VnCoreNLP: A Vietnamese Natural Language Processing Toolkit

Thanh Vu[1], **Dat Quoc Nguyen**[2], **Dai Quoc Nguyen**[3], **Mark Dras**[4] and **Mark Johnson**[4]

[1]Newcastle University, United Kingdom; [2]The University of Melbourne, Australia;
[3]Deakin University, Australia; [4]Macquarie University, Australia

thanh.vu@newcastle.ac.uk, dqnguyen@unimelb.edu.au,
dai.nguyen@deakin.edu.au, {mark.dras, mark.johnson}@mq.edu.au

Abstract

We present an easy-to-use and fast toolkit, namely VnCoreNLP—a Java NLP annotation pipeline for Vietnamese. Our VnCoreNLP supports key natural language processing (NLP) tasks including word segmentation, part-of-speech (POS) tagging, named entity recognition (NER) and dependency parsing, and obtains state-of-the-art (SOTA) results for these tasks. We release VnCoreNLP to provide rich linguistic annotations to facilitate research work on Vietnamese NLP. Our VnCoreNLP is open-source and available at: https://github.com/vncorenlp/VnCoreNLP.

1 Introduction

Research on Vietnamese NLP has been actively explored in the last decade, boosted by the successes of the 4-year KC01.01/2006-2010 national project on Vietnamese language and speech processing (VLSP). Over the last 5 years, standard benchmark datasets for key Vietnamese NLP tasks are publicly available: datasets for word segmentation and POS tagging were released for the first VLSP evaluation campaign in 2013; a dependency treebank was published in 2014 (Nguyen et al., 2014); and an NER dataset was released for the second VLSP campaign in 2016. So there is a need for building an NLP pipeline, such as the Stanford CoreNLP toolkit (Manning et al., 2014), for those key tasks to assist users and to support researchers and tool developers of downstream tasks.

Nguyen et al. (2010) and Le et al. (2013) built Vietnamese NLP pipelines by wrapping existing word segmenters and POS taggers including: JVnSegmenter (Nguyen et al., 2006), vnTokenizer (Le et al., 2008), JVnTagger (Nguyen et al., 2010) and vnTagger (Le-Hong et al., 2010). However, these word segmenters and POS taggers are no longer considered SOTA models for Vietnamese (Nguyen and Le, 2016; Nguyen et al., 2016b).
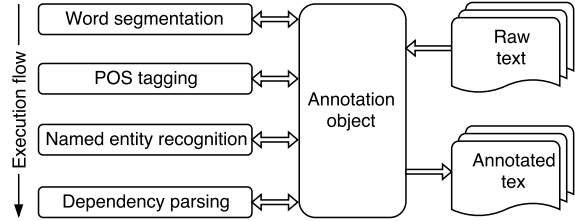

Figure 1: In pipeline architecture of VnCoreNLP, annotations are performed on an `Annotation` object.

Pham et al. (2017) built the NNVLP toolkit for Vietnamese sequence labeling tasks by applying a BiLSTM-CNN-CRF model (Ma and Hovy, 2016). However, Pham et al. (2017) did not make a comparison to SOTA traditional feature-based models. In addition, NNVLP is slow with a processing speed at about 300 words per second, which is not practical for real-world application such as dealing with large-scale data.

In this paper, we present a Java NLP toolkit for Vietnamese, namely VnCoreNLP, which aims to facilitate Vietnamese NLP research by providing rich linguistic annotations through key NLP components of word segmentation, POS tagging, NER and dependency parsing. Figure 1 describes the overall system architecture. The following items highlight typical characteristics of VnCoreNLP:

- **Easy-to-use** – All VnCoreNLP components are wrapped into a single .jar file, so users do not have to install external dependencies. Users can run processing pipelines from either the command-line or the Java API.

- **Fast** – VnCoreNLP is fast, so it can be used for dealing with large-scale data. Also it benefits users suffering from limited computation resources (e.g. users from Vietnam).

- **Accurate** – VnCoreNLP components obtain higher results than all previous published results on the same benchmark datasets.

2 Basic usages

Our design goal is to make VnCoreNLP simple to setup and run from either the command-line or the Java API. Performing linguistic annotations for a given file can be done by using a simple command as in Figure 2.

```
$ java -Xmx2g -jar VnCoreNLP.jar -fin
input.txt -fout output.txt
```

Figure 2: Minimal command to run VnCoreNLP.

Suppose that the file input.txt in Figure 2 contains a sentence "Ông Nguyễn Khắc Chúc đang làm việc tại Đại học Quốc gia Hà Nội." (Mr$_{\text{Ông}}$ Nguyen Khac Chuc is$_{\text{đang}}$ working$_{\text{làm\_việc}}$ at$_{\text{tại}}$ Vietnam National$_{\text{quốc\_gia}}$ University$_{\text{đại\_học}}$ Hanoi$_{\text{Hà\_Nội}}$). Table 1 shows the output for this sentence in plain text form.

1	Ông	Nc	O	4	sub
2	Nguyễn_Khắc_Chúc	Np	B-PER	1	nmod
3	đang	R	O	4	adv
4	làm_việc	V	O	0	root
5	tại	E	O	4	loc
6	Đại_học	N	B-ORG	5	pob
7	Quốc_gia	N	I-ORG	6	nmod
8	Hà_Nội	Np	I-ORG	6	nmod
9	.	CH	O	4	punct

Table 1: The output in file output.txt for the sentence 'Ông Nguyễn Khắc Chúc đang làm việc tại Đại học Quốc gia Hà Nội." from file input.txt in Figure 2. The output is in a 6-column format representing word index, word form, POS tag, NER label, head index of the current word, and dependency relation type.

Similarly, we can also get the same output by using the API as easy as in Listing 1.

```
VnCoreNLP pipeline = new VnCoreNLP() ;
Annotation annotation = new Annotation("
    Ông Nguyễn Khắc Chúc đang làm việc
    tại Đại học Quốc gia Hà Nội.");
pipeline.annotate(annotation);
String annotatedStr = annotation.
    toString();
```

Listing 1: Minimal code for an analysis pipeline.

In addition, Listing 2 provides a more realistic and complete example code, presenting key components of the toolkit. Here an annotation pipeline can be used for any text rather than just a single sentence, e.g. for a paragraph or entire news story.

3 Components

This section briefly describes each component of VnCoreNLP. Note that our goal is not to develop

```java
import vn.pipeline.*;
import java.io.*;
public class VnCoreNLPExample {
 public static void main(String[] args)
    throws IOException {
  // "wseg", "pos", "ner", and "parse"
     refer to as word segmentation, POS
      tagging, NER and dependency
      parsing, respectively.
  String[] annotators = {"wseg", "pos",
     "ner", "parse"};
  VnCoreNLP pipeline = new VnCoreNLP(
     annotators);
  // Mr Nguyen Khac Chuc is working at
     Vietnam National University, Hanoi
     . Mrs Lan, Mr Chuc's wife, is also
      working at this university.
  String str = "Ông Nguyễn Khắc Chúc
     đang làm việc tại Đại học Quốc gia
     Hà Nội. Bà Lan, vợ ông Chúc, cũng
     làm việc tại đây.";
  Annotation annotation = new Annotation
     (str);
  pipeline.annotate(annotation);
  PrintStream outputPrinter = new
     PrintStream("output.txt");
  pipeline.printToFile(annotation,
     outputPrinter);
  // Users can get a single sentence to
     analyze individually
  Sentence firstSentence = annotation.
     getSentences().get(0);
 }
}
```

Listing 2: A simple and complete example code.

new approach or model for each component task. Here we focus on incorporating existing models into a single pipeline. In particular, except a new model we develop for the language-dependent component of word segmentation, we apply traditional feature-based models which obtain SOTA results for English POS tagging, NER and dependency parsing to Vietnamese. The reason is based on a well-established belief in the literature that for a less-resourced language such as Vietnamese, we should consider using feature-based models to obtain fast and accurate performances, rather than using neural network-based models (King, 2015).

- **wseg** – Unlike English where white space is a strong indicator of word boundaries, when written in Vietnamese white space is also used to separate syllables that constitute words. So word segmentation is referred to as the key first step in Vietnamese NLP. We have proposed a transformation rule-based learning model for Vietnamese word segmentation, which obtains better segmentation accuracy and speed than all previous word segmenters. See details in Nguyen et al. (2018).

- **pos** – To label words with their POS tag, we apply MarMoT which is a generic CRF framework and a SOTA POS and morphological tagger (Mueller et al., 2013).[1]

- **ner** – To recognize named entities, we apply a dynamic feature induction model that automatically optimizes feature combinations (Choi, 2016).[2]

- **parse** – To perform dependency parsing, we apply the greedy version of a transition-based parsing model with selectional branching (Choi et al., 2015).[3]

4 Evaluation

We detail experimental results of the word segmentation (**wseg**) and POS tagging (**pos**) components of VnCoreNLP in Nguyen et al. (2018) and Nguyen et al. (2017b), respectively. In particular, our word segmentation component gets the highest results in terms of both segmentation F1 score at 97.90% and speed at 62K words per second.[4] Our POS tagging component also obtains the highest accuracy to date at 95.88% with a fast tagging speed at 25K words per second, and outperforms BiLSTM-CRF-based models. Following subsections present evaluations for the NER (**ner**) and dependency parsing (**parse**) components.

4.1 Named entity recognition

We make a comparison between SOTA feature-based and neural network-based models, which, to the best of our knowledge, has not been done in any prior work on Vietnamese NER.

Dataset: The NER shared task at the 2016 VLSP workshop provides a set of 16,861 manually annotated sentences for training and development, and a set of 2,831 manually annotated sentences for test, with four NER labels PER, LOC, ORG and MISC. Note that in both datasets, words are also supplied with gold POS tags. In addition, each word representing a full personal name are separated into syllables that constitute the word. So this annotation scheme results in an unrealistic scenario for a pipeline evaluation because: (**i**)

gold POS tags are not available in a real-world application, and (**ii**) in the standard annotation (and benchmark datasets) for Vietnamese word segmentation and POS tagging (Nguyen et al., 2009), each full name is referred to as a word token (i.e., all word segmenters have been trained to output a full name as a word and all POS taggers have been trained to assign a label to the entire full-name).

For a more realistic scenario, we merge those contiguous syllables constituting a full name to form a word.[5] Then we replace the gold POS tags by automatic tags predicted by our POS tagging component. From the set of 16,861 sentences, we sample 2,000 sentences for development and using the remaining 14,861 sentences for training.

Models: We make an empirical comparison between the VnCoreNLP's NER component and the following neural network-based models:

- BiLSTM-CRF (Huang et al., 2015) is a sequence labeling model which extends the BiLSTM model with a CRF layer.

- BiLSTM-CRF + CNN-char, i.e. BiLSTM-CNN-CRF, is an extension of BiLSTM-CRF, using CNN to derive character-based word representations (Ma and Hovy, 2016).

- BiLSTM-CRF + LSTM-char is an extension of BiLSTM-CRF, using BiLSTM to derive the character-based word representations (Lample et al., 2016).

- BiLSTM-CRF$_{+POS}$ is another extension to BiLSTM-CRF, incorporating embeddings of automatically predicted POS tags (Reimers and Gurevych, 2017).

We use a well-known implementation which is optimized for performance of all BiLSTM-CRF-based models from Reimers and Gurevych (2017).[6] We then follow Nguyen et al. (2017b, Section 3.4) to perform hyper-parameter tuning.[7]

Main results: Table 2 presents F1 score and speed of each model on the test set, where VnCoreNLP obtains the highest score at 88.55% with a fast speed at 18K words per second. In particular, VnCoreNLP obtains 10 times faster speed than

---

[1] http://cistern.cis.lmu.de/marmot/
[2] https://emorynlp.github.io/nlp4j/components/named-entity-recognition.html
[3] https://emorynlp.github.io/nlp4j/components/dependency-parsing.html
[4] All speeds reported in this paper are computed on a personal computer of Intel Core i7 2.2 GHz.

[5] Based on the gold label PER, contiguous syllables such as "Nguyễn/B-PER", "Khắc/I-PER" and "Chúc/I-PER" are merged to form a word as "Nguyễn_Khắc_Chúc/B-PER."
[6] https://github.com/UKPLab/emnlp2017-bilstm-cnn-crf
[7] We employ pre-trained Vietnamese word vectors from https://github.com/sonvx/word2vecVN.

Model	F1	Speed
VnCoreNLP	**88.55**	**18K**
BiLSTM-CRF	86.48	2.8K
+ CNN-char	88.28	1.8K
+ LSTM-char	87.71	1.3K
BiLSTM-CRF$_{+POS}$	86.12	_
+ CNN-char	88.06	_
+ LSTM-char	87.43	_

Table 2: F1 scores (in %) on the test set w.r.t. gold word-segmentation. "**Speed**" denotes the processing speed of the number of words per second (for VnCoreNLP, we include the time POS tagging takes in the speed).

the second most accurate model BiLSTM-CRF + CNN-char.

It is initially surprising that for such an isolated language as Vietnamese where all words are not inflected, using character-based representations helps producing 1+% improvements to the BiLSTM-CRF model. We find that the improvements to BiLSTM-CRF are mostly accounted for by the PER label. The reason turns out to be simple: about 50% of named entities are labeled with tag PER, so character-based representations are in fact able to capture common family, middle or given name syllables in 'unknown' full-name words. Furthermore, we also find that BiLSTM-CRF-based models do not benefit from additional predicted POS tags. It is probably because BiLSTM can take word order into account, while without word inflection, all grammatical information in Vietnamese is conveyed through its fixed word order, thus explicit predicted POS tags with noisy grammatical information are not helpful.

4.2 Dependency parsing

Experimental setup: We use the Vietnamese dependency treebank VnDT (Nguyen et al., 2014) consisting of 10,200 sentences in our experiments. Following Nguyen et al. (2016a), we use the last 1020 sentences of VnDT for test while the remaining sentences are used for training. Evaluation metrics are the labeled attachment score (LAS) and unlabeled attachment score (UAS).

Main results: Table 3 compares the dependency parsing results of VnCoreNLP with results reported in prior work, using the same experimental setup. The first six rows present the scores with gold POS tags. The next two rows show scores of VnCoreNLP with automatic POS tags which are produced by our POS tagging component. The last

	Model	LAS	UAS	Speed
Gold POS	VnCoreNLP	**73.39**	79.02	_
	VnCoreNLP$_{-NER}$	73.21	78.91	_
	BIST-bmstparser	73.17	**79.39**	_
	BIST-barchybrid	72.53	79.33	_
	MSTParser	70.29	76.47	_
	MaltParser	69.10	74.91	_
Auto POS	VnCoreNLP	**70.23**	76.93	8K
	VnCoreNLP$_{-NER}$	70.10	76.85	**9K**
	jPTDP	69.49	**77.68**	700

Table 3: LAS and UAS scores (in %) computed on all tokens (i.e. including punctuation) on the test set w.r.t. gold word-segmentation. "**Speed**" is defined as in Table 2. The subscript "–NER" denotes the model without using automatically predicted NER labels as features. The results of the MSTParser (McDonald et al., 2005), MaltParser (Nivre et al., 2007), and BiLSTM-based parsing models BIST-bmstparser and BIST-barchybrid (Kiperwasser and Goldberg, 2016) are reported in Nguyen et al. (2016a). The result of the jPTDP model for Vietnamese is mentioned in Nguyen et al. (2017b).

row presents scores of the joint POS tagging and dependency parsing model jPTDP (Nguyen et al., 2017a). Table 3 shows that compared to previously published results, VnCoreNLP produces the highest LAS score. Note that previous results for other systems are reported without using additional information of automatically predicted NER labels. In this case, the LAS score for VnCoreNLP without automatic NER features (i.e. VnCoreNLP$_{-NER}$ in Table 3) is still higher than previous ones. Notably, we also obtain a fast parsing speed at 8K words per second.

5 Conclusion

In this paper, we have presented the VnCoreNLP toolkit—an easy-to-use, fast and accurate processing pipeline for Vietnamese NLP. VnCoreNLP provides core NLP steps including word segmentation, POS tagging, NER and dependency parsing. Current version of VnCoreNLP has been trained without any linguistic optimization, i.e. we only employ existing pre-defined features in the traditional feature-based models for POS tagging, NER and dependency parsing. So future work will focus on incorporating Vietnamese linguistic features into these feature-based models.

VnCoreNLP is released for research and educational purposes, and available at: `https://github.com/vncorenlp/VnCoreNLP`.

References

Jinho D. Choi. 2016. Dynamic Feature Induction: The Last Gist to the State-of-the-Art. In *Proceedings of NAACL-HLT*. pages 271–281.

Jinho D. Choi, Joel Tetreault, and Amanda Stent. 2015. It Depends: Dependency Parser Comparison Using A Web-based Evaluation Tool. In *Proceedings of ACL-IJCNLP*. pages 387–396.

Zhiheng Huang, Wei Xu, and Kai Yu. 2015. Bidirectional LSTM-CRF models for sequence tagging. *arXiv preprint* arXiv:1508.01991.

Benjamin Philip King. 2015. *Practical Natural Language Processing for Low-Resource Languages*. Ph.D. thesis, The University of Michigan.

Eliyahu Kiperwasser and Yoav Goldberg. 2016. Simple and Accurate Dependency Parsing Using Bidirectional LSTM Feature Representations. *Transactions of the Association for Computational Linguistics* 4:313–327.

Guillaume Lample, Miguel Ballesteros, Sandeep Subramanian, Kazuya Kawakami, and Chris Dyer. 2016. Neural Architectures for Named Entity Recognition. In *Proceedings of NAACL-HLT*. pages 260–270.

Hong Phuong Le, Thi Minh Huyen Nguyen, Azim Roussanaly, and Tuong Vinh Ho. 2008. A hybrid approach to word segmentation of Vietnamese texts. In *Proceedings of LATA*. pages 240–249.

Ngoc Minh Le, Bich Ngoc Do, Vi Duong Nguyen, and Thi Dam Nguyen. 2013. VNLP: An Open Source Framework for Vietnamese Natural Language Processing. In *Proceedings of SoICT*. pages 88–93.

Phuong Le-Hong, Azim Roussanaly, Thi Minh Huyen Nguyen, and Mathias Rossignol. 2010. An empirical study of maximum entropy approach for part-of-speech tagging of Vietnamese texts. In *Proceedings of TALN*.

Xuezhe Ma and Eduard Hovy. 2016. End-to-end Sequence Labeling via Bi-directional LSTM-CNNs-CRF. In *Proceedings of ACL (Volume 1: Long Papers)*. pages 1064–1074.

Christopher D. Manning, Mihai Surdeanu, John Bauer, Jenny Finkel, Steven J. Bethard, and David McClosky. 2014. The Stanford CoreNLP natural language processing toolkit. In *Proceedings of ACL 2014 System Demonstrations*. pages 55–60.

Ryan McDonald, Koby Crammer, and Fernando Pereira. 2005. Online Large-margin Training of Dependency Parsers. In *Proceedings of ACL*. pages 91–98.

Thomas Mueller, Helmut Schmid, and Hinrich Schütze. 2013. Efficient Higher-Order CRFs for Morphological Tagging. In *Proceedings of EMNLP*. pages 322–332.

Cam-Tu Nguyen, Trung-Kien Nguyen, et al. 2006. Vietnamese Word Segmentation with CRFs and SVMs: An Investigation. In *Proceedings of PACLIC*. pages 215–222.

Cam-Tu Nguyen, Xuan-Hieu Phan, and Thu-Trang Nguyen. 2010. JVnTextPro: A Java-based Vietnamese Text Processing Tool. http://jvntextpro.sourceforge.net/.

Dat Quoc Nguyen, Mark Dras, and Mark Johnson. 2016a. An empirical study for Vietnamese dependency parsing. In *Proceedings of ALTA*. pages 143–149.

Dat Quoc Nguyen, Mark Dras, and Mark Johnson. 2017a. A Novel Neural Network Model for Joint POS Tagging and Graph-based Dependency Parsing. In *Proceedings of the CoNLL 2017 Shared Task*. pages 134–142.

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, Dai Quoc Nguyen, Thanh Vu, Mark Dras, and Mark Johnson. 2018. A Fast and Accurate Vietnamese Word Segmenter. In *Proceedings of LREC*. page to appear.

Dat Quoc Nguyen, Thanh Vu, Dai Quoc Nguyen, Mark Dras, and Mark Johnson. 2017b. From Word Segmentation to POS Tagging for Vietnamese. In *Proceedings of ALTA*. pages 108–113.

Phuong Thai Nguyen, Xuan Luong Vu, et al. 2009. Building a Large Syntactically-Annotated Corpus of Vietnamese. In *Proceedings of LAW*. pages 182–185.

Tuan-Phong Nguyen and Anh-Cuong Le. 2016. A Hybrid Approach to Vietnamese Word Segmentation. In *Proceedings of RIVF*. pages 114–119.

Tuan Phong Nguyen, Quoc Tuan Truong, Xuan Nam Nguyen, and Anh Cuong Le. 2016b. An Experimental Investigation of Part-Of-Speech Taggers for Vietnamese. *VNU Journal of Science: Computer Science and Communication Engineering* 32(3):11–25.

Joakim Nivre, Johan Hall, et al. 2007. MaltParser: A language-independent system for data-driven dependency parsing. *Natural Language Engineering* 13(2):95–135.

Thai-Hoang Pham, Xuan-Khoai Pham, Tuan-Anh Nguyen, and Phuong Le-Hong. 2017. NNVLP: A Neural Network-Based Vietnamese Language Processing Toolkit. In *Proceedings of the IJCNLP 2017 System Demonstrations*. pages 37–40.

Nils Reimers and Iryna Gurevych. 2017. Reporting Score Distributions Makes a Difference: Performance Study of LSTM-networks for Sequence Tagging. In *Proceedings of EMNLP*. pages 338–348.