*Journal of Computer Science and Cybernetics, V.34, N.4 (2018), 311–321*
DOI 10.15625/1813-9663/34/4/13163

# A FEATURE-BASED MODEL FOR NESTED NAMED-ENTITY RECOGNITION AT VLSP-2018 NER EVALUATION CAMPAIGN

PHAM QUANG NHAT MINH

*Alt Vietnam Co., Ltd. Hanoi, Vietnam*
*pham.minh@alt.ai*

**Abstract.** In this paper, we describe our named-entity recognition system at VLSP 2018 evaluation campaign. We formalized the task as a sequence labeling problem using B-I-O encoding scheme and applied a feature-based model which combines word, word-shape features, Brown-cluster-based features, and word-embedding-based features. We compared several methods to deal with nested entities in the dataset. We showed that combining tags of entities at all levels to train a single sequence labeling model (joint-tag model) improved the accuracy of nested named-entity recognition.

**Keywords.** Nested named-entity recognition; Feature-based model; Conditional random fields.

## 1. INTRODUCTION

Named-entity recognition (NER) is an important task in information extraction. The task is to identify in a text, spans that are entities and classify them into pre-defined categories. There have been some conferences and shared tasks for evaluating NER systems in English and other languages, such as MUC- [17], CoNLL 2002 [15] and CoNLL 2003 [16].

In Vietnamese language, VLSP 2016 NER evaluation campaign [3] is the first evaluation campaign that aims to systematically compare NER systems for Vietnamese language. Similar to CoNLL 2003 shared-task, in VLSP 2016, four named-entity types were considered: person (PER), organization (ORG), location (LOC), and miscellaneous entities (MISC). In VLSP 2016, organizers provided the training/test data with gold word segmentation, PoS and chunking tags. While that setting can help participant teams to reduce effort of data processing and solely focus on developing NER algorithms, it is not a realistic setting. In VLSP 2018 NER evaluation campaign, only raw texts with XML tags were provided. Therefore, we need to choose appropriate Vietnamese NLP tools for preprocessing steps such as word segmentation, PoS tagging, and chunking. VLSP 2018 NER campaign also differs from VLSP 2016 NER campaign in that the official evaluation considers nested named-entities of all levels.

There are quite few work on nested named-entity recognition. Previous work approached to nested named-entity recognition by formalizing the task as a discriminative constituency parsing [2], or learning a hypergraph representation for nested entities using features extracted from a recurrent neural network [4]. For Vietnamese language, nested-named entity recognition has been addressed in [12] in which Multilayer Recurrent Neural Networks was used to recognize all nested entities at the same time. In [12], authors also investigated methods of using a sequence of BI-LSTM-CRF models and using a sequence of CRF models in which the output of lower-level model will be used as input for higher-level models. Experiments were conducted on VLSP 2016 NER data.

In this paper, we describe our NER system at VLSP 2018 NER evaluation campaign. We applied a feature-based model which combines word, word-shape features, Brown-cluster-based features, and word-embedding-based features and adopted Conditional Random Fields (CRF) [5] for training and testing. We proposed some treatments for nested-named entity recognition including: 1) combining results of separated NER models in which each is trained for one nested level; and 2) using a single NER model which is trained by using the data whose labels are generated by combining labels of all nested levels (join-tag model). To the best of our understanding, for Vietnamese language, the joint-tag model is the first work that combines entity tags of all nested levels to train a single joint model for recognizing nested entities. Experimental results showed that the joint-tag model obtained the best overall F1 score on the test set among methods that we investigated. Our system also obtained the first rank among participating systems at VLSP 2018 NER task. Another advantage of our proposed methods is that they are easy to implement and do not require intense computing resource for training models. We released the code and necessary resources for the sake of research reproducibility [1].

The rest of the paper is organized as follows. In Section 2, we described our participant NER system. In Section 3, we present our evaluation results. Finally, Section 4 gives conclusions about the work.

## 2. SYSTEM DESCRIPTION

We formalize NER task as a sequence labeling problem by using the B-I-O tagging scheme and we apply a popular sequence labeling model, Conditional Random Fields (CRF) to the problem.

In this section, first we present our proposed methods of recognizing nested named-entities. After that, we present how we preprocessed the data and then describe features that we used in our NER models.

### 2.1. Treatments of nested named-entities

#### 2.1.1. Categories of entity levels

In the VLSP 2018 NER task, there are nested entities in the provided datasets. An entity may contain other entities inside them. We categorize entities in VLSP 2018 NER dataset into three levels.

- Level-1 entities are entities that do not contain any other entities inside them. For example: ⟨ENAMEX TYPE="LOC"⟩ Hà Nội ⟨/ENAMEX⟩.
- Level-2 entities are entities that contain only level-1 entities inside them. For example: ⟨ENAMEX TYPE="ORG"⟩ UBND thành phố ⟨ENAMEX TYPE="LOC"⟩ Hà Nội⟨ /ENAMEX⟩ ⟨/ENAMEX⟩ .
- Level-3 entities are entities that contain at least one level-2 entity and may contain some level-1 entities. For example ⟨ENAMEX TYPE="ORG"⟩ Khoa Toán, ⟨ENAMEX TYPE="ORG"⟩ ĐHQG ⟨ENAMEX TYPE="LOC"⟩ Hà Nội ⟨/ENAMEX⟩ ⟨/ENAMEX⟩ ⟨/ENAMEX⟩

Our categorization scheme is different from the common categorization scheme which categorizes entities into top-level entities (i.e. entities that are not included in any entity)

[1]The code and resources are available at `https://github.com/minhpqn/vietner`

*Table 1.* Generating joint-tags by combining entity tags at all levels of a token

| **Word** | **Level-1 Tag** | **Level-2 Tag** | **Joint Tag** |
|---|---|---|---|
| ông | O | O | O+O |
| Ngô_Văn_Quý | B-PER | O | B-PER+O |
| - | O | O | O+O |
| Phó | O | O | O+O |
| Chủ_tịch | O | O | O+O |
| UBND | O | B-ORG | O+B-ORG |
| TP | B-LOC | I-ORG | B-LOC+I-ORG |
| Hà_Nội | I-LOC | I-ORG | I-LOC+I-ORG |

and entities of other nested levels [12]. We observe that in our categorization scheme, entities in the same level may be more similar in terms of entity lengths compared with the others. However, the limitation of our categorization scheme is that most of level-2 and level-3 entities (categorized by our scheme) is of the ORGANIZATION type.

In our data statistics, we see that the number of level-3 entities is too small compared with the number of level-1 and level-2 entities, so we decided to ignore them in building models. We just train models to recognize level-1 and level-2 entities.

### 2.1.2. Treatments of nested named-entities

In order to recognize nested named-entities, we investigated the two methods.

- In the first method, we combined results of two separated NER models. Level-1 model, which is trained by using level-1 entity tags of tokens, is used for recognizing level-1 entities. Level-2 model, which is trained by using level-2 entity tags of tokens, is used for recognizing level-2 entities.

- In the second method, we used a joint-tag model which is a single model for recognizing both level-1 and level-2 entities. Joint-tag model is trained by using joint tags which combine level-1 and level-2 tags of each word. In testing, after the joint-tag model returned the predicted tags for tokens, we split joint tags to get level-1 and level-2 tags of tokens. Table 1 shows an example of how we combined entity tags at all levels of a token to create joint tags.

The advantage of the joint-tag model against the method of using two separated models for level-1 and level-2 entity recognition is that the joint-tag model uses supervised signals from both level-1 and level-2 entity tags. Therefore, the joint-tag model may be more precise than separated models, especially in level-2 entity recognition. Experimental results confirmed our hypothesis.

The disadvantage of the joint-tag model is that there are more labels in the model than in separated models, so it requires larger training time than separated models.

After predicting level-1 and level-2 tags of tokens in a sentence, we combine them to extract named-entities of the two levels in the sentence. In the example shown in Table 1, if we

have predicted level-1 and level-2 tags for tokens in the example sentence (in columns "Level-1 Tag" and "Level-2 Tag", we can extract two level-1 entities "Ngô_Văn_Quý" (PERSON), "TP Hà_Nội" and one level-2 entity "UBND TP Hà_Nội" (ORG).

In recognition, there are some cases that a predicted level-1 entity contains level-2 entities inside them. In such cases, we omit level-2 entities included in level-1 entities. The reason is that in preliminary experiments conducted on the development set, we see that the accuracy of level-1 entity recognition is higher than the accuracy of level-2 entity recognition.

### 2.2. Preprocessing

In the proposed NER system, we performed sentence and word segmentation on the data. We did not perform POS tagging and chunking because automatically extracted POS tagging and chunking tags were shown not to be effective in our previous work of feature-based NER models for Vietnamese [10]. For sentence segmentation, we just used a simple regular expression to detect sentence boundaries that match the pattern: period followed by a space and upper-case character. Actually, to produce result submissions, we also tried not to perform sentence segmentation. Experiments showed that performing sentence segmentation did not increase the overall result.

For word segmentation, we adopted RDRsegmenter [11] which is the state-of-the-art Vietnamese word segmentation tool. Both training and development data are then converted into data files in CoNLL 2003 format with two columns: words and their B-I-O tags. Due to errors of word segmentation tool, there may be boundary-conflict problem between entity boundary and word boundary. In such cases, we decided to tag words as "O" (outside entity).

### 2.3. Features

Basically, features in the proposed NER model are categorized into word, word-shape features, features based on word representations including word clusters and word embedding. Note that, we extract unigram and bigram features within the context surrounding the current token with the window size of 5. More specifically, for a feature $F$ of the current word, unigram and bigram features are as follows.

- **unigrams**: $F$[-2], $F$[-1], $F$[0], $F$[1], $F$[2]
- **bigrams**: $F$[-2]$F$[-1], $F$[-1]$F$[0], $F$[0]$F$[1], $F$[1]$F$[2]

#### 2.3.1. Word features

We extract word-identity unigrams and bigrams within the window of size 5. We use both word surfaces and their lower-case forms. Beside words, we also extract prefixes and suffixes of surfaces of words within the context of the current word. In our model, we use prefixes and suffixes of lengths from 1 to 4 characters.

#### 2.3.2. Word shapes

In addition to word identities, we use word shapes to improve the prediction ability of the model (especially for unknown or rare words) and to reduce the data spareness problem.

*Table 2.* Word shape features

| **Feature** | **Description** | **Example** |
|---|---|---|
| shape | orthographic shapes of the token | "*Dồng*" → "*ULLL*" |
| shaped | shorten version of shape | "*Dồng*" → "*UL*" |
| type | category of the token such as "AllUpper", "AllDigit", etc | "1234" → "AllDigit" |
| fregex | features based on token regular expression [6] | |
| mix | is mixed case letters | "*iPhone*" |
| acr | is capitalized letter with period | "*H.*", "*Th.*", "*U.S.*' |
| ed | token starts with alphabet chars and ends with digits | "*A9*", "*B52*" |
| hyp | contains hyphen | "*New-York*" |
| da | is date | "*03-11-1984*", "*03/10*" |
| na | is name | "*Buôn_Mê_Thuột*" |
| co | is code | "*21B*" |
| wei | is weight | "*2kg*" |
| 2d | is two-digit number | "*12*" |
| 4d | is four-digit number | "*1234*" |
| d&a | contains digits and alphabet | "*12B*" |
| d&- | contains digits and hyphens | "*9-2*" |
| d&/ | contains digits and backslash | "*9/2*" |
| d&, | contains digits and comma | "*10,000*" |
| d&. | contains digits and period | "*10.000*" |
| up | contains an upper-case character followed by a period | "*M.*" |
| iu | first character is upper-case | "*Việt_Nam*" |
| au | all characters of the token are upper-case | "*IBM*" |
| al | all characters are lower-case | "*học_sinh*" |
| ad | all digits | "*1234*" |
| ao | all characters are neither alphabet characters nor digits | "*;*" |
| cu | contains at least one upper-case character | "*iPhone*" |
| cl | contains at least one lower-case character | "*iPhone*" |
| ca | contains at least one alphabet character | "*s12456*" |
| cd | contains at least one digit | "*1A*" |
| cs | contains at least 1 character that is not alphabet or digit | "*10.000*" |

We used the same word shapes as presented in [10]. Table 2 shows the list of word-shape features used in our NER model.

### 2.3.3. Brown cluster-based features

Brown clustering algorithm is a hierarchical clustering algorithm for assigning words to clusters [1]. Each cluster contains words which are semantically similar. Output clusters are represented as bit-strings. Brown-cluster-based features in our NER model include whole bit-string representations of words and their prefixes of lengths 2, 4, 6, 8, 10, 12, 16, 20.

*Table 3.* Number of entities of each type in each level in train/development and test set. **Lv** stands for Level

| **Type** | **Train** | | | **Dev** | | | **Test** | | |
|---|---|---|---|---|---|---|---|---|---|
| | **Lv-1** | **Lv-2** | **Lv-3** | **Lv-1** | **Lv-2** | **Lv-3** | **Lv-1** | **Lv-2** | **Lv-3** |
| LOC | 8831 | 7 | 0 | 3043 | 2 | 0 | 2525 | 2 | 0 |
| ORG | 3471 | 1655 | 63 | 1203 | 690 | 14 | 1616 | 557 | 22 |
| PER | 6427 | 0 | 0 | 2168 | 0 | 0 | 3518 | 1 | 0 |
| MISC | 805 | 1 | 0 | 179 | 1 | 0 | 296 | 0 | 0 |
| Total | 19534 | 1663 | 63 | 6593 | 694 | 14 | 7955 | 561 | 22 |

Note that, we only extract unigrams for Brown-cluster-based features.

In experiments, we used the Brown clustering implementation of Liang [8] and applied the tool on the raw text data collected through a Vietnamese news portal. We performed word clustering on the same preprocessed text data which were used to generate word embeddings in [7]. The number of word clusters used in our experiments is 5120.

#### 2.3.4. Word embeddings

Word-embedding features have been used for a CRF-based Vietnamese NER model in [7]. The basic idea is adding unigram features corresponding to dimensions of word representation vectors.

In the paper, we apply the same word-embedding features as in [7]. We generated pre-trained word vectors by applying Glove [14] on the same text data used to run Brown clustering. The dimension of word vectors is 25.

## 3. EVALUATION

### 3.1. Data sets

Table 3 showed the data statistics on training set, development set, and official test set. The number of ORGANIZATION entities (ORG) at level 3 is too small, so we only consider level-1 and level-2 entities in training models. Almost level-2 entities are of ORG types.

### 3.2. Evaluation measures

Evaluation measures in our experiments are Precision, Recall, and F1 score. We report results of recognizing level-1 entities, level-2 entities and entities of all levels. We use the Perl script provided in CoNLL-2000 Shared Task [2] for evaluating level-1 and level-2 named-entity recognition. Due to the fact that word segmentation may cause boundary conflict between entities and words, we convert words in the data into syllables before we evaluate Precision, Recall, and F1 score.

For calculating Precision, Recall, and F1 score of recognizing entities of all levels, we used the evaluation program provided by VLSP 2018 organizers. [3]

[2] `https://www.clips.uantwerpen.be/conll2000/chunking/conlleval.txt`

[3] The evaluation program is available at `https://github.com/minhpqn/vietner`

### 3.3. CRF tool and parameters

In experiments, we adopted CRF suite [13], an implementation of linear-chain (first-order Markov) CRF. That toolkit allows us to easily incorporate both binary and numeric features such as word embedding features. In training, we use Stochastic Gradient Descent algorithm with L2 regularization and the coefficient for L2 regularization is 3.2.

### 3.4. Nested named-entity recognition methods

We compare three methods of recognizing nested named-entity recognition as follows.

- Using Level-1 model and Level-2 model for recognizing level-1 and level-2 entities, respectively. We refer this method as **Separated** method.
- Using Joint-tag model to recognize joint tags for each word of a sentence, then split joint tags into level-1 and level-2 tags. We refer this method as **Joint** method.
- We use the Joint-tag model for recognizing level-2 entities and level-1 model for recognizing level-1 entities. We refer this method as **Hybrid** method.

Our intention of using three above methods for nested named-entity recognition is to test our hypothesis that the joint-tag model can leverage supervised signals from both level-1 and level-2 entity tags, so it will improve the overall result of nested named-entity recognition.

### 3.5. Experiments

We conducted two experiments as follows.

- Experiment 1: We used the training set, which was provided by VLSP 2018 organizers for training level-1, level-2 and joint-tag models. Sentence segmentation was not used.
- Experiment 2: We combined provided training and development data to make a larger training data, then used the combined training data to train NER models. That is the method we used to generate official submission results at VLSP 2018. In experiment 2, we compared two preprocessing methods: performing sentence segmentation and not performing sentence segmentation.

In each experiment, we reported results of entity recognition for level-1 and level-2 entities and the overall nested named-entity recognition results of three methods **Separated**, **Joint** and **Hybrid**.

### 3.6. Results

#### 3.6.1. Experiment 1

Table 4 and Table 5 show the experimental results of recognizing level-1 and level-2 entities, respectively. Table 6 presents the overall results on development and test set which consider all entity levels. **Joint** method and **Hybrid** method outperformed **Separated** in terms of level-2 entity recognition. That result indicated that the joint-tag model is better than level-2 model in recognizing level-2 entities.

*Table 4.* Level-1 entity recognition results of three methods, which used models trained on the provided training data

| Method | Dev set | | | Test set | | |
|---|---|---|---|---|---|---|
| | **Precision** | **Recall** | **F1** | **Precision** | **Recall** | **F1** |
| **Separated** | 84.98 | 89.38 | 87.12 | 72.17 | 78.50 | 75.20 |
| **Joint** | 85.30 | 88.85 | 87.04 | 73.36 | 78.30 | 75.75 |
| **Hybrid** | 85.04 | 89.35 | 87.15 | 72.16 | 78.44 | 75.17 |

*Table 5.* Level-2 entity recognition results of three methods, which used models trained on the provided training data

| Model | Dev set | | | Test set | | |
|---|---|---|---|---|---|---|
| | **Precision** | **Recall** | **F1** | **Precision** | **Recall** | **F1** |
| **Separated** | 64.41 | 90.67 | 75.32 | 35.12 | 82.08 | 49.19 |
| **Joint** | 70.61 | 87.03 | 77.96 | 44.03 | 78.66 | 56.46 |
| **Hybrid** | 69.31 | 88.42 | 77.71 | 41.18 | 80.77 | 54.55 |

*Table 6.* NER results on development and test data sets for all entity levels. We used models trained on the provided training data

| Method | Dev set | | | Test set | | |
|---|---|---|---|---|---|---|
| | **Precision** | **Recall** | **F1** | **Precision** | **Recall** | **F1** |
| **Separated** | 87.01 | 81.08 | 83.94 | 76.83 | 69.12 | 72.77 |
| **Joint** | 86.17 | 81.84 | 83.95 | 76.98 | 71.10 | 73.92 |
| **Hybrid** | 86.86 | 81.64 | 84.17 | 76.81 | 69.58 | 73.02 |

*Table 7.* Six submitted runs

| Runs | Method | Sent Segmentation? |
|---|---|---|
| Run-1 | Hybrid | YES |
| Run-2 | Hybrid | NO |
| Run-3 | Joint | YES |
| Run-4 | Joint | NO |
| Run-5 | Separated | YES |
| Run-6 | Separated | NO |

### 3.6.2. Experiment 2 (submission results)

In Experiment 2, we trained models on the data set obtained by combining provided training and development data and used the trained models for recognizing entities on the test set.

We submitted six runs at VLSP 2018 NER evaluation campaign as showed in Table 7. We compared two preprocessing approaches: with sentence segmentation and without sentence segmentation. The reason why we tried those preprocessing approaches is that we would like to know the influence of sequence lengths on the accuracy of our model.

Table 8 presents the results of our six submission runs in recognizing level-1 and level-2

*Table 8.* Evaluation results of recognizing level-1 entities and level-2 entities on the test set of the six submission runs

| **Method** | **Level-1 Entity** | | | **Level-2 Entity** | | |
|---|---|---|---|---|---|---|
| | **Precision** | **Recall** | **F1** | **Precision** | **Recall** | **F1** |
| Run-1 (Hybrid + SentSeg) | 73.82 | 79.43 | 76.52 | 43.32 | 82.94 | 56.91 |
| Run-2 (Hybrid) | 73.45 | 80.04 | 76.60 | 43.14 | 82.59 | 56.67 |
| Run-3 (Joint + SentSeg) | 73.21 | 79.56 | 76.26 | 45.28 | 81.41 | 58.19 |
| Run-4 (Joint) | 73.95 | 79.33 | 76.55 | 44.56 | 82.51 | 57.87 |
| Run-5 (Separated + SentSeg) | 73.80 | 79.46 | 76.53 | 39.39 | 83.08 | 53.45 |
| Run-6 (Separated) | 73.46 | 80.08 | 76.63 | 36.90 | 84.15 | 51.30 |

*Table 9.* Official evaluation results on test set of our six submitted runs for nested-named entity recognition

| **Run** | **Precision** | **Recall** | **F1** |
|---|---|---|---|
| Run-1 (Hybrid + SentSeg) | 77.85 | 71.08 | 74.31 |
| Run-2 (Hybrid) | 78.32 | 70.88 | 74.41 |
| Run-3 (Joint + SentSeg) | 78.07 | 70.98 | 74.35 |
| Run-4 (Joint) | 78.0 | 71.69 | **74.70** |
| Run-5 (Separated + SentSeg) | 77.83 | 70.78 | 74.14 |
| Run-6 (Separated) | 78.35 | 70.44 | 74.19 |

entities. While for F1 scores of six runs for level-1 entity recognition are very close, **Joint** method outperformed the other methods in recognizing level-2 entities.

Table 9 shows the official evaluation results for our six submitted runs. As indicated in the table, run 4 which uses **Joint** model obtained the highest F1 score among six runs. Using **Joint** method or **Hybrid** method obtained better F1 scores than using **Separated** methods. We also see that the difference between a system that performs sentence segmentation and a system that does not perform sentence segmentation is very small.

The reason why **Joint** method and **Hybrid** method obtained better F1 scores than **Separated** method is that both **Joint** an **Hybrid** methods used joint-tag model while **Separated** method used level-2 model to recognize level-2 entities. We already pointed out that joint-tag model outperforms level-2 model in level-2 entity recognition.

## 4. CONCLUSIONS

We presented a feature-based model for Vietnamese named-entity recognition and evaluation results at VLSP 2018 NER evaluation campaign. We compared several methods for recognizing nested entities. Experimental results showed that combining tags of entities at all levels for training a sequence labeling model improved the accuracy of nested named-entity recognition. As the future work, we plan to investigate deep learning methods such as BiLSTM-CNN-CRF [9] for nested named entity recognition.

## References

[1] P. F. Brown, P. V. deSouza, R. L. Mercer, V. J. D. Pietra, and J. C. Lai, "Class-based n-gram models of natural language," *Comput. Linguist.*, vol. 18, no. 4, pp. 467–479, Dec. 1992. [Online]. Available: http://dl.acm.org/citation.cfm?id=176313.176316

[2] J. R. Finkel and C. D. Manning, "Nested named entity recognition," in *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing: Volume 1 - Volume 1*, ser. EMNLP '09. Stroudsburg, PA, USA: Association for Computational Linguistics, 2009, pp. 141–150. [Online]. Available: http://dl.acm.org/citation.cfm?id=1699510.1699529

[3] N. T. M. Huyen and V. X. Luong, "VLSP 2016 shared task: Named entity recognition," in *Proceedings of Vietnamese Speech and Language Processing (VLSP)*, 2016.

[4] A. Katiyar and C. Cardie, "Nested named entity recognition revisited," in *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*. Association for Computational Linguistics, 2018, pp. 861–871. [Online]. Available: http://aclweb.org/anthology/N18-1079

[5] J. Lafferty, A. McCallum, and F. Pereira, "Conditional random fields: Probabilistic models for segmenting and labeling sequence data," in *ICML*, 2001, pp. 282–289.

[6] H. P. Le, "Vietnamese named entity recognition using token regular expressions and bidirectional inference," *CoRR*, vol. abs/1610.05652, 2016.

[7] P. Le-Hong, Q. N. M. Pham, T.-H. Pham, T.-A. Tran, and D.-M. Nguyen, "An empirical study of discriminative sequence labeling models for vietnamese text processing," in *Proceedings of the $9^{th}$ International Conference on Knowledge and Systems Engineering (KSE 2017)*, 2017.

[8] P. Liang, "Semi-supervised learning for natural language," Ph.D. dissertation, Massachusetts Institute of Technology, 2005.

[9] X. Ma and E. Hovy, "End-to-end sequence labeling via bi-directional LSTM-CNNs-CRF," in *Proceedings of the $54^{th}$ Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, vol. 1, 2016, pp. 1064–1074.

[10] P. Q. N. Minh, "A feature-rich vietnamese named-entity recognition model," *arXiv preprint arXiv:1803.04375*, 2018.

[11] D. Q. Nguyen, D. Q. Nguyen, T. Vu, M. Dras, and M. Johnson, "A fast and accurate Vietnamese word segmenter," in *Proceedings of the $11^{th}$ International Conference on Language Resources and Evaluation (LREC 2018)*, 2018.

[12] T.-S. Nguyen and L.-M. Nguyen, "Nested named entity recognition using multilayer recurrent neural networks," in *International Conference of the Pacific Association for Computational Linguistics*. Springer, 2017, pp. 233–246.

[13] N. Okazaki, "Crfsuite: a fast implementation of conditional random fields (CRFs)," 2007. [Online]. Available: http://www.chokkan.org/software/crfsuite/

[14] J. Pennington, R. Socher, and C. D. Manning, "Glove: Global vectors for word representation," in *Empirical Methods in Natural Language Processing (EMNLP)*, 2014, pp. 1532–1543. [Online]. Available: http://www.aclweb.org/anthology/D14-1162

[15] E. F. T. K. Sang, "Introduction to the conll-2002 shared task: Language-independent named entity recognition," *CoRR*, vol. cs.CL/0209010, 2002.

[16] E. F. T. K. Sang and F. D. Meulder, "Introduction to the conll-2003 shared task: Language-independent named entity recognition," in *CoNLL*, 2003.

[17] B. Sundheim, "Overview of results of the muc-6 evaluation," in *MUC*, 1995.

*Received on October 03, 2018*
*Revised on December 28, 2018*