# Improving Neural Machine Translation with Linearized Dependency Tree Decoder

An Nguyen Le[1,a)] Ander Martinez[1,b)] Yuji Matsumoto[1,c)]

**Abstract:** In spite of achieving significant performance in recent years, there are some existing issues that Sequence-to-Sequence Neural Machine Translation still does not solve completely. Two of them are translation for long sentences and the over-translation. To address these two problems, we propose an approach that utilize more syntactic information, syntactic dependency information, so that the output is generated based on more abundant information. In addition, the outputs are presented not as a simple sequence of tokens but as a linear grammatical tree structure. In addition, the output of the model is presented not as a simple sequence of tokens but as a linearized tree construction. Experiments on the Europarl-v7 dataset of French-to-English translation demonstrate that our proposed method can produce dependency relations between words in the target language and improve BLEU scores by 1.57 and 2.40 on datasets consisting of sentences with up to 50 and 80 tokens, respectively. Furthermore, the proposed method also solved the ineffective translation for long sentences and repetition problems in Neural Machine Translation.

## 1. Introduction

In this work, we propose an approach in which syntactic dependencies are incorporated into the model to represent more grammatical information of output. As we know, the Sequence to Sequence (`Seq2Seq`) Learning model [17], [26] is extremely effective on a variety of tasks that require a mapping between a sequence to sequence. Thus, it is used to solve many tasks in natural language processing. The `Seq2Seq` model consists of an encoder-decoder neural network which encodes a variable-length input sequence into a vector and decodes it into a variable-length output. Because the model uses the information of the source representation and the previously generated words to produce the next-word token, this distributed representation allows the `Seq2Seq` model to generate appropriate mapping between the input and the output [22]. In Neural Machine Translation (NMT) task, which is based on the Seq2Seq learning, has achieved excellent translation performance in recent years on [4], [17], [20], [21]. In particular, the NMT model which is built upon an encoder-decoder framework with attention mechanism [20] can also pay attention and its decoder knows which part of the input is relevant for the word that is currently being translated. Therefore, it has shown competitive results and outperformed conventional statistical methods [1]. Despite of these advantages, NMT model still has a couple particular issues to be solved such as dealing with fixed vocabulary, not applicable to small-data, additional phrases, wrong lexical choice errors, long sentence translation and overproduction of strange tokens... In this paper, we touch upon the following two major problems:

- Translation of long sentences
- Over-translation

Since the decoder of the model produces the target language word by word simply based on the previous target words and the source-side representation vector until it reaches the special end token, it is incapable in capturing long-distance dependencies in history, so ineffective for long sentences translation [2], [11]. Even with an attention mechanism, NMT model just pays attention to the current alignment information between the inputs and the output at the current position but ignores past alignments information. Therefore, it cannot keep track of the attention history when it updates information at each current time step, leading to the over-production [12], [23], [24], [25].

In order to address the above two issues, it is worth considering that using syntactic dependency information and representing the output as a tree structure would be effective. This approach allows the next tokens to be output based on not only the previous tokens but also the syntactic dependencies so far, thereby conditioning them on more abundant information so it has the ability to make smarter predictions. Basically, in this paper, we train the model with an encoder-decoder neural network and using dependencies in which the input of the source language is in sequence form and the output of the target language will be generated in a linearized dependency-based tree structure. That is, instead of predicting only words at each time step, the model trains the network to predict both words and their grammatical dependencies as a dependency tree at each time step. Therefore, it is hoped that the accuracy of output will be improved.

The major contributions of this work are as follows:

- To utilize the information of both "head" words and syntactic dependencies between them to produce better output.

[1] Nara Institute of Science and Technology, Ikoma, Nara 630–0192, Japan
[a)] nguyen.an.mr9@is.naist.jp
[b)] ander.martinez.zy4@is.naist.jp
[c)] matsu@is.naist.jp

- To settle the problems in the NMT task. In this paper, we desire to solve two tasks. First is the ineffective translation for long sentences. Second is the over-translation in NMT task.

Empirically, to assess the performance of the proposed method, we used Conditional Gated Recurrent Unit with Attention mechanism model of Bahdanau [20] on the French-English portions of the Europarl-v7 dataset. As a result, the BLEU score is improved by 1.57 and 2.40 points for sentences of length up to 50 and 80 tokens, respectively. Also, we compare and analyze the results of attention-based `Seq2Seq` model and the proposed approach.

## 2. Related Work

In fact, the effectiveness of using dependency information of words has been reported in some previous NLP tasks, for example, in dependency-based word embeddings, relation classification and sentence classification tasks [18], [27], [28], [29], [30]. It has been shown that the combination of words and their dependency information can boost performance. Besides, in the work of Vinyals et al. [10], they also represent output as a linearized tree structure, but their work showed that generic sequence-to-sequence approaches can achieve excellent results on syntactic constituency parsing. At a glance, our proposed method is a little similar to the works of Dyer et al., Aharoni et al., Eriguchi et al., Wu et al. [5], [8], [9], [14] in use of parse tree and generation. However, Dyer et al. and Aharoni et al.'s works concern predicting constituent trees. Eriguchi et al.'s model employs syntactic dependency parsing but their model is hybridized the decoder of NMT and the Recurrent Neural Network Grammars, and the target sentences are parsed in transition-based parsing. Wu et al.'s model also employs dependency parsing but their model separately predicts the target translation sequence and parsing action sequence which maps to translation. On the other hand, our proposed model's decoder directly predicts the linearized dependency tree itself in a single neural network in *depth-first pre-order* order so that the next-word token is generated based on syntactic relations and tree construction itself. In other words, our model is able to learn and produce a tree of words and their dependency relations by itself.

## 3. Related Processing

### 3.1 Stanford Dependency Parser

Stanford Dependency Parser, the popular transition-based parser in natural language processing, produces results in the form of a tree structure in which each word of the sentence is the dependent of exactly one token, either another word in the sentence or the distinguished "ROOT-0" token. The parsing result is represented in the format "*abbreviated relation name(governor, dependent)*" in which a governor is a head word and dependency is a syntactic relation between a governor and a dependent. The governor and the dependent are words in the sentence.

### 3.2 *Depth-First Pre-order* Tree Search

The search tree is deepened as much as possible on each child before going to the next sibling. *Depth-first pre-order* Tree Search starts with the root node and displays the data part of the current node by traversing the left subtree to right subtree recursively calling the pre-order function. The reason we use *Depth-first pre-order* Tree Search algorithm is we want our model to predict the next-word tokens from *left-to-right* so that it can keep the order of words in output sequences the model generates.

## 4. Sequence-to-Dependency Model

In our proposed approach, the neural network model is trained to map the target language output in linearized dependency tree construction from the source language input which is as a sequence. Thus, we can define the problem as follows. Given a source sequence $\boldsymbol{X} = (x_1, x_2, \ldots, x_N)$ of length $N$, we want the model to encode the input sequence $X$ and decode it to a tree structure with both words and dependency information conditioned on the encoded vector. Therefore, the output will be represented in the form $(\boldsymbol{LY}) = (\mathbf{ly}_1, \mathbf{ly}_2, \ldots, \mathbf{ly}_M)$. The conditional probability $p(ly|x)$ is decomposed as:

$$p(\mathbf{ly}|x) = \prod_{i=1}^{\infty} p(\mathbf{ly}_i|\mathbf{ly}_{<i}, x), \tag{1}$$

in which $(\mathbf{ly}_1, \mathbf{ly}_2, ..., \mathbf{ly}_M)$ are words or dependency labels.

Therefore, the hidden state $\mathbf{s}_j$ at time step $j$ is computed as follows:

$$\mathbf{s}_j = \mathrm{cGRU}_{\mathrm{att}}\left(\mathbf{s}_{j-1}, \mathbf{ly}_{j-1}, \mathrm{C}_j\right), \tag{2}$$

and the next token $\mathbf{ly}_j$, which may be a word or dependency label, will be generated as follows:

$$\mathbf{ly}_j = f\left(\mathbf{s}_j, \mathbf{ly}_{j-1}, \mathrm{C}_j\right), \tag{3}$$

In this paper, dependencies are defined as the dependency labels which are achieved from the Stanford Dependency Parser [31]. The decoder will decode the next output based on relations between governors and dependents in a linearized tree structure. In regards to the order of generating the dependency labels and the words, the decoder will produce these symbols in a manner called *depth-first pre-order* traversal. In this section, we will describe the model step-by-step as follows:

### 4.1 Dependency Parsing

Since there is not a parallel corpus in which target-side is represented in linearized dependency tree, we have to create the parallel corpus for training dataset by doing dependency parsing for the target-side language. In this paper, we do experiments on a French-English language pair so we use the Stanford Dependency Parser to obtain dependency parsing results for English. After that, this dependency parsing results will be transformed in another step for traversing the tree, which will be described in the next section to create a dependency tree. The dependency tree represents the target language as an ordered tree structure which is necessary for training. The reason we chose the Stanford Dependency Parser for the parsing portion of this method is because it can represent the order of words in sentence. This information of the order is useful to traverse tree in the following step.

### 4.2 Transformation and Tree Traversal

In this section, we describe the Tree Transform and Tree

**Algorithm 1** Tree Transform

```
1:  procedure TRANSFORM TREE
2:  Transform(T,Labels):
3:      for label in Labels do
4:          if label.children.size! = 0 then
5:              Recur Transform(T,Labels)
6:          else
7:              Compare the order of current label's parent & children
8:              if (label's children order is larger than label's parent order then
9:                  INSERT label's parent first
10:             else
11:                 INSERT label's children
```

Traversal process in which output in a linearized dependency tree form is created from the Stanford Dependency Parsing tree. For example, given a sentence "*I ate the fish with a fork.*", after obtaining dependency parsing tree from the above dependency parsing phase, we move the "*head*" word to the same layer with *dependents* which are directly connected to the "*head*" word so that these dependents and "*head*" word are rooted to {*ROOT*. The "*head*" word is put in a position so that the word order in sentence can be preserved. Next, the tree structure obtained in the fist step will be transformed into another tree structure for the next tree traversal step. Next we traverse this tree in a *depth-first pre-order* traversal to create output with a linearized tree structure to train the model. That is, for each rooted subtree, governors and dependency labels of the sentence are predicted first, and their information will be used to predict the next dependent words. In other words, the model can capture the dependency information between label-word and word-word pairs to predict the next tokens. This means that the model is capable of modeling grammatical dependencies in the output symbols. Also, in `Seq2Dep` model, we define the *Nonterminal* "{*DEPENDENCY LABEL*", and *Node-closing* "}" tokens. *Nonterminal* indicates subtree [15], which means open subtree to visit children node. *Node-closing* indicates end-of-subtree, that means the subtree is finished and return to the upper layer. And these defined tokens donot appear in original source and target datasets. Algorithms 1 and 2 show the definition of transformation and tree traversal in more detail respectively. The purpose of using *depth-first pre-order* traversal is as follows:

- To keep the words of the target language sequence in order when they are generated. With this generating order, the word order of the sentence is preserved, thus, we do not have to do any post-processing subsequently.
- To utilize both information of the words and the dependency labels generated in the previous rooted subtree to predict the tokens of the next rooted subtree.

Figures 1, 2 and 3 show the Stanford dependency parsing tree, tree after changing "*head*" word and Transform Tree.

### 4.3 Sequence-to-Dependency Model

The proposed Sequence-to-Dependency (Seq2Dep) model consists of an encoder which is a bidirectional GRU layer as in Bahdanau's model [20][*1]. The input embeddings of the source sentences are shared by the forward and backward GRU, and the hidden states of the corresponding forward and backward GRU are added to obtain the hidden representation for that time step. The decoder of the model will decode the output as words and dependency labels in a linearized dependency tree structure in a *depth-first pre-order* traversal. Figure 4 shows the decoder which generates both dependency labels and words in the `Seq2Dep` model. In Figure 4, the previous token and context vector feeding are omitted for simplicity.

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

**Algorithm 2** Tree *depth-first pre-order* traversal

```
Input: Sentence
Output: Linearized Dependency Tree
1:  Stanford Dependency Parsing
2:  Make Tree from Dependency Parsing Result
3:  Tree transform
4:  procedure TRAVERSE TREE
5:  Traverse(T,N):
6:      N as discovered
7:      for all Node not in N do
8:          if Node.children.size! = 0 then
9:              Recursively call Traverse(T,N)
10:                 in pre-order traverse
11:         else
12:             if Node is Nonterminal then
13:                 OUTPUT Node-opening
14:                 VISIT children
15:                 OUTPUT Node-closing
16:             else
17:                 OUTPUT Node
```

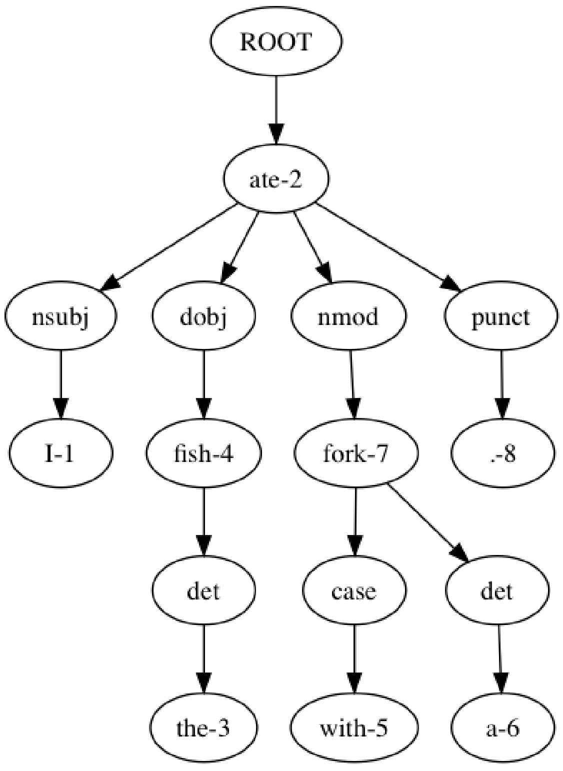

**Fig. 1** Stanford Dependency Parsing Tree

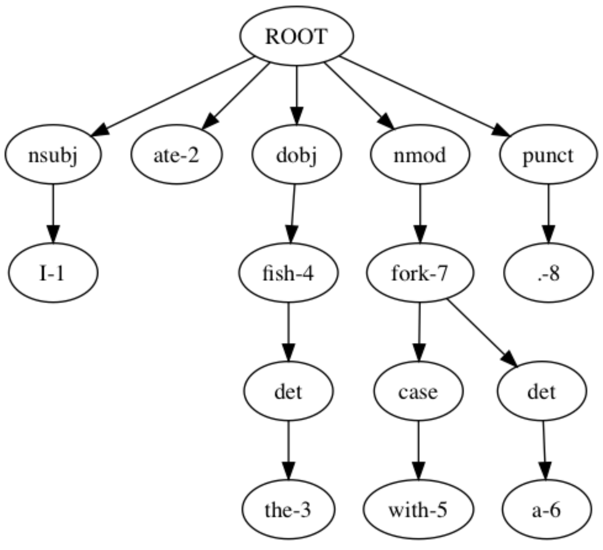

**Fig. 2** Dependency tree after changing position of "*head*" word

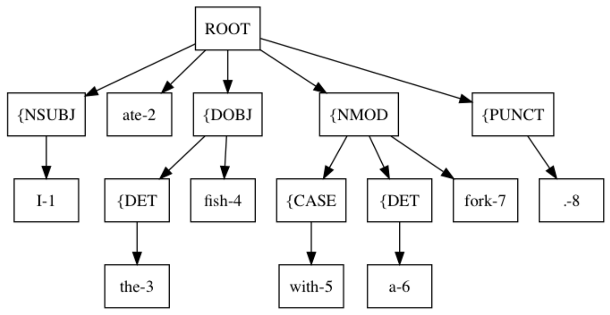

**Fig. 3** Transform Tree

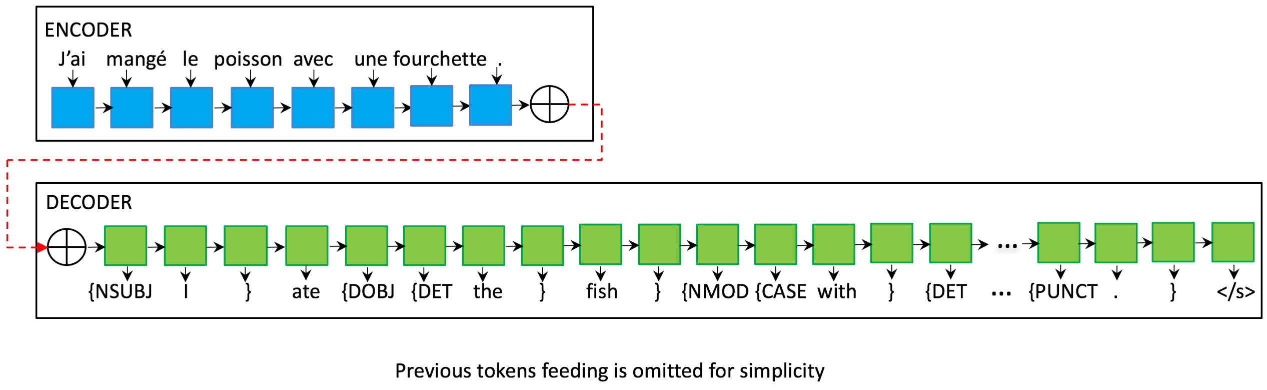

**Fig. 4** Seq2Dep model

## 5. Experiments

### 5.1 Dataset

In our experiment, the proposed model was trained on the French-English parallel corpus of the *Europarl-v7* dataset. We used *newstest2011* and *newstest2012* of WMT16 as development and test data respectively. To confirm translation for long sentences, the whole test set was used without removing any sentences with a maximum length of 50 or 80. We performed experiment on the following two datasets:

- Europarl-v7 dataset consisting of sentences with a maximum length of 50.
- Europarl-v7 dataset consisting of sentences with a maximum length of 80.

For data preprocessing, we filtered out sentences which were longer than the above maximum lengths and cleaned the special symbols or characters which were not strings. We also omitted sentences which had multiple sentences in one line. The reason is that the parsing results obtained from the Stanford Dependency Parser in parsing step would contain multi "{ROOT" tokens for sentences which have multiple sentences in one line, while it is necessary to generate the next child nodes starting from just one top {*ROOT* of a tree. Next, we tokenize and lowercase this dataset and perform dependency parsing. After that, we traverse the tree in a *depth-first pre-order* to create the parallel corpus for the training model in which the source language, French is in sequence form, and the target language, English is in a linearized dependency tree structure form. The longer sentences are(in particular, sentences with a maximum length of 80), the more CPU memory and time cost for this processing data step.

In addition, we built a dictionary of the target language (English) that consists of both words and dependency labels. In this dictionary, we define 74 dependency labels based on the current representation of grammatical relations of the Stanford Dependency Parser.

### 5.2 Settings

In order to evaluate the performance of the proposed method, we set the same hyperparameters as the attention-based cGRU model in DL4MT-Tutorial and compare the obtained results of both `Seq2Seq` and `Seq2Dep` models.

The recurrent transformation weights for gates and hidden state proposal matrices were initialized as random orthogonal matrices. Weights were optimized using the Adadelta algorithm and were updated with a mini-batch size of 32 sentences. The vocabulary sizes of both source and target languages were set at 30k

**Table 1** Translation quality as measured by different metrics.

| Model | Post-processing | | |
|---|---|---|---|
| | **BLEU** | **METEOR** | **TER** |
| Seq2Seq-50 | 19.31 | 26.3 | 66.1 |
| Seq2Dep-50 | **20.88** | **27.0** | **62.5** |
| Seq2Seq-80 | 16.97 | 25.5 | 78.5 |
| Seq2Dep-80 | **19.37** | **25.6** | **65.6** |

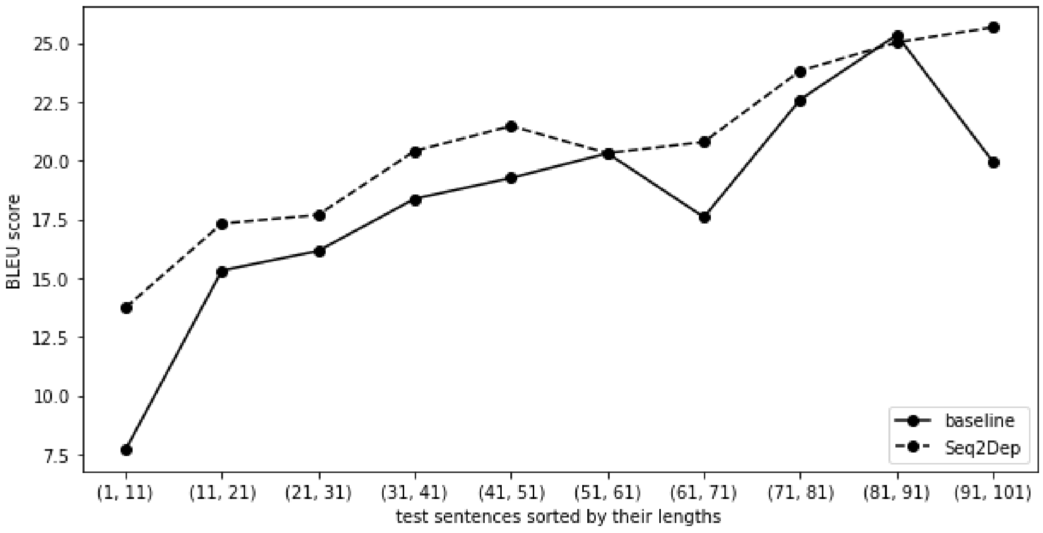

**Fig. 5** Comparison of BLEU score with respect to the length of sentences

words, the beam size was set to 5, dropout was not applied and the gradients were clipped at 1.0. Morever, because the generated tokens are not only words but also dependency labels in `Seq2Dep` model, the maxlen parameter was set up so that dependency labels are not counted, therefore long sentences will not be removed in training.

### 5.3 Model Training

In the experiments, we trained the following 2 models on the 1.65M sentences with a maximum length of 50 and 1.89M sentences with a maximum length of 80 from the Europarl-v7 French-English bitext.

**Attention-based Seq2Seq Model**
This model is a `Seq2Seq` model with attention mechanism as in Firat [4] that consists of an encoder that encodes the source language input in sequence form and a decoder that decodes target language output in sequence form.

**Seq2Dep Model** The proposed method. In this model, the model architecture is the same as the attention-based Seq2Seq model but the input is in sequence form and the output is in linearized dependency tree structure.

## 6. Results

In the `Seq2Dep` model, because the output consists of both words and dependency labels, we evaluated the result with post-processing, which is the process that removes the dependency labels from the translated result. From this section onwards, we will refer to the `Seq2Seq` and `Seq2Dep` models with sentences of maximum length 50 and 80 tokens as `Seq2Seq-50`, `Seq2Dep-50`, `Seq2Seq-80` and `Seq2Dep-80`. As a result, the BLEU score of `Seq2Dep-50` with post-processing was 20.88, which is higher than the BLEU score of 19.31 obtained by the attention-based `Seq2Seq-50` model with a gain of up to 1.57 points. Similarly, the BLEU score improved by 2.40 points for datasets with maximum sentence lengths of 80. Table 1 shows BLEU and METEOR scores and TER error of the attention-based `Seq2Seq` and `Seq2Dep` models. Figure 5 shows the relation between BLEU score and the length of sentence.

Moreover, when we made a trial to evaluate the translation results without post-processing, the BLEU scores without post-processing were 42.76 and 43.41 for both datasets. From these scores, it is thought that the model can predict not only word-based tokens but also dependency labels well.

## 7. Testing Repetition

In order to verify the ability of the proposed approach to solve the repetition problem of NMT, over-translation, we measured the repetition of words in the translation results of attention-based `Seq2Seq` and `Seq2Dep` learnings in this section. The repetition rate is measured by the following formula:

$$rep\_rat = \sum_{i=1}^{T(y)} \frac{1 + r(\widetilde{y_i})}{1 + r(Y)}, \qquad (4)$$

in which $\widetilde{y_i}$ and $Y_i$ are the $i^{th}$ hypothesis sentence and $i^{th}$ reference sentence respectively, and $r$ is the number of the repeated words and is computed by:

$$r(X) = len(X) - len(set(X)) \qquad (5)$$

in which len(X) is the length of the sentence X and len(set(X)) is the number of words that are not repeated in sentence X. For example, given the sentence X="*The big fish ate the smaller*

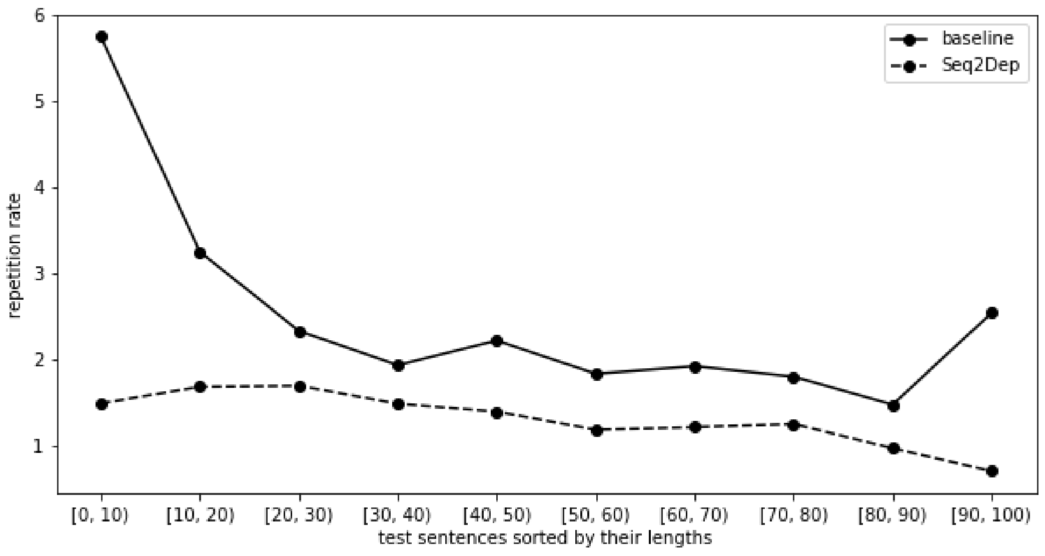

**Fig. 6** Comparison of the repetition rate of the baseline and Seq2Dep models

*fish*", in this case, set(X)={The, big, fish, ate, smaller}, len(X)=7, len(set(X))=5. Figure 6 shows the comparison of repetition rate in both models in which the horizontal axis is the length of sentences, vertical axis is the repetition rate respectively. In Figure 6, the repetition rate in both Seq2Seq and Seq2Dep learnings decreases as the length of the sentences increases. From Figure 6, we can see that the more tokens the model learns, the more the repetition rate decreases. Also, the repetition is reduced in the Seq2Dep model compared to the attention-based Seq2Seq model.

## 8. Analysis and Discussion

In figure 5, except the span in which the sentence length is between 41 and 51 words, the BLEU score of the Seq2Dep model goes up gradually and almost overcomes that of the attention-based Seq2Seq model. The BLEU score falls from 19.31 to 16.97 with a 2.34 points difference for the attention-based Seq2Seq model while the point difference is 1.51 in the Seq2Dep model. From the experiments, we confirm that by using the syntactic dependency information, the Seq2Dep model can learn well and reduce the drop in BLEU score compared to the baseline model even if the sentence is very long. Besides, we can see the BLEU score is low for short sentences which have a length of 10 words or less. This is because of the brevity penalty on short sentences in BLEU [13].

With regards to the BLEU score without post-processing, we see that the score of the Seq2Dep-80 model is higher than that of the Seq2Dep-50 model. The reason could be: The longer the sentences are, the more syntactic dependencies the models require for generating better outputs.

Also, in terms of the over-translation problem, Figure 6 shows that the repetition rates of the two models decrease gradually with respect to the length of the sentences and the Seq2Dep model has a lower repetition rate. When we checked the translation results, we saw that *Node-closing* token "}" was almost generated after each subtree . Moreover, we saw that there were some very long sentences which the over-generation of "*UNK*"s occurred in the translation result of Seq2Seq model while that did not occur in translation results of Seq2Deq model. Our assumption is that after generating subtree, the Seq2Dep model can learn that it should generate the *Node-closing* token "}" next, instead of a chain of words. In other words, as mentioned in Kuncoro et al.'s work [3] in which modeling of composition can achieve better performance, the Seq2Dep model which learns about the syntactic dependencies and tree structure performance is probably able to learn the blocks of the form "*Nonterminal* word }" like a phrase-structure in sentences, so it is unlikely to generate the same word repeatedly. Therefore, it is possible to prevent the long repeated words in long sentences. Usually, because the block of the form "*Nonterminal* word }" is seen as a phrase in sentence or a subtree in tree structure, and it is rare for a phrase to occur repeatedly in sentence or for a subtree to repeat in a tree structure, so it is assumed that repetition of the blocks of form "*Nonterminal* word }" are also rare.

## 9. Conclusion

In this work, we proposed a method in which the Seq2Dep NMT model is trained by utilizing syntactic dependencies to provide the model more abundant information. In other words, Seq2Dep model learns the potential internal relative connections among tokens and their long term syntactic dependencies to predict the next-word tokens. Furthermore, the Seq2Dep model can also generate output as a linearized dependency tree structure in a *Depth-first pre-order* tree traversal over words and dependencies. The purpose of this work is to alleviate issues of translating long sentences and repetitive translation. We conduct experiments on the French-English parallel corpus of the Europarl-v7 dataset and the results demonstrated the performance of the proposed model which improved BLEU scores by 1.57 and 2.40 points for sentences of length at most 50 and 80 tokens respectively. The translation results, especially on long sentences, were better than the Seq2Sep model with attention mechanism.

## 10. Future work

- Confirm how accurate the Seq2Dep model generates the dependency labels as well.
- In this paper, we set the same hyperparameters as the attention-based cGRU model in DL4MT-Tutorial and trained the Seq2Dep model on only Europarl-v7 dataset. Since experiments were done on small vocabulary size and dataset, we plan to train the model on larger vocabulary and datasets with subword units segmentation.
- For future work, we plan to train models on datasets which consist of only long sentences with more than 50 or 80 tokens to compare the performance of long-sentences translation of the approach and baseline model.

## Acknowledgments

We thank Assistant Professor Shindo Hiroyuki, Ouchi Hiroki, Michael Wentao Li of the NAIST Computational Linguistics Laboratory for their valuable comments.

### References

[1] Luisa, B., Arianna, B., Mauro, C. and Marcello, F.: Neural versus Phrase-Based Machine Translation Quality: a Case Study, *CoRR*, arXiv.org1608.04631 (2016).

[2] Antonio, T. and Victor, M.: A Multifaceted Evaluation of Neural versus Phrase-Based Machine Translation for 9 Language Directions, *CoRR*, arXiv.org1701.02901 (2017).

[3] Adhiguna, K., Miguel, B., Lingpeng, K., Chris, D., Graham N. and Noah, A.S.: What Do Recurrent Neural Network Grammars Learn About Syntax?, *CoRR*, arXiv.org1611.05774 (2016).

[4] Orhan, F. Kyunghyun, C. and Yoshua, B.: Multi-Way, Multilingual Neural Machine Translation with a Shared Attention Mechanism, *CoRR*, arXiv.org1601.01073 (2016).

[5] Roee Aharoni and Yoav Goldberg: Towards String-to-Tree Neural Machine Translation, *CoRR*, abs/1704.04743 (2017).

[6] Rico, S., Barry, H., and Alexandra, B.: Neural Machine Translation of Rare Words with Subword Units, *CoRR*, abs/1508.07909 (2015).

[7] Sébastien, J., Kyunghyun, C., Roland, M. and Yoshua, B.: On Using Very Large Target Vocabulary for Neural Machine Translation, *CoRR*, abs/1412.2007 (2014).

[8] Chris, D., Adhiguna, K., Miguel, B., and Noah, A. S.: Recurrent Neural Network Grammars, *CoRR*, abs/1602.07776 (2016).

[9] Akiko, E., Yoshimasa, T., and Kyunghyun, C.: Learning to Parse and Translate Improves Neural Machine Translation, *CoRR*, abs/1702.03525 (2017).

[10] Oriol, V., Lukasz, K., Terry, K., Slav, P., Ilya, S., and Geoffrey, E. H.: Grammar as a Foreign Language, *CoRR*, abs/1412.7449 (2014).

[11] Biao, Z., Deyi, X., and Jinsong, S.: Recurrent Neural Machine Translation, *CoRR*, abs/1607.08725 (2016).

[12] Zhaopeng, T., Zhengdong, L., Yang, L., Xiaohua, L., and Hang, L.: Coverage-based Neural Machine Translation, *CoRR*, abs/1601.04811 (2016).

[13] Kishore, P., Salim, R., Todd, W., and Wei-Jing, Z.: BLEU: a Method for Automatic Evaluation of Machine Translation, *Proceedings of ACL 2002*, pp. 311–318, Philadelphia, America (2002).

[14] Shuangzhi, W., Dongdong, Z., Nan, Y., Mu, L., and Ming, Z.: Sequence-to-Dependency Neural Machine Translation, *Proceedings of ACL 2017*, pp. 698–707, Vancouver, Canada (2017).

[15] Li, D., and Mirella, L.: Language to Logical Form with Neural Attention, *Proceedings of ACL 2016*, pp. 33–43, Berlin, Germany (2016).

[16] Xingxing, Z., Liang, L. and Mirella, L.: Top-down Tree Long Short-Term Memory Networks, *Proceedings of NAACL-HLT 2016*, vol. abs/1511.00060, pp. 310–320, San Diego, California, USA (2016).

[17] Ilya, S., Oriol, V. and Quoc, V. L.: Sequence to Sequence Learning with Neural Networks, *Proceedings of NIPS 2014*, Montreal, Canada (2014).

[18] Kazuki, O. and Kenji, H.: Dependency Parsing and Its Application using Hierarchical Structure in Japanese Language, *International Journal on Advances in Internet Technology*, vol. 7 no 3, 4 (2014).

[19] Jacob, A., Andreas, V. and Stephen, C.: Semantic Parsing as Machine Translation, *Proceedings of the 51st ACL*, pp. 47–52, Sofia, Bulgaria (2013).

[20] Dzmitry, B., Kyunghyun, C. and Yoshua, B.: Neural Machine Translation by Jointly Learning to Align and Translate, *Proceedings of ICLR 2015*, San Diego, California, USA (2015).

[21] Thang, L., Hieu P. and Christopher, D. M.: Effective Approaches to Attention-based Neural Machine Translation, *Proceedings of EMNLP 2015*, pp. 1412–1421, Lisbon, Portugal (2015).

[22] Xiaoqing, L., Jiajun, Z. and Chengqing, Z.: Towards Zero Unknown Word in Neural Machine Translation, *Proceedings of IJCAI 2016*, New York, USA (2015).

[23] Haitao, M., Baskaran, S., Zhiguo, W. and Abe, I.: Coverage Embedding Models for Neural Machine Translation, *Proceedings of EMNLP 2016*, pp. 955–960, Austin, Texas, USA (2016).

[24] Zhaopeng, T., Yang, L., Lifeng, S., Xiaohua, L. and Hang, L.: Neural Machine Translation with Reconstruction, *Proceedings of AAAI 2016*, Phoenix, Arizona, USA (2016).

[25] Zhaopeng, T., Zhengdong, L., Yang, L., Xiaohua, L. and Hang, L.: Modeling Coverage for Neural Machine Translation, *Proceedings of ACL 2016*, pp. 76–85, Beijing, China (2016).

[26] Roee, A., Yoav, G. and Yonatan, B.: Improving Sequence to Sequence Learning for Morphological Inflection Generation: The BIU-MIT Systems for the SIGMORPHON 2016 Shared Task for Morphological Reinflection, *Proceedings of the 14th Annual SIGMORPHON Workshop on Computational Research in Phonetics, Phonology, and Morphology*, pp. 41–48, Berlin, Germany (2016).

[27] Omer, L. and Yoav, G.: Dependency-Based Word Embeddings, *Proceedings of ACL 2014*, pp. 302–308, Baltimore, Maryland, USA (2014).

[28] Alexandros K.: Dependency Based Embeddings for Sentence Classification Tasks, *Proceedings of NAACL-HLT 2016*, pp. 1490–1500, San Diego, California, USA (2014).

[29] Yang, L., Furu, W., Sujian, L., Heng, J., Ming, Z. and Houfeng, W.: A Dependency-based Neural Network for Relation Classification, *Proceedings of ACL 2015*, pp. 285–290, Beijing, China (2015).

[30] Richard, S., Andrej, K., Quoc V. L., Christopher, D. M. and Andrew Y. N.: Grounded Compositional Semantics for Finding and Describing Images with Sentences, *Transactions of the Association for Computational Linguistics"*, pp. 2: 207–218 (2015).

[31] Danqi, C. and Christoher, D. M.: A Fast and Accurate Dependency Parser using Neural Networks, *Proceedings of EMNLP 2014*, Doha, Qatar (2015).