMT	29.80 (——)	69.19
Baseline 2: Hierarchical Phrase-based SMT	32.56 (——)	74.70
Baseline 3: Tree-to-string SMT	33.44 (——)	75.80

Table 3: The results of our systems and the baseline systems.

beam search to confirm how much it effects the BLEU scores in the character-based models. We showed the results of our proposed character-based decoder models by using the beam search method proposed in Eriguchi et al. (2016). We collects the statistics of the relation between the source sentence length (L_s) and the target sentence length (L_t) from training dataset and adds its log probability ($\log p(L_t|L_s)$) as the penalty of the beam score when the model predicts "EOS". The BLEU score is sensitive to the value of BP, and we observe the same trend in that the character-based approaches generate a shorter sentence by the original beam search. As a result, each of the character-based models can generate longer translation by +0.09 BP scores at least than System 1 using the original beam search.

The word-based tree-to-sequece decoder model shows slightly better performance than the word-based sequence-to-sequence ANMT model (Luong et al., 2015a) in both of the scores. The results of the baseline systems are the ones reported in Nakazawa et al. (2015). Compared to these SMT baselines, each of the character-based models clearly outperforms the phrase-based system in both of the BLEU and RIBES scores. Although the hierarchical phrase-based SMT system and the tree-to-string SMT system outperforms the single character-based model without phrase label inputs by +1.04 and by +1.92 BLEU scores, respectively, our best ensemble of character-based models shows better performance (+5.65 RIBES scores) than the tree-to-string SMT system.

All the submitted systems are evaluated by pairwise cloudsourcing. System 1 is ranked as the 9th out of the 10 submitted models, and System 2 is ranked as the 6th.

5 Discussion

Table 4 shows a comparison of the speeds to predict the next word between the word-based decoder and the character-based decoder when generating a sentence by a beam size of 1. The character-based decoder is about 41 times faster than the word-based decoder. It is because the time to output a word by using a softmax layer is roughly proportional to the vocabulary sizes of the decoders. In addition to the low cost of predicting the outputs in the character-based model, the character-based decoder requires the smaller size of beam search than the word-based model. The word-based decoder requires a beam size of 20 when decoding, but a beam size of 5 is enough for the character-based decoder. It requires smaller beam size for the character-based decoder to find the best hypothesis. We therefore conclude that the character-based model works more efficiently as a translation model than the word-based model in terms of the cost of the outputs.

Some translation examples of our systems are shown in Table 5. There are two types of source sen-

	Time (msec / sentence)
Word-based decoder	363.7
Character-based decoder	8.8

Table 4: Comparison of the times when outputting a sentence.

Source sentence A	The electric power generation was the 380 micro watt.
Ground truth A	発電量は３８０マイクロワットであった。
Word-based	発電は３８０ UNKW であった。
Character-based	発電は３８０マイクロワットであった。
+ label input ($m = 64$)	電力発電は３８０マイクロワットであった。
Source sentence B	This paper describes development outline of low-loss forsterite porcelain.
Ground truth B	低損失フォルステライト磁器の開発概要などをのべた 。
Word-based	ここでは，UNK UNK の開発概要を述べた。
Character-based	低損失フォルステライト磁器の開発概要を述べた。
+ label input ($m = 64$)	低損失フォルステライト磁器の開発概要を述べた。

Table 5: Translation examples of test data.

tences, the ground truth target sentences, and the translated sentences by the word-based model, by the character-based model, and the character-based model using the syntactic features embedded with a dimension size of 64. The words in the same color semantically correspond to each other.

In sentence A, we can see that the character-based models correctly translated the source word "micro" with the characters "マイクロ", while the word-based decoder requires the unknown replacement (Luong et al., 2015b; Jean et al., 2015). When the word-based model outputs the target word "UNK", the source phrase "micro watt" has the highest attention score ($\alpha = 0.78$) and the source word "micro" has the second highest attention score ($\alpha = 0.16$). The word-based decoder model is successful in outputting the original number ("３８０") in the source side to the target side, and both of the character-based decoder model has also succeeded in predicting a correct sequence of characters "３", "８", and "０" one by one. The training dataset includes the translation of the word "380" into the characters"３８０", so the character-based model can be trained without copy mechanism (Ling et al., 2016) in this case.

In sentence B, the character-based models successfully translate "low-loss forsterite porcelain" into "低損失フォルステライト磁器". The word-based decoder model generates two "UNK"s. The source word "forsterite" ("フォルステライト" in Japanese) has the highest attention score ($\alpha = 0.23$) to the first "UNK", and the phrase "forsterite porcelain" has the second highest attention score ($\alpha = 0.21$). The second "UNK" is softly alined to the source word "porcelain" ("磁器" in Japanese) with the highest attention score ($\alpha = 0.26$) and to the source phrase "forsterite porcelain" with the second highest attention score ($\alpha = 0.16$).

6 Related Work

There are many NMT architectures: a convolutional network-based encoder (Kalchbrenner and Blunsom, 2013), sequence-to-sequence models (Cho et al., 2014b; Sutskever et al., 2014) and a tree-based encoder (Eriguchi et al., 2016). The objective of these research efforts focused on how to encode the data in a language into a vector space and to decode the data in another language from the vector space. Sennrich and Haddow (2016) improved the vector space of NMT models by adding linguistic features. The text data is basically considered as the sequence of words.

The word-based NMT models cannot usually cover the whole vocabulary in the corpus. Rare words are mapped into "unknown" words when the NMT models are trained. Luong et al. (2015b) proposed an ex post facto replacement technique for such unknown words, and Jean et al. (2015) replace the unknown word with the source word which has the highest attention score to the unknown word. Sennrich et al. (2016) adopted a sub-word as a unit of the vocabulary for the NMT models and created the sub-word-based vocabulary by the BPE method. The vocabulary based on the sub-words can cover much more words in German and Russian, compared to the vocabulary based on the words. The NMT models trained with the sub-word-based vocabulary performed better than the ones trained on the word-based vocabulary.

Since the smallest units of text data are characters, character-based approaches have been introduced into the fields of NMT. Costa-jussà and Fonollosa (2016) have shown that the character-based encoding by using convolutional networks and the highway network as shown in Kim et al. (2016). Chung et al. (2016) applied the character-based decoding to the NMT models, whose encoder is based on the BPE units. Luong and Manning (2016) have proposed a hybrid NMT model flexibly switching from the word-based to the character-based model. Each character-based NMT model shows better performance than the word-based NMT models. All of theses models are, however, applied to sequence-based NMT models, and there are no results of the character-based decoding applied to tree-based NMT models yet.

7 Conclusion

In this paper, we reported our systems (UT-AKY) submitted to the English-to-Japanese translation task in WAT'16. Both of our systems are based on tree-to-sequence ANMT models, and one is a word-based decoder model and the other is a character-based decoder model. We incorporated phrase category labels into the tree-based ANMT model. The experimental results on English-to-Japanese translation shows that the character-based decoder does not outperform the word-based decoder but exhibits two promising properties: 1) It takes much less time to compute the softmax layer; and 2) It can translate any word in a sentence.

Acknowledgements

This work was supported by CREST, JST, and JSPS KAKENHI Grant Number 15J12597.

References

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2015. Neural Machine Translation by Jointly Learning to Align and Translate. In *Proceedings of the 3rd International Conference on Learning Representations*.

KyungHyun Cho, Bart van Merrienboer, Dzmitry Bahdanau, and Yoshua Bengio. 2014a. On the Properties of Neural Machine Translation: Encoder-Decoder Approaches. In *Proceedings of Eighth Workshop on Syntax, Semantics and Structure in Statistical Translation (SSST-8)*.

Kyunghyun Cho, Bart van Merrienboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. 2014b. Learning Phrase Representations using RNN Encoder–Decoder for Statistical Machine Translation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing*, 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*, pages 1693–1703.

R. Marta Costa-jussà and R. José A. Fonollosa. 2016. Character-based neural machine translation. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics*, pages 357–361.

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*, pages 823–833.

Felix A. Gers, Jürgen Schmidhuber, and Fred A. Cummins. 2000. Learning to Forget: Continual Prediction with LSTM. *Neural Computation*, 12(10):2451–2471.

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long Short-Term Memory. *Neural Computation*, 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*, pages 944–952.

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.

Shihao Ji, S. V. N. Vishwanathan, Nadathur Satish, Michael J. Anderson, and Pradeep Dubey. 2016. BlackOut: Speeding up Recurrent Neural Network Language Models With Very Large Vocabularies. In *Proceedings of the 4th International Conference on Learning Representations*.

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*, volume 37, pages 2342–2350.

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.

Yoon Kim, Yacine Jernite, David Sontag, and Alexander M. Rush. 2016. Character-aware neural language models. In *The Thirtieth AAAI Conference on Artificial Intelligence, AAAI 2016*.

Wang Ling, Phil Blunsom, Edward Grefenstette, Moritz Karl Hermann, Tomáš Kočiský, Fumin Wang, and Andrew Senior. 2016. Latent predictor networks for code generation. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 599–609. Association for Computational Linguistics.

Minh-Thang Luong and D. Christopher Manning. 2016. Achieving open vocabulary neural machine translation with hybrid word-character models. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics*, pages 1054–1063.

Thang Luong, Hieu Pham, and Christopher D. Manning. 2015a. Effective Approaches to Attention-based Neural Machine Translation. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 1412–1421.

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.

Yusuke Miyao and Jun'ichi Tsujii. 2008. Feature Forest Models for Probabilistic HPSG Parsing. *Computational Linguistics*, 34(1):35–80.

Toshiaki Nakazawa, Hideya Mino, Isao Goto, Graham Neubig, Sadao Kurohashi, and Eiichiro Sumita. 2015. Overview of the 2nd Workshop on Asian Translation. In *Proceedings of the 2nd Workshop on Asian Translation (WAT2015)*, pages 1–28.

Toshiaki Nakazawa, Hideya Mino, Chenchen Ding, Isao Goto, Graham Neubig, Sadao Kurohashi, and Eiichiro Sumita. 2016a. Overview of the 3rd workshop on asian translation. In *Proceedings of the 3rd Workshop on Asian Translation (WAT2016)*, Osaka, Japan, December.

Toshiaki Nakazawa, Manabu Yaguchi, Kiyotaka Uchimoto, Masao Utiyama, Eiichiro Sumita, Sadao Kurohashi, and Hitoshi Isahara. 2016b. Aspec: Asian scientific paper excerpt corpus. In *Proceedings of the 10th Conference on International Language Resources and Evaluation (LREC2016)*, Portoroz, Slovenia, 5.

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*, pages 529–533.

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*, pages 311–318.

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

Rico Sennrich and Barry Haddow. 2016. Linguistic input features improve neural machine translation. In *Proceedings of the First Conference on Machine Translation*, 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*, pages 1715–1725.

Ilya Sutskever, Oriol Vinyals, and Quoc V. Le. 2014. Sequence to Sequence Learning with Neural Networks. In *Advances in Neural Information Processing Systems 27*, pages 3104–3112.

Kai Sheng Tai, Richard Socher, and Christopher D. Manning. 2015. Improved Semantic Representations From Tree-Structured Long Short-Term Memory Networks. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing*, pages 1556–1566.

Zhongyuan Zhu. 2015. Evaluating Neural Machine Translation in English-Japanese Task. In *Proceedings of the 2nd Workshop on Asian Translation (WAT2015)*, pages 61–68.