TẠP CHÍ KHOA HỌC
TRƯỜNG ĐẠI HỌC SƯ PHẠM TP HỒ CHÍ MINH
Tập 19, Số 10 (2022): 1725-1734
Website: https://journal.hcmue.edu.vn
ISSN: 2734-9918

HO CHI MINH CITY UNIVERSITY OF EDUCATION
JOURNAL OF SCIENCE
Vol. 19, No. 10 (2022): 1725-1734
https://doi.org/10.54607/hcmue.js.19.10.3617(2022)

**Research Article**

# SYNTAX-ENHANCED NEURAL MACHINE TRANSLATION WITH GRAPH ENCODER

***Nguyen Hong Buu Long*, Pham Hong Viet***
*University of Science, Vietnam National University Ho Chi Minh City, Vietnam*
*Corresponding Author: Nguyen Hong Buu Long – Email: nhblong@fit.hcmus.edu.vn*
*Received: October 11, 2022; Revised: October 25, 2022; Accepted: October 26, 2022*

**ABSTRACT**

*Neural Machine Translation (NMT) is a new paradigm in machine translation (MT) powered by recent advances in sequence to sequence learning frameworks. With the advance of Neural Networks, NMT has become the most promising MT approach in recent years. Despite the apparent success, NMT still suffers from one significant drawback in integrating syntactic knowledge into neural networks. This paper proposes an extension of the NMT model to incorporate additional syntactic information from constituency trees. We represent the constituency trees under graph forms encoded by a graph encoder to enhance the attention layer, which allows the decoder to focus on both sequential and graph representation at each decoding step. The experiments show promising results of the proposed method on English-Vietnamese datasets, proving the effectiveness of our syntax-enhanced NMT method.*

***Keywords:*** constituent tree; graph neural networks; neural machine translation; syntax

## 1. Introduction

Neural Machine Translation (NMT) has emerged as the most promising machine translation approach in recent years, showing great progress with the state-of-the-art result. Despite promising, NMT still lacks the ability to model deeper syntactic aspects of languages. Motivated by the success of adding syntactic information to Statistical Machine Translation (SMT) (Koehn & Hoang, 2007; Wang et al., 2014), we explore these syntactic aspects in NMT models.

Recent works have established that explicitly leveraging syntactic information can improve NMT quality. Li et al. (2017) linearized a parse tree into a structural label sequence and let the model automatically learn syntactic knowledge through it. Eriguchi et al. (2016) proposed an approach that focuses on the phrase structure of the input sentence to extend attentional NMT models. Chen et al. (2017) investigated NMT, using explicit source-side syntactic trees, by proposing a syntax-aware encoder-decoder model. Nˇadejde et al. (2017)

***Cite this article as:*** Nguyen Hong Buu Long, & Pham Hong Viet (2022). Syntax-enhanced neural machine translation with graph encoder. *Ho Chi Minh City University of Education Journal of Science, 19*(10), 1725-1734.

introduced a method for modeling explicit target syntax by interleaving target words with their corresponding CCG supertags in NMT systems. However, the above approaches must normalize the tree structure of syntactic trees to sequential representations, which usually lose dependencies between nodes.

In this paper, we investigate utilizing syntactic information from constituency trees under a graph perspective in the context of the NMT framework. Specifically, we employed a graph encoder to encode the constituency trees as graph vector representations. Second, we enhance the normal attention layer to adapt with a new graph attention layer (i.e., dual attention), which decides where the decoder should focus at each decoding step.

This method shows several advantages. Firstly, it could be a way to model extra syntactic knowledge to NMT systems. Then, it could solve the dependency problem between nodes in the constituency trees. Finally, dual attention could help NMT models focusing on sequential and tree representations at each decoding step.

## 2. Methodology

We first present a brief introduction to NMT background. After that, we focus on our graph encoder (illustrated in Figure 1), which is the means to encode constituency trees to continuous representations and dual attention decoder, which is the place that we integrate the graph representations to each decoding step.

### *2.1. The NMT background*

We use a standard encoder-decoder attention-based model (Bahdanau et al., 2015) in which the encoder takes the source sentence **x** as its input and computes a representation for each word $w_i$ in **x**. The decoder calculates the target sentence **y** based on the representation of the source sentence produced by the encoder.

Traditionally, the encoder and decoder are parametrized by a Recurrent Neural Network. Recently, Convolutional Neural Networks (Gehring et al., 2017), and Transformers (Vaswani et al., 2017) with parallel computation and self-attention mechanisms have also achieved competitive results in NMT. In this paper, we investigated RNN and Lightweight Convolution (Wu et al., 2019) (a variant of CNN).

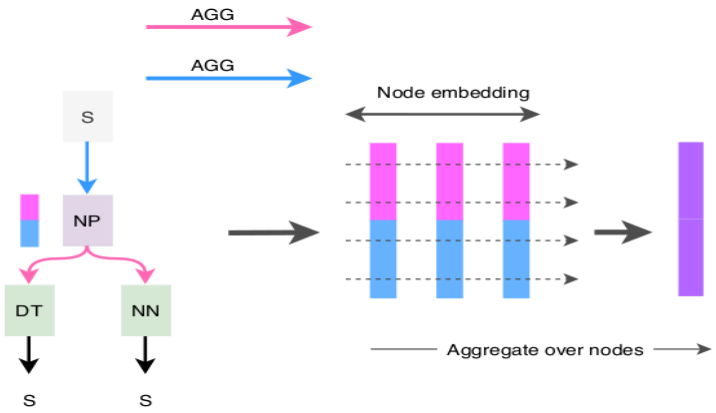


***Figure 1.*** *Graph encoder architecture*

RNNs compute the hidden representation $h_t$ of a word $x_t$ based on the previously hidden state $h_{1:t-1}$ or its left context. Bidirectional RNNs comprise two RNNs: one runs in the forward direction, and another runs in the backward direction. In other words, the forward RNN presents the left context of the word $x_t$,and the backward RNN presents the right context of the word $x_t$. The final representation is their concatenation:

$$h_t = \left[h_t^{fw}, h_t^{bw}\right] \tag{1}$$

Once the source sentence has been encoded, the target sentence **y** is produced word by word using an RNN decoder which computes the probability of the next word $y_t$ given a context vector $c_t$ and previously hidden states of the RNN decoder. The context vector $c_t$ is calculated using an attention mechanism (Bahdanau et al., 2015).

Similar to BiRNN encoder-decoder architecture, we used Lightweight Convolution (Wu et al., 2019) for both the encoder and decoder. Depthwise Convolution (DConv): perform a convolution operation independently over every channel, thereby reducing the number of parameters significantly from $d^2k$ to $dk$ with $k$ as the kernel width and $d$ as the dimension of word embedding. In general, at the position $i$ and direction $c$, the output $O_{i,c}$ is calculated as follows:

$$O_{i,c} = \sum_{j=1}^{k} W_{c,j} \cdot X_{(x+j-\left[\frac{k+1}{2}\right]),c} \tag{2}$$

where $X \in \mathcal{R}^{n \times d}$ is the representation of the sentence.

### *2.2. Syntactic Graph Encoder*

Given a constituency tree $\boldsymbol{T}$, $\boldsymbol{\mathcal{V}}$ is the node set of the tree. We first describe the node embedding algorithm adopted from (Xu et al., 2018):

1. We first transformed the text attribute of node $v$ into a feature vector, $\boldsymbol{a}_v$ by looking up the embedding matrix $W_E$.

2. Next, we categorized the neighbors of $v$ into two subsets: forward neighbors, $\mathcal{N}_{\vdash}(v)$ and backward neighbors, $\mathcal{N}_{\dashv}(v)$. Particularly, $\mathcal{N}_{\vdash}(v)$ returns the nodes that $v$ directs to and vice versa.

3. We aggregated the forward information of $v$'s forward neighbors $\{\boldsymbol{h}_{u\vdash}^{k-1}, \forall u \in \mathcal{N}_{\vdash}(v)\}$ with the current node feature $h_{v\vdash}^{k-1}$, where $k \in \{1, \ldots, K\}$ is the iteration index. We did this by using one of three **AGG**$\vdash$ mentioned below:

$$\boldsymbol{h}_{v\vdash}^{k} = AGG^{\vdash}(h_{v\vdash}^{k-1}, h_{u\vdash}^{k-1}, \forall u \in \mathcal{N}_{\vdash}(v)) \tag{3}$$

4. Similar for backward information:

$$\boldsymbol{h}_{v\dashv}^{k} = AGG^{\dashv}(h_{v\dashv}^{k-1}, h_{u\dashv}^{k-1}, \forall u \in \mathcal{N}_{\dashv}(v)) \tag{4}$$

5. Repeat steps (3)~(5) $K$ times. The concatenation of the final forward and backward

representation is used as the final bi-directional representation of $v$.

$$z_v = CONCAT(\boldsymbol{h}_{v\vdash}^{K}, \boldsymbol{h}_{v\dashv}^{K}), \forall v \in \boldsymbol{\mathcal{V}} \quad (5)$$

In steps (3) and (5), we aggregated v's representation by using one of these aggregator architectures:

**Mean aggregator**: This aggregator function takes the element-wise mean of the vectors in $\{\boldsymbol{h}_{u\vdash}^{k-1}, \forall u \in \mathcal{N}_{\vdash}(v)\}$ and $\{\boldsymbol{h}_{u\dashv}^{k-1}, \forall u \in \mathcal{N}_{\dashv}(v)\}$.

**GCN aggregator**: Similar to Mean aggregator, but followed by a feed-forward layer and a non-linearity activation function:

$$AGG_k^{\vdash} = \sigma(\boldsymbol{W}MEAN(\boldsymbol{h}_{u\vdash}^{k}) + \boldsymbol{b}), \forall u \in \mathcal{N}_{\vdash}(v) \quad (6)$$

$$AGG_k^{\dashv} = \sigma(\boldsymbol{W}MEAN(\boldsymbol{h}_{u\dashv}^{k}) + \boldsymbol{b}), \forall u \in \mathcal{N}_{\dashv}(v) \quad (7)$$

where $MEAN$ denotes the element-wise average operator, and $\sigma$ is a nonlinear activation function.

**Pooling aggregator**: In this aggregator, each neighbor's vector is fed through a fully-connected neural network, and an element-wise max-pooling operation is applied:

$$AGG_k^{\vdash} = max\left(\left\{\sigma\left(\boldsymbol{W}_p\boldsymbol{h}_{u\vdash}^{k} + \boldsymbol{b}\right), \forall u \in \mathcal{N}_{\vdash}(v)\right\}\right) \quad (8)$$

$$AGG_k^{\dashv} = max\left(\left\{\sigma\left(\boldsymbol{W}_p\boldsymbol{h}_{u\dashv}^{k} + \boldsymbol{b}\right), \forall u \in \mathcal{N}_{\dashv}(v)\right\}\right) \quad (9)$$

where $max$ denotes the element-wise max operator, and $\sigma$ is a nonlinear activation function.

The graph embedding vector, $Z$ which contains information in the entire graph, is computed as follows:

$$Z = AGG(z_v, \forall v \in \boldsymbol{\mathcal{V}}) \quad (10)$$

where $AGG$ denotes the aggregator.

### *2.3. Dual Attention Decoder*

We found that using an additional attention mechanism not only helped the model learn the alignment between the source sentence and constituency tree automatically but also addressed the bottleneck problem when adopting only a single attention mechanism because a large amount of information when combining features from the source sentence and the corresponding constituency tree that could not effectively leverage with just one attention mechanism. Our decoder architecture is illustrated in Figure 2.

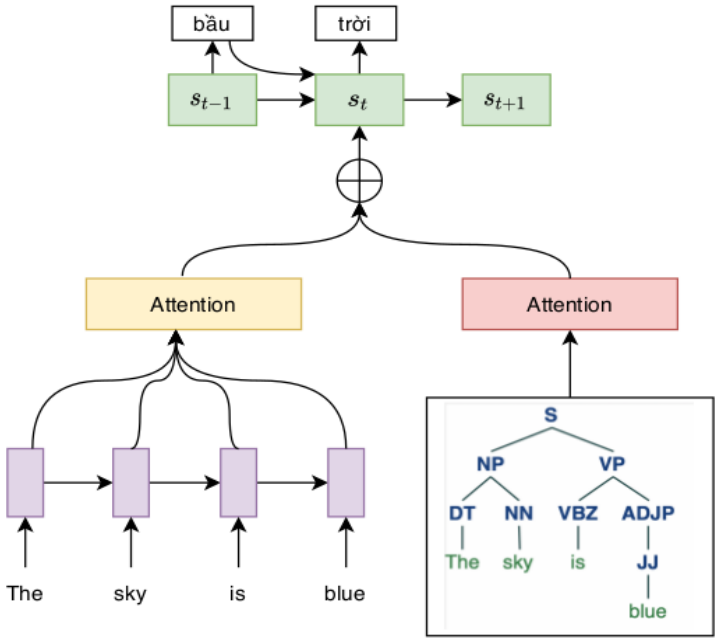


***Figure 2.*** *Decoder architecture*

The graph attention takes graph hidden $h_j$ and the decoder state $s_{i-1}$ as its input. The context vector $\hat{c}_i$ is computed as:

$$\hat{e}_{ij} = a(s_{i-1}, h_j) \tag{11}$$

$$\hat{a}_{ij} = \frac{exp(\hat{e}_{ij})}{\sum_{k=1}^{v} exp(\hat{e}_{ik})} \tag{12}$$

$$\hat{c}_i = \sum_{k=1}^{v} \hat{a}_{ij} h_j \tag{13}$$

where $a$ is the alignment model, which is a feed-forward network, scores how well the inputs surround position $j$ and the input at position $i$ match. Then, a probability distribution over the target vocabulary to produce the output:

$$P_{vocab} = softmax(W_o[s_i, c_i, \hat{c}_i] + b_o) \tag{14}$$

where $W_o$ and $b_o$ are model parameters, $[s_i, c_i, \hat{c}_i]$ demotes a concatenation operation between $s_i, c_i, \hat{c}_i$.

## 3. Experiments

We provided our dataset information, configuration settings as well as our experimental results and analyses.

### *3.1.* Datasets

We used the IWSLT 2015 English-Vietnamese dataset (Cettolo et al., 2015), which contains around 130 thousand sentence pairs for training, and used tst2012 for tuning model parameters and early stopping. We evaluate the official test sets tst2013 and tst2015, which are also included in the IWSLT dataset.

***Table 1.*** *Statistics of the English-Vietnamese datasets*

| Dataset | #tokens | | #types | | #sents |
|---|---|---|---|---|---|
| | en | vi | en | vi | |
| train | 2,435,771 | 2,867,788 | 44,573 | 21,611 | 131,263 |
| dev (tst2012) | 27,988 | 34,298 | 3,518 | 2,170 | 1,553 |
| test (tst2013) | 26,729 | 33,683 | 3,676 | 2,332 | 1,268 |
| test (tst2015) | 20,850 | 26,235 | 3,127 | 2,059 | 1,080 |

For the preprocessing phase, we used byte-pair encoding (BPE) (Sennrich et al., 2016) with 8000 merge operations to deal with rare and compound words and apply them to both English and Vietnamese sides. We measured the end translation quality with case-insensitive BLEU (Papineni et al., 2002). We also applied the bootstrap re-sampling method (Koehn, 2004) to measure the statistical significance ($p < 0.05$) of BLEU score differences between the translation outputs of proposed models compared to the baseline.

### *3.2. Configurations*

The proposed models use bi-LSTM with 512 hidden units for the encoder and decoder. The word embedding dim was set to 512. In the training phase, Adam optimizer (Kingma & Ba, 2015) with a fixed learning rate of 0.001 was used, and the number of tokens per batch was 3500. , A number of epochs is set to 10. For the graph encoder, we used 128-dimensional vectors for node representations. We stacked two layers of graph encoder to learn higher-level representation. The testing process was executed on Google Colab, which offers accessible GPUs for each session lasting up to 12 hours. The configuration in Google Colab consists of 12GB RAM and 16GB NVIDIA Tesla P100 GPU. The training time for the base and the proposed models is around 40 minutes.

### *3.3. Results and Discussions*

Once the models have been trained, a beam search with the size of 5 is utilized to find a translation that maximizes the conditional probabilities. First, we start with the results of the tst2013 test set as shown in Table 2.

As expected, bi-LSTM results are stronger than CNN ones. In general, we observed that using graph encoders and additional attention mechanisms leads to an improvement over the baseline model for both encoders. Particularly, the improvement is 0.53 BLEU for bi-LSTM-mean and 0.33 BLEU for CNN. This is slightly surprising as the potentially non-local semantic information should in principle, be more beneficial within a less powerful and local CNN encoder. Models with mean aggregators appear stronger than others with different aggregators.

***Table 2.*** *Experimental results*

| **Model** | **bi-LSTM** | **LightConv** |
|---|---|---|
| NMT baseline | 28.13 | 27.67 |
| Syntax-enhanced-NMT (mean aggregator) | **28.69** | 27.84 |
| Syntax-enhanced-NMT (max pooling aggregator) | 28.36 | **28.00** |
| Syntax-enhanced-NMT (gcn aggregator) | 28.44 | 27.50 |

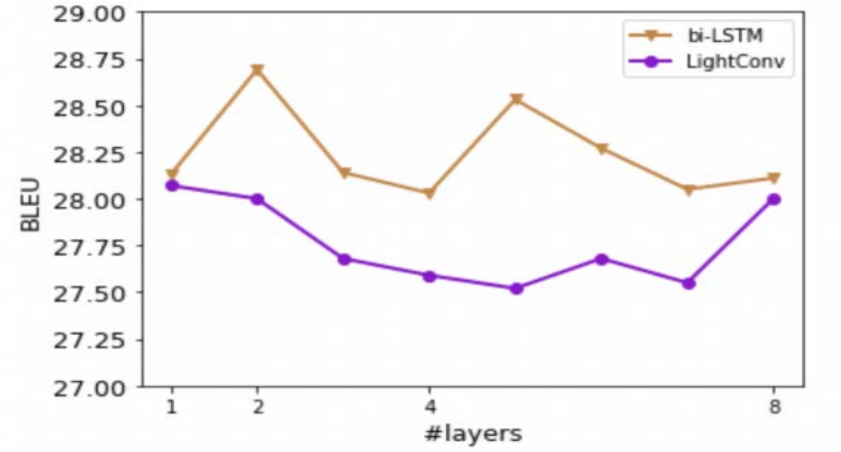


***Figure 3.*** *Results on various graph layers*
*(mean aggregator for bi-LSTM, max pooling aggregator for LightConv)*

We also use the tst2013 test set to investigate whether the more graph encoder layers are stacked, the better performance the model could achieve. To do so, we evaluated our proposed methods with one to eight layers. As shown in Figure 3, our models with two graph encoder layers performed the best. In general, there is a downward trend in the BLEU score as we stack more layers. If the number of layers is excessive, the model will become unstable due to the vanishing gradient and information redundancy. On the other hand, node representations cannot propagate far when the number of layers is small.

## 4. Conclusion

In this paper, we explore a graph encoder to explicitly model the syntactic information from constituency trees to NMT models. The experiments indicate that the syntax-enhanced

NMT models really outperformed the baseline and with the help of syntactic information, the translation quality was improved even on low-resource settings.

In the future, we aim to analyze the effect of immediate nodes as well as tree depth to each syntactic component during the decoding phase. Moreover, it would be interesting to study the impact of using source-side syntax together with the target-side syntax in the same NMT model.


- ***Conflict of Interest:*** *Authors have no conflict of interest to declare.*
- ***Acknowledgement:*** *We would like to thank the Computational Linguistics Center,University of Science, Vietnam National University – HCM City for providing linguistic resources.*


## REFERENCES


Bahdanau, D., Cho, K., & Bengio, Y. (2015). Neural machine translation by jointly learning to align and translate. In Y. Bengio & Y. LeCun (Eds.), 3rd international conference on learning representations, ICLR 2015, Sandiego, Ca, Usa, May 7-9, 2015, conference track proceedings. Retrieved from http://arxiv.org/abs/1409.0473

Cettolo, M., Niehues, J., Stüker, S., Bentivogli, L., Cattoni, R., & Federico, M. (2015). The iwslt 2015 evaluation campaign. Chen, H., Huang, S., Chiang, D., & Chen, J. (2017, July). Improved neural machine translation with a syntax-aware encoder and decoder. In Proceedings of the 55th annual meeting of the association for computational linguistics (volume 1: Long papers) (pp. 1936–1945). Vancouver, Canada: Association for Computational Linguistics. Retrieved from https://aclanthology.org/P17-1177 doi: https://doi.org/10.18653/v1/P17-1177

Eriguchi, A., Hashimoto, K., & Tsuruoka, Y. (2016, August). Tree-to-sequence attentional neural machine translation. In Proceedings of the 54th annual meeting of the association for computational linguistics (volume 1: Long papers) (pp. 823-833). Berlin, Germany: Association for Computational Linguistics. Retrieved from https://aclanthology.org/P16-1078 doi: https://doi.org/10.18653/v1/P16-1078

Gehring, J., Auli, M., Grangier, D., Yarats, D., & Dauphin, Y. N. (2017). Convolutional sequence to sequence learning. CoRR, abs/1705.03122 . Retrieved from http://arxiv.org/abs/1705.03122

Kingma, D. P., & Ba, J. (2015). Adam: A method for stochastic optimization. In Y. Bengio & Y. LeCun (Eds.), 3rd international conference on learning representations, ICLR 2015, San Diego, Ca, Usa, May 7-9, 2015, conference track proceedings. Retrieved from http://arxiv.org/abs/1412.6980

Koehn, P. (2004, July). Statistical significance tests for machine translation evaluation. In Proceedings of the 2004 conference on empirical methods in natural language processing (pp. 388-395). Barcelona, Spain: Association for Computational Linguistics. Retrieved from https://aclanthology.org/W04-3250

Koehn, P., & Hoang, H. (2007, June). Factored translation models. In Proceedings of the 2007 joint conference on empirical methods in natural language processing and computational natural language learning (EMNLP-CoNLL) (pp. 868-876). Prague, Czech Republic: Association for Computational Linguistics. Retrieved from https://aclanthology.org/D07-1091

Li, J., Xiong, D., Tu, Z., Zhu, M., Zhang, M., & Zhou, G. (2017, July). Modeling source syntax for neural machine translation. In Proceedings of the 55th annual meeting of the association for computational linguistics (volume 1: Long papers) (pp. 688-697). Vancouver, Canada: Association for Computational Linguistics. Retrieved from https://aclanthology.org/P17-1064 doi: https://doi.org/10.18653/v1/P17-1064

Nădejde, M., Reddy, S., Sennrich, R., Dwojak, T., Junczys-Dowmunt, M., Koehn, P., & Birch, A. (2017, September). Predicting target language CCG supertags improves neural machine translation. In Proceedings of the second conference on machine translation (pp. 68-79). Copenhagen, Denmark: Association for Computational Linguistics. Retrieved from https://aclanthology.org/W17-4707 doi: https://doi.org/10.18653/v1/W17-4707

Papineni, K., Roukos, S., Ward, T., & Zhu, W. J. (2002). Bleu: A method for automatic evaluation of machine translation. In Proceedings of the 40th annual meeting on association for computational linguistics (p. 311-318). USA: Association for Computational Linguistics. Retrieved from https://doi.org/10.3115/1073083.1073135 doi: https://doi.org/10.3115/1073083.1073135

Sennrich, R., Haddow, B., & Birch, A. (2016, August). Neural machine translation of rare words with subword units. In Proceedings of the 54th annual meeting of the association for computational linguistics (volume 1: Long papers) (pp. 1715-1725). Berlin, Germany: Association for Computational Linguistics. Retrieved from https://www.aclweb.org/anthology/P16-1162 doi: https://doi.org/10.18653/v1/P16-1162

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., . . . Polosukhin, I. (2017). Attention is all you need. CoRR, abs/1706.03762. Retrieved from http://arxiv.org/abs/1706.03762

Wang, Y., Wang, L., Zeng, X., Wong, D. F., Chao, L. S., & Lu, Y. (2014, June). Factored statistical machine translation for grammatical error correction. In Proceedings of the eighteenth conference on computational natural language learning: Shared task (pp. 83-90). Baltimore, Maryland: Association for Computational Linguistics. Retrieved from https://aclanthology.org/W14-1711 doi: https://doi.org/10.3115/v1/W14-1711

Wu, F., Fan, A., Baevski, A., Dauphin, Y. N., & Auli, M. (2019). Pay less attention with lightweight and dynamic convolutions. CoRR, abs/1901.10430 Retrieved from http://arxiv.org/abs/1901.10430

Xu, K., Wu, L., Wang, Z., Feng, Y., & Sheinin, V. (2018). Graph2seq: Graph to sequence learning with attention-based neural networks. CoRR, abs/1804.00823. Retrieved from http://arxiv.org/abs/1804.00823

# TĂNG CƯỜNG TRI THỨC CÚ PHÁP CHO DỊCH MÁY MẠNG NEURAL SỬ DỤNG BỘ MÃ HÓA ĐỒ THỊ

***Nguyễn Hồng Bửu Long*, Phạm Hùng Việt***

*Trường Đại học Khoa học Tự nhiên, Đại học Quốc gia Thành phố Hồ Chí Minh, Việt Nam*

**Tác giả liên hệ: Nguyễn Hồng Bửu Long – Email: nhblong@fit.hcmus.edu.vn*

*Ngày nhận bài: 11-10-2022; ngày nhận bài sửa: 25-10-2022; ngày duyệt đăng: 26-10-2022*

## TÓM TẮT

*Dịch máy mạng neural (NMT) là một mô hình mới trong dịch máy (MT) được hỗ trợ bởi những tiến bộ gần đây trong kĩ thuật học sâu. Với các mạng neural, NMT đã trở thành hướng tiếp cận dịch tự động hứa hẹn trong những năm gần đây. Mặc dù, đã có những thành công rõ ràng, NMT có một nhược điểm quan trọng là không có khả năng tích hợp tri thức cú pháp vào mô hình dịch. Bài báo này đề xuất mở rộng mô hình NMT để kết hợp thông tin cú pháp bổ sung từ cây phân tích cú pháp thành phần. Chúng tôi biểu diễn các cây cấu trúc thành phần dưới dạng biểu đồ được mã hóa bằng bộ mã hóa đồ thị để nâng cao cơ chế tập trung, giúp bộ giải mã có thể tập trung vào cả biểu diễn chuỗi tuần tự và đồ thị ở mỗi bước giải mã. Các thực nghiệm cho thấy kết quả khả quan của phương pháp được đề xuất trên bộ dữ liệu Anh-Việt, chứng minh tính hiệu quả của phương pháp NMT khi được tích hợp thêm thông tin tri thức cú pháp.*

***Từ khóa:*** cây cú pháp thành phần; mạng neural đồ thị; dịch máy mạng neural; cú pháp