PhoNLP: A joint multi-task learning model for Vietnamese part-of-speech tagging, named entity recognition and dependency parsing

Linh The Nguyen and **Dat Quoc Nguyen**
VinAI Research, Hanoi, Vietnam
{v.linhnt140, v.datnq9}@vinai.io

Abstract

We present the first multi-task learning model—named PhoNLP—for joint Vietnamese part-of-speech (POS) tagging, named entity recognition (NER) and dependency parsing. Experiments on Vietnamese benchmark datasets show that PhoNLP produces state-of-the-art results, outperforming a single-task learning approach that fine-tunes the pre-trained Vietnamese language model PhoBERT (Nguyen and Nguyen, 2020) for each task independently. We publicly release PhoNLP as an open-source toolkit under the Apache License 2.0. Although we specify PhoNLP for Vietnamese, our PhoNLP training and evaluation command scripts in fact can directly work for other languages that have a pre-trained BERT-based language model and gold annotated corpora available for the three tasks of POS tagging, NER and dependency parsing. We hope that PhoNLP can serve as a strong baseline and useful toolkit for future NLP research and applications to not only Vietnamese but also the other languages. Our PhoNLP is available at https://github.com/VinAIResearch/PhoNLP.

1 Introduction

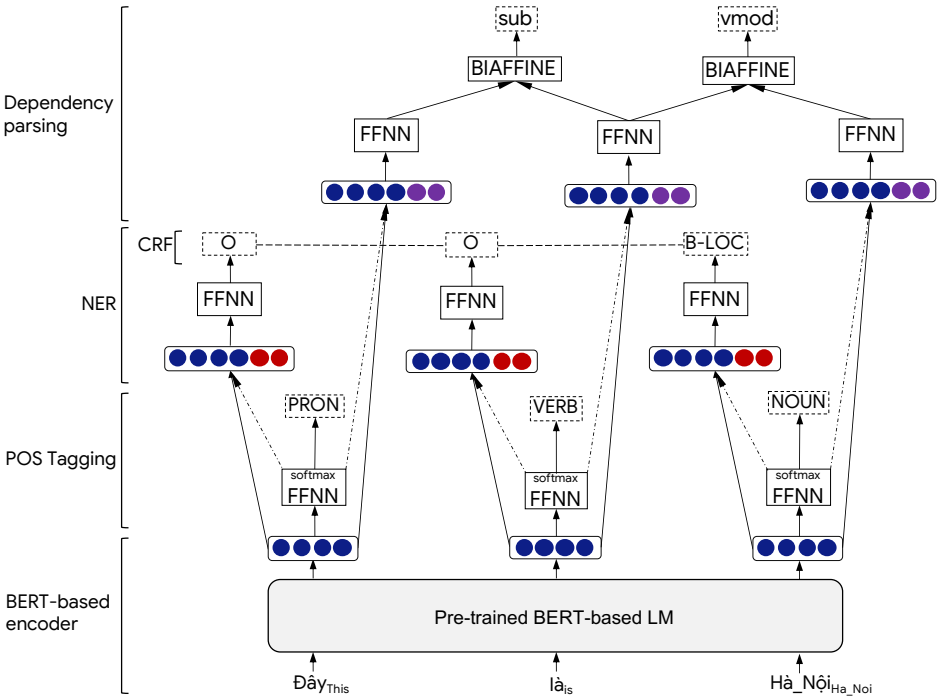
Vietnamese NLP research has been significantly explored recently. It has been boosted by the success of the national project on Vietnamese language and speech processing (VLSP) KC01.01/2006-2010 and VLSP workshops that have run shared tasks since 2013.[1] Fundamental tasks of POS tagging, NER and dependency parsing thus play important roles, providing useful features for many downstream application tasks such as machine translation (Tran et al., 2016), sentiment analysis (Bang and Sornlertlamvanich, 2018), relation extraction (To and Do, 2020), semantic parsing (Nguyen et al., 2020), open information extraction (Truong et al., 2017) and question answering (Nguyen et al., 2017; Le-Hong and Bui, 2018). Thus, there is a need to develop NLP toolkits for linguistic annotations w.r.t. Vietnamese POS tagging, NER and dependency parsing.

VnCoreNLP (Vu et al., 2018) is the previous public toolkit employing traditional feature-based machine learning models to handle those Vietnamese NLP tasks. However, VnCoreNLP is now no longer considered state-of-the-art because its performance results are significantly outperformed by ones obtained when fine-tuning PhoBERT—the current state-of-the-art monolingual pre-trained language model for Vietnamese (Nguyen and Nguyen, 2020). Note that there are no publicly available fine-tuned BERT-based models for the three Vietnamese tasks. Assuming that there would be, a potential drawback might be that an NLP package wrapping such fine-tuned BERT-based models would take a large storage space, i.e. three times larger than the storage space used by a BERT model (Devlin et al., 2019), thus it would not be suitable for practical applications that require a smaller storage space. Jointly multi-task learning is a promising solution as it might help reduce the storage space. In addition, POS tagging, NER and dependency parsing are related tasks: POS tags are essential input features used for dependency parsing and POS tags are also used as additional features for NER. Jointly multi-task learning thus might also help improve the performance results against the single-task learning (Ruder, 2019).

In this paper, we present a new multi-task learning model—named PhoNLP—for joint POS tagging, NER and dependency parsing. In particular, given an input sentence of words to PhoNLP, an encoding layer generates contextualized word embeddings that represent the input words. These contextualized word embeddings are fed into a POS tagging layer that is in fact a linear prediction layer (Devlin et al., 2019) to predict POS tags for the corresponding input words. Each predicted POS

[1] https://vlsp.org.vn/

Proceedings of NAACL-HLT 2021: Demonstrations, pages 1–7
June 6–11, 2021. ©2021 Association for Computational Linguistics



ID	Form	POS	NER	Head	DepRel
1	Đây$_{\text{This}}$	PRON	O	2	sub
2	là$_{\text{is}}$	VERB	O	0	root
3	Hà_Nội$_{\text{Ha_Noi}}$	NOUN	B-LOC	2	vmod

Figure 1: Illustration of our PhoNLP model.

tag is then represented by two "soft" embeddings that are later fed into NER and dependency parsing layers separately. More specifically, based on both the contextualized word embeddings and the "soft" POS tag embeddings, the NER layer uses a linear-chain CRF predictor (Lafferty et al., 2001) to predict NER labels for the input words, while the dependency parsing layer uses a Biaffine classifier (Dozat and Manning, 2017) to predict dependency arcs between the words and another Biaffine classifier to label the predicted arcs. Our contributions are summarized as follows:

- To the best of our knowledge, PhoNLP is the first proposed model to jointly learn POS tagging, NER and dependency parsing for Vietnamese.
- We discuss a data leakage issue in the Vietnamese benchmark datasets, that has not yet been pointed out before. Experiments show that PhoNLP obtains state-of-the-art performance results, outperforming the PhoBERT-based single task learning.
- We publicly release PhoNLP as an open-source toolkit that is simple to setup and efficiently run from both the command-line and Python API. We hope that PhoNLP can serve as a strong baseline and useful toolkit for future NLP research and downstream applications.

2 Model description

Figure 1 illustrates our PhoNLP architecture that can be viewed as a mixture of a BERT-based encoding layer and three decoding layers of POS tagging, NER and dependency parsing.

2.1 Encoder & Contextualized embeddings

Given an input sentence consisting of n word tokens $w_1, w_2, ..., w_n$, the encoding layer employs PhoBERT to generate contextualized latent feature embeddings $\mathbf{e}_i$ each representing the i^{th} word w_i:

$$\mathbf{e}_i = \text{PhoBERT}_{\text{base}}\big(w_{1:n}, i\big) \quad (1)$$

In particular, the encoding layer employs the **PhoBERT$_{\text{base}}$** version. Because PhoBERT uses BPE (Sennrich et al., 2016) to segment the input sentence with subword units, the encoding layer in fact represents the i^{th} word w_i by using the contextualized embedding of its first subword.

2.2 POS tagging

Following a common manner when fine-tuning a pre-trained language model for a sequence labeling

task (Devlin et al., 2019), the POS tagging layer is a linear prediction layer that is appended on top of the encoder. In particular, the POS tagging layer feeds the contextualized word embeddings $\mathbf{e}_i$ into a feed-forward network (FFNN_{POS}) followed by a softmax predictor for POS tag prediction:

$$\mathbf{p}_i = \text{softmax}\big(\text{FFNN}_{\text{POS}}\big(\mathbf{e}_i\big)\big) \quad (2)$$

where the output layer size of FFNN_{POS} is the number of POS tags. Based on probability vectors $\mathbf{p}_i$, a cross-entropy objective loss $\mathcal{L}_{\textbf{POS}}$ is calculated for POS tagging during training.

2.3 NER

The NER layer creates a sequence of vectors $\mathbf{v}_{1:n}$ in which each $\mathbf{v}_i$ is resulted in by concatenating the contextualized word embedding $\mathbf{e}_i$ and a "soft" POS tag embedding $\mathbf{t}_i^{(1)}$:

$$\mathbf{v}_i = \mathbf{e}_i \circ \mathbf{t}_i^{(1)} \quad (3)$$

where following Hashimoto et al. (2017), the "soft" POS tag embedding $\mathbf{t}_i^{(1)}$ is computed by multiplying a label weight matrix $\mathbf{W}^{(1)}$ with the corresponding probability vector $\mathbf{p}_i$:

$$\mathbf{t}_i^{(1)} = \mathbf{W}^{(1)}\mathbf{p}_i$$

The NER layer then passes each vector $\mathbf{v}_i$ into a FFNN (FFNN_{NER}):

$$\mathbf{h}_i = \text{FFNN}_{\text{NER}}\big(\mathbf{v}_i\big) \quad (4)$$

where the output layer size of FFNN_{NER} is the number of BIO-based NER labels.

The NER layer feeds the output vectors $\mathbf{h}_i$ into a linear-chain CRF predictor for NER label prediction (Lafferty et al., 2001). A cross-entropy loss $\mathcal{L}_{\textbf{NER}}$ is calculated for NER during training while the Viterbi algorithm is used for inference.

2.4 Dependency parsing

The dependency parsing layer creates vectors $\mathbf{z}_{1:n}$ in which each $\mathbf{z}_i$ is resulted in by concatenating $\mathbf{e}_i$ and another "soft" POS tag embedding $\mathbf{t}_i^{(2)}$:

$$\mathbf{z}_i = \mathbf{e}_i \circ \mathbf{t}_i^{(2)} \quad (5)$$
$$\mathbf{t}_i^{(2)} = \mathbf{W}^{(2)}\mathbf{p}_i$$

Following Dozat and Manning (2017), the dependency parsing layer uses FFNNs to split $\mathbf{z}_i$ into *head* and *dependent* representations:

$$\mathbf{h}_i^{(\text{A-H})} = \text{FFNN}_{\text{Arc-Head}}\big(\mathbf{z}_i\big) \quad (6)$$
$$\mathbf{h}_i^{(\text{A-D})} = \text{FFNN}_{\text{Arc-Dep}}\big(\mathbf{z}_i\big) \quad (7)$$
$$\mathbf{h}_i^{(\text{L-H})} = \text{FFNN}_{\text{Label-Head}}\big(\mathbf{z}_i\big) \quad (8)$$
$$\mathbf{h}_i^{(\text{L-D})} = \text{FFNN}_{\text{Label-Dep}}\big(\mathbf{z}_i\big) \quad (9)$$

To predict potential dependency arcs, based on input vectors $\mathbf{h}_i^{(\text{A-H})}$ and $\mathbf{h}_j^{(\text{A-D})}$, the parsing layer uses a Biaffine classifier's variant (Qi et al., 2018) that additionally takes into account the distance and relative ordering between two words to produce a probability distribution of arc heads for each word. For inference, the Chu–Liu/Edmonds' algorithm is used to find a maximum spanning tree (Chu and Liu, 1965; Edmonds, 1967). The parsing layer also uses another Biaffine classifier to label the predicted arcs, based on input vectors $\mathbf{h}_i^{(\text{L-H})}$ and $\mathbf{h}_j^{(\text{L-D})}$. An objective loss $\mathcal{L}_{\textbf{DEP}}$ is computed by summing a cross entropy loss for unlabeled dependency parsing and another cross entropy loss for dependency label prediction during training based on gold arcs and arc labels.

2.5 Joint multi-task learning

The final training objective loss $\mathcal{L}$ of our model PhoNLP is the weighted sum of the POS tagging loss $\mathcal{L}_{\text{POS}}$, the NER loss $\mathcal{L}_{\text{NER}}$ and the dependency parsing loss $\mathcal{L}_{\text{DEP}}$:

$$\mathcal{L} = \lambda_1\mathcal{L}_{\text{POS}} + \lambda_2\mathcal{L}_{\text{NER}} + (1-\lambda_1-\lambda_2)\mathcal{L}_{\text{DEP}} \quad (10)$$

Discussion: Our PhoNLP can be viewed as an extension of previous joint POS tagging and dependency parsing models (Hashimoto et al., 2017; Li et al., 2018; Nguyen and Verspoor, 2018; Nguyen, 2019; Kondratyuk and Straka, 2019), where we additionally incorporate a CRF-based prediction layer for NER. Unlike Hashimoto et al. (2017), Nguyen and Verspoor (2018), Li et al. (2018) and Nguyen (2019) that use BiLSTM-based encoders to extract contextualized feature embeddings, we use a BERT-based encoder. Kondratyuk and Straka (2019) also employ a BERT-based encoder. However, different from PhoNLP where we construct a hierarchical architecture over the POS tagging and dependency parsing layers, Kondratyuk and Straka (2019) do not make use of POS tag embeddings for dependency parsing.[2]

[2] In our preliminary experiments, not feeding the POS tag embeddings into the dependency parsing layer decreases the performance.

Task	#train	#valid	#test
POS tagging (leakage)	27000	870	2120
POS tagging (re-split)	23906	2009	3481
NER	14861	2000	2831
Dependency parsing	8977	200	1020

Table 1: Dataset statistics. **#train**, **#valid** and **#test** denote the numbers of training, validation and test sentences, respectively. Here, "POS tagging (leakage)" and "POS tagging (re-split)" refer to the statistics for POS tagging before and after re-splitting & sentence duplication removal, respectively.

3 Experiments

3.1 Setup

3.1.1 Datasets

To conduct experiments, we use the benchmark datasets of the VLSP 2013 POS tagging dataset,[3] the VLSP 2016 NER dataset (Nguyen et al., 2019) and the VnDT dependency treebank v1.1 Nguyen et al. (2014), following the setup used by the VnCoreNLP toolkit (Vu et al., 2018). Here, VnDT is converted from the Vietnamese constituent treebank (Nguyen et al., 2009).

Data leakage issue: We further discover an issue of data leakage, that has not yet been pointed out before. That is, all sentences from the VLSP 2016 NER dataset and the VnDT treebank are included in the VLSP 2013 POS tagging dataset. In particular, 90+% of sentences from both validation and test sets for NER and dependency parsing are included in the POS tagging training set, resulting in an unrealistic evaluation scenario where the POS tags are used as input features for NER and dependency parsing.

To handle the data leakage issue, we have to re-split the VLSP 2013 POS tagging dataset to avoid the data leakage issue: The POS tagging validation/test set now only contains sentences that appear in the union of the NER and dependency parsing validation/test sets (i.e. the validation/test sentences for NER and dependency parsing only appear in the POS tagging validation/test set). In addition, there are 594 duplicated sentences in the VLSP 2013 POS tagging dataset (here, sentence duplication is not found in the union of the NER and dependency parsing sentences). Thus we have to perform duplication removal on the POS tagging dataset. Table 1 details the statistics of the experimental datasets.

[3] https://vlsp.org.vn/vlsp2013/eval

3.1.2 Implementation

PhoNLP is implemented based on PyTorch (Paszke et al., 2019), employing the PhoBERT encoder implementation available from the transformers library (Wolf et al., 2020) and the Biaffine classifier implementation from Qi et al. (2020). We set both the label weight matrices $\mathbf{W}^{(1)}$ and $\mathbf{W}^{(2)}$ to have 100 rows, resulting in 100-dimensional soft POS tag embeddings. In addition, following Qi et al. (2018, 2020), FFNNs in equations 6–9 use 400-dimensional output layers.

We use the AdamW optimizer (Loshchilov and Hutter, 2019) and a fixed batch size at 32, and train for 40 epochs. The sizes of training sets are different, in which the POS tagging training set is the largest, consisting of 23906 sentences. Thus for each training epoch, we repeatedly sample from the NER and dependency parsing training sets to fill the gaps between the training set sizes. We perform a grid search to select the initial AdamW learning rate, λ_1 and λ_2. We find the optimal initial AdamW learning rate, λ_1 and λ_2 at 1e-5, 0.4 and 0.2, respectively. Here, we compute the average of the POS tagging accuracy, NER F_1-score and dependency parsing score LAS after each training epoch on the validation sets. We select the model checkpoint that produces the highest average score over the validation sets to apply to the test sets. Each of our reported scores is an average over 5 runs with different random seeds.

3.2 Results

Table 2 presents results obtained for our PhoNLP and compares them with those of a baseline approach of single-task training. For the single-task training approach: (i) We follow a common approach to fine-tune a pre-trained language model for POS tagging, appending a linear prediction layer on top of PhoBERT, as briefly described in Section 2.2. (ii) For NER, instead of a linear prediction layer, we append a CRF prediction layer on top of PhoBERT. (iii) For dependency parsing, predicted POS tags are produced by the learned single-task POS tagging model; then POS tags are represented by embeddings that are concatenated with the corresponding PhoBERT-based contextualized word embeddings, resulting in a sequence of input vectors for the Biaffine-based classifiers for dependency parsing (Qi et al., 2018). Here, the single-task training approach is based on the $\text{PhoBERT}_{\text{base}}$ version, employing the same hyper-

	Model	POS	NER	LAS	UAS
Leak.	Single-task	96.7†	93.69	78.77†	85.22†
	PhoNLP	**96.76**	**94.41**	**79.11**	**85.47**
Re-spl	Single-task	93.68	93.69	77.89	84.78
	PhoNLP	**93.88**	**94.51**	**78.17**	**84.95**

Table 2: Performance results (in %) on the test sets for POS tagging (i.e. accuracy), NER (i.e. F_1-score) and dependency parsing (i.e. LAS and UAS scores). "Leak." abbreviates "leakage", denoting the results obtained w.r.t. the data leakage issue. "Re-spl" denotes the results obtained w.r.t. the data re-split and duplication removal for POS tagging to avoid the data leakage issue. "Single-task" refers to as the single-task training approach. † denotes scores taken from the PhoBERT paper (Nguyen and Nguyen, 2020). Note that "Single-task" NER is not affected by the data leakage issue.

parameter tuning and model selection strategy that we use for PhoNLP.

Note that PhoBERT helps produce state-of-the-art results for multiple Vietnamese NLP tasks (including but not limited to POS tagging, NER and dependency parsing in a single-task training strategy), and obtains higher performance results than VnCoreNLP. However, in both the PhoBERT and VnCoreNLP papers (Nguyen and Nguyen, 2020; Vu et al., 2018), results for POS tagging and dependency parsing are reported w.r.t. the data leakage issue. Our "Single-task" results in Table 2 regarding "Re-spl" (i.e. the data re-split and duplication removal for POS tagging to avoid the data leakage issue) can be viewed as new PhoBERT results for a proper experimental setup. Table 2 shows that in both setups "Leak." and "Re-spl", our joint multi-task training approach PhoNLP performs better than the PhoBERT-based single-task training approach, thus resulting in state-of-the-art performances for the three tasks of Vietnamese POS tagging, NER and dependency parsing.

4 PhoNLP toolkit

We present in this section a basic usage of our PhoNLP toolkit. We make PhoNLP simple to setup, i.e. users can install PhoNLP from either source or pip (e.g. pip3 install phonlp). We also aim to make PhoNLP simple to run from both the command-line and the Python API. For example, annotating a corpus with POS tagging, NER and dependency parsing can be performed by using a simple command as in Figure 2.

Assume that the input file "`input.txt`" in Figure 2 contains a sentence "Tôi đang làm_việc tại VinAI ." ($I_{Tôi}$ $am_{đang}$ $working_{làm\_việc}$ $at_{tại}$ VinAI). Table 3 shows the annotated output in plain text form for this sentence. Similarly, we also get the same output by using the Python API as simple as in Figure 3. Furthermore, commands to (re-)train and evaluate PhoNLP using gold annotated corpora are detailed in the PhoNLP GitHub repository. Note that it is absolutely possible to directly employ our PhoNLP (re-)training and evaluation command scripts for other languages that have gold annotated corpora available for the three tasks and a pre-trained BERT-based language model available from the transformers library.

```
python3 run_phonlp.py --save_dir
./pretrained_phonlp --mode
annotate --input_file input.txt
--output_file output.txt
```

Figure 2: Minimal command to run PhoNLP. Here "save_dir" denote the path to the local machine folder that stores the pre-trained PhoNLP model.

1	Tôi	P	O	3	sub
2	đang	R	O	3	adv
3	làm_việc	V	O	0	root
4	tại	E	O	3	loc
5	VinAI	Np	B-ORG	4	pob
6	.	CH	O	3	punct

Table 3: The output in the output file "`output.txt`" for the sentence "Tôi đang làm_việc tại VinAI ." from the input file "`input.txt`" in Figure 2. The output is formatted with 6 columns representing word index, word form, POS tag, NER label, head index of the current word and its dependency relation type.

Speed test: We perform a sole CPU-based speed test using a personal computer with Intel Core i5 8265U 1.6GHz & 8GB of memory. For a GPU-based speed test, we employ a machine with a single NVIDIA RTX 2080Ti GPU. For performing the three NLP tasks jointly, PhoNLP obtains a speed at 15 sentences per second for the CPU-based test and 129 sentences per second for the GPU-based test, respectively, with an average of 23 word tokens per sentence and a batch size of 8.

5 Conclusion and future work

We have presented the first multi-task learning model PhoNLP for joint POS tagging, NER and dependency parsing in Vietnamese. Experiments on Vietnamese benchmark datasets show that PhoNLP outperforms its strong fine-tuned PhoBERT-based

```
import phonlp
# Automatically download the pre-trained PhoNLP model
# and save it in a local machine folder
phonlp.download(save_dir='./pretrained_phonlp')
# Load the pre-trained PhoNLP model
model = phonlp.load(save_dir='./pretrained_phonlp')
# Annotate a corpus
model.annotate(input_file='input.txt', output_file='output.txt')
# Annotate a sentence
model.print_out(model.annotate(text="Tôi đang làm_việc tại VinAI ."))
```

Figure 3: A simple and complete example code for using PhoNLP in Python.

single-task training baseline, producing state-of-the-art performance results. We publicly release PhoNLP as an easy-to-use open-source toolkit and hope that PhoNLP can facilitate future NLP research and applications. In future work, we will also apply PhoNLP to other languages.

References

Tran Sy Bang and Virach Sornlertlamvanich. 2018. Sentiment classification for hotel booking review based on sentence dependency structure and sub-opinion analysis. *IEICE Transactions on Information and Systems*, E101.D(4):909–916.

Yoeng-Jin Chu and Tseng-Hong Liu. 1965. On the Shortest Arborescence of a Directed Graph. *Science Sinica*, 14:1396–1400.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of NAACL*, pages 4171–4186.

Timothy Dozat and Christopher D. Manning. 2017. Deep Biaffine Attention for Neural Dependency Parsing. In *Proceedings of ICLR*.

Jack Edmonds. 1967. Optimum Branchings. *Journal of Research of the National Bureau of Standards*, 71:233–240.

Kazuma Hashimoto, Caiming Xiong, Yoshimasa Tsuruoka, and Richard Socher. 2017. A Joint Many-Task Model: Growing a Neural Network for Multiple NLP Tasks. In *Proceedings of EMNLP*, pages 1923–1933.

Dan Kondratyuk and Milan Straka. 2019. 75 Languages, 1 Model: Parsing Universal Dependencies Universally. In *Proceedings of EMNLP-IJCNLP*, pages 2779–2795.

John D. Lafferty, Andrew McCallum, and Fernando C. N. Pereira. 2001. Conditional Random Fields: Probabilistic Models for Segmenting and Labeling Sequence Data. In *Proceedings of ICML*, pages 282–289.

Phuong Le-Hong and Duc-Thien Bui. 2018. A Factoid Question Answering System for Vietnamese. In *Companion Proceedings of the The Web Conference 2018*, page 1049–1055.

Zuchao Li, Shexia He, Zhuosheng Zhang, and Hai Zhao. 2018. Joint Learning of POS and Dependencies for Multilingual Universal Dependency Parsing. In *Proceedings of the CoNLL 2018 Shared Task*, pages 65–73.

Ilya Loshchilov and Frank Hutter. 2019. Decoupled Weight Decay Regularization. In *Proceedings of ICLR*.

Anh Tuan Nguyen, Mai Hoang Dao, and Dat Quoc Nguyen. 2020. A Pilot Study of Text-to-SQL Semantic Parsing for Vietnamese. In *Findings of EMNLP 2020*, pages 4079–4085.

Dat Quoc Nguyen. 2019. A neural joint model for Vietnamese word segmentation, POS tagging and dependency parsing. In *Proceedings of ALTA*, pages 28–34.

Dat Quoc Nguyen and Anh Tuan Nguyen. 2020. PhoBERT: Pre-trained language models for Vietnamese. In *Findings of EMNLP 2020*, pages 1037–1042.

Dat Quoc Nguyen, Dai Quoc Nguyen, and Son Bao Pham. 2017. Ripple Down Rules for Question Answering. *Semantic Web*, 8(4):511–532.

Dat Quoc Nguyen, Dai Quoc Nguyen, Son Bao Pham, Phuong-Thai Nguyen, and Minh Le Nguyen. 2014. From Treebank Conversion to Automatic Dependency Parsing for Vietnamese. In *Proceedings of NLDB*, pages 196–207.

Dat Quoc Nguyen and Karin Verspoor. 2018. An improved neural network model for joint POS tagging and dependency parsing. In *Proceedings of the CoNLL 2018 Shared Task*, pages 81–91.

Huyen Nguyen, Quyen Ngo, Luong Vu, Vu Tran, and Hien Nguyen. 2019. VLSP Shared Task: Named Entity Recognition. *Journal of Computer Science and Cybernetics*, 34(4):283–294.

Phuong-Thai Nguyen, Xuan-Luong Vu, Thi-Minh-Huyen Nguyen, Van-Hiep Nguyen, and Hong-Phuong Le. 2009. Building a Large Syntactically-Annotated Corpus of Vietnamese. In *Proceedings of LAW*, pages 182–185.

Adam Paszke, Sam Gross, et al. 2019. PyTorch: An Imperative Style, High-Performance Deep Learning Library. In *Proceedings of NeurIPS 2019*, pages 8024–8035.

Peng Qi, Timothy Dozat, Yuhao Zhang, and Christopher D. Manning. 2018. Universal Dependency parsing from scratch. In *Proceedings of the CoNLL 2018 Shared Task*, pages 160–170.

Peng Qi, Yuhao Zhang, Yuhui Zhang, Jason Bolton, and Christopher D. Manning. 2020. Stanza: A python natural language processing toolkit for many human languages. In *Proceedings of ACL: System Demonstrations*, pages 101–108.

Sebastian Ruder. 2019. *Neural Transfer Learning for Natural Language Processing*. Ph.D. thesis, National University of Ireland, Galway.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016. Neural Machine Translation of Rare Words with Subword Units. In *Proceedings of ACL*, pages 1715–1725.

Huong Duong To and Phuc Do. 2020. Extracting triples from vietnamese text to create knowledge graph. In *Proceedings of KSE*, pages 219–223.

Viet Hong Tran, Huyen Thuong Vu, Thu Hoai Pham, Vinh Van Nguyen, and Minh Le Nguyen. 2016. A reordering model for Vietnamese-English statistical machine translation using dependency information. In *Proceedings of RIVF*, pages 125–130.

Diem Truong, Duc-Thuan Vo, and Uyen Trang Nguyen. 2017. Vietnamese open information extraction. In *Proceedings of SoICT*, page 135–142.

Thanh Vu, Dat Quoc Nguyen, Dai Quoc Nguyen, Mark Dras, and Mark Johnson. 2018. VnCoreNLP: A Vietnamese Natural Language Processing Toolkit. In *Proceedings of NAACL: Demonstrations*, pages 56–60.

Thomas Wolf, Lysandre Debut, et al. 2020. Transformers: State-of-the-Art Natural Language Processing. In *Proceedings of EMNLP 2020: System Demonstrations*, pages 38–45.