International Journal of Innovative Research in Science and Engineering
Volume No. 12, Issue No. 03, Year 2026
ISSN : 2454-9665

Neural Network – Based Named Entity Recognition for Bodo: A Deep Learning Approach

Mr.T.Sambasiva Rao
Department of CSE,
Tirumala Engineering College

K.kalpana
kalpana2005ksr@gmail.com
M. DivyaSree
Divyasreemangisetty@gmail.com

M.Nandini Bai
nandhinibaimudavath@gmail.com
M. Tulasi ram
tulasiramchowdary9999@gmail.com

***Abstract*—Named Entity Recognition (NER) in low-resource languages like Bodo faces challenges due to limited annotated data, tools, and digital resources, but deep learning approaches such as LSTM, GRU, and CNN help overcome these issues. The system leverages data augmentation, transliteration, pre-trained embeddings, and both character- and word-level features, along with bidirectional processing and CRF for better sequence labeling and context understanding. Among the models, CNN achieved the highest accuracy (99.91%), outperforming GRU and LSTM. These methods improve generalization and highlight the importance of feature extraction and architecture design in underrepresented languages. The study demonstrates the effectiveness of deep learning for multilingual NER and supports applications like information extraction, machine translation, and question answering, setting a foundation for advancing other low-resource languages.**

***Index Terms*—Named entity recognition, bi-directional long short-term memory, convolutional neural network, conditional random field.**

I. Introduction

The task of named entity recognition (NER) is often one of the first important steps in a natural language processing pipeline. It is used in many recent applications such as machine translation, information extraction as well as question answering systems. Before the advent of deep learning, the NER task was addressed with Hidden Markov Model ([1], [2]), Conditional Random Field ([3], [4]) or hand-crafted rules ([5], [6]). In recent years, deep neural network models have already outperformed the traditional approaches and achieved state-of-the-art results. In [7] Gang Luong *et al*. proposed combined model, in which NER and linking tasks are jointly modeled to capture their mutual dependencies; and achieved 91.20% of F1 on CoNLL-2003 dataset [8]. In [9] Zhiheng Huang *et al*. used Bi-LSTM in combination with CRF model for sequence tagging and also reached a competitive tagging performance: 90.10% of F1 on CoNLL-2003 dataset. In the more recent paper [10], Emma Strubell with colleagues proposed a variant of CNN, Iterated Dilated Convolution model, to address the task of NER and also got 90.54% of F1, close to state-of-the-art performances tested on CoNLL-2003 dataset.

Modern methods solve NER task by exploiting (1) semantic content of words via vector embeddings such as Word2Vec[1], GloVe [11] or FastText[2], (2) character-level features of named entities via convolutional neural networks, (3) word order via bi-directional LSTM [12], (4) probabilistic modeling of tag sequence via CRF [13]. Another important feature is capitalization of words because a named entity often is a combination of some capitalized words in the sentence. In our work we proposed a combined model consisting of three encoding sub-networks to fully utilize semantic, sequential and character level aspects of NER task. The difference of our model from Zhiheng Huang *et al*.'s model [9] is that we supplemented CNN to extract character-level features. Our work is also close to the work of Lample *et al*. [14]. Both approaches extract character-level features and employ Bi-LSTM to capture both character-level features and word contextual representation. They directly combine capitalization features with pre-trained word embeddings. In our model, we use two sub-networks, CNN and Bi-LSTM, to capture character-level and capitalization features independently. Outputs of these sub-networks are then concatenated with pre-trained word embeddings to represent rich semantic and grammatical aspects of each input word in the sentence. We experimented our model on Vietnamese, Russian, English, and Chinese datasets and obtained state-of-the-art performances. We also demonstrated that our model well adapted to decreasing amount of the training data.

II. Combined Bi-LSTM-CNN-CRF Model

In this section, we describe step by step the way our model was built, its sub-networks and why they were employed.

A. NER Task

Sequence labeling is a generic task in the field of Natural Language Processing (NLP) which aims to assigns labels to the elements of a sequence. Typical applications include part of speech tagging, word segmentation, speech recognition, and named entity recognition. From machine learning perspective, this task can be considered as building the function f that maps an observed sequence to a sequence of labels:

$$f: x \rightarrow y, \quad (1)$$

where x and y are sequences which have the same length.

Let's X is a list of observed sequences and Y is a list of sequences of corresponding labels, we need to build a model:

Manuscript received August 12, 2018; revised November 3, 2018. This work was supported by National Technology Initiative and PAO Sberbank project ID 0000000007417F630002.

The Anh Le is with Neural Networks and Deep Learning Lab, Moscow Institute of Physics and Technology, Russia. He is also with Faculty of Information Technology, Vietnam Maritime University, Viet Nam (e-mail: anhlt@vimaru.edu.vn).

Mikhail S. Burtsev is with Neural Networks and Deep Learning Lab, Moscow Institute of Physics and Technology, Russia (e-mail: burtcev.ms@mipt.ru).

[1] https://code.google.com/archive/p/word2vec/
[2] https://fasttext.cc/

$$\theta = argmin_\theta \sum_{x \in X, y \in Y} L(y, f(x, \theta)), \quad (2)$$

where L is a loss function, θ denotes the model parameters.

In the inference stage, we need to find the sequence that maximize the conditional probability $P(y|x, \theta)$:

$$\hat{y} = argmax_y P(y|x, \theta) \quad (3)$$

B. Character Representation

In both the training and testing stages there are a lot of entities whose words are not initialized by pre-trained word embeddings, even do not exist in the word vocabulary due to limitations in building the dictionaries and pre-trained word embeddings. Such words have to be replaced by a special word (e.g., *unknown*). The prediction result for such words are often worse than the others. To deal with this issue, we use a CNN model to represent words from their characters due to ability of CNN to capture morphological information of characters in a word such as prefix and suffix ([15], [16]).

Given a character dictionary D, the character lookup table $L \in \mathbb{R}^{|D| \times d_c}$, where $|D|$ is the size of D, is used to map each character to a dense vector representation with dimension d_c. This lookup table is tuned during the training stage.

Let $X \in \mathbb{R}^{nb_w \times nb_c}$ is the input sentence. Here nb_w is the number of words in the sentence, and nb_c is number of characters in each word. The embedded sentence $E \in \mathbb{R}^{nb_w \times nb_c \times d_c}$ is created by looking up X in L.

Let $F \in \mathbb{R}^{f_h \times f_w \times c_i \times c_o}$ are filters of a convolutional layer, where f_h, f_w, c_i, c_o are filter height, filter width, number of input channels, and number of output channels, respectively. The position (i, j) on the t^{th} slice of the output is calculated as[3]:

$$O^t_{(i,j)} = \sum_{r=0}^{f_h-1} \sum_{c=0}^{f_w-1} \sum_{k=0}^{c_i-1} E_{(r_x, c_x, k)} \times F^t_{(r,c,k)} + b^t_k, \quad (4)$$

where:

$$r_x = i + r - \frac{f_h}{2} + 1, \quad (5)$$

$$c_x = j + c - \frac{f_w}{2} + 1 \quad (6)$$

In our model, we use two convolutional layers followed by a max pooling layer. Note that in the formulas and notations we omit the dimension of batch size in order to increase readability.

C. Capitalization Extraction

For the task of NER, several additional features are often used such as part of speech, character-level features, capitalization features, gazetteers. From experiments, we realized that capitalization features of words are really effective because names of persons, locations or organizations usually are combinations of several capitalized words in a sentence (e.g., "An Nhien will visit Saint Petersburg in the near future."). Our idea is to transform each sentence into its capitalization format. For instance, the sentence mentioned above will be transformed into the sequence: 2 2 1 1 2 2 1 1 1 1, where "2" denotes a word starting with a capitalized letter, and "1" is encoding of a word whose characters are all in lowercase (refer to Table I for a complete description about capitalization types we used in our implementation).

TABLE I: CAPITALIZATION TYPES OF WORD

ID	Capitalization Types	Description
0	UPPER_CASE	All characters are uppercase
1	lower_case	All characters are lowercase
2	First_Cap	The first letter is capitalized
3	Otherwise	The words that do not belong to three formats above

In our implementation, we used Bi-LSTM [12] to extract capitalization features of words in combination with their left and right contexts. The architecture of this sub-network is graphical illustrated in the Fig. 1.

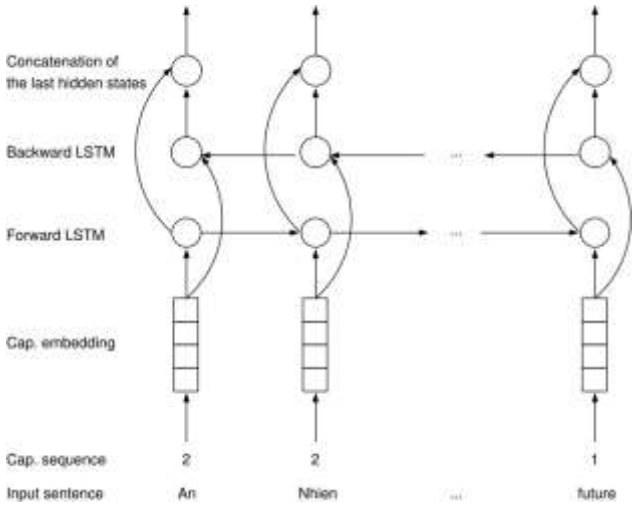

Fig. 1. The Bi-LSTM network for capitalization features extraction.

D. Combined Bi-LSTM-CNN-CRF Model

Outputs of two sub-networks mentioned above are then concatenated with the pre-trained word embedding to create a vector which represents rich semantic and grammatical aspects of the input sentence. These vectors are then fed into another Bi-LSTM network named word-contextual network (for easy of description) to capture the context of words in their sentence. The output of this word-contextual network can be directly fed into a fully connected layer followed by a softmax layer to output the probability distribution over the possible tags. However, to further improve the model performance, in our model a CRF layer [13] is applied instead of the softmax layer to exploit the implicit constraints on the order of tags. Let's O is output of word-contextual network, where $O_{i,j}$ represents score of the j^{th} tag for the i^{th} word. T is a transition matrix, where $T_{i,j}$ is the transition score from tag i to tag j. Then score of each pair of input sentence $X = (x_1, x_2, .., x_n)$ and tagging sequence $\boldsymbol{y} = (y_1, y_2, .., y_n)$ is calculated by equation below:

$$s(X, \boldsymbol{y}) = T_{\boldsymbol{y}_0, \boldsymbol{y}_1} + \sum_{i=1}^{n} (O_{i, \boldsymbol{y}_i} + T_{\boldsymbol{y}_i, \boldsymbol{y}_{i+1}}), \quad (7)$$

where $\boldsymbol{y}_0, \boldsymbol{y}_{n+1}$ are added to denote the beginning and the end of the sequence of tags.

After that, the softmax function is applied to produce

[3] In this formula the strides = (1, 1) and 'same' padding type are used.

conditional probabilities of tag sequence:

$$p(\boldsymbol{y}|X) = \frac{e^{s(X,\boldsymbol{y})}}{\sum_{\hat{y}\in \boldsymbol{Y_X}} e^{s(X,\hat{y})}}, \quad (8)$$

where $\boldsymbol{Y_X}$ is the set of all possible tag sequences for the input sentence X.

In the training stage, the log-probability of the correct sequence of tags is optimized:

$$\boldsymbol{y}^* = argmax_{\hat{y}\in \boldsymbol{Y}_X} s(X, \hat{\boldsymbol{y}}) \quad (9)$$

A graphical illustration of the completed model is provided in the Fig. 2.

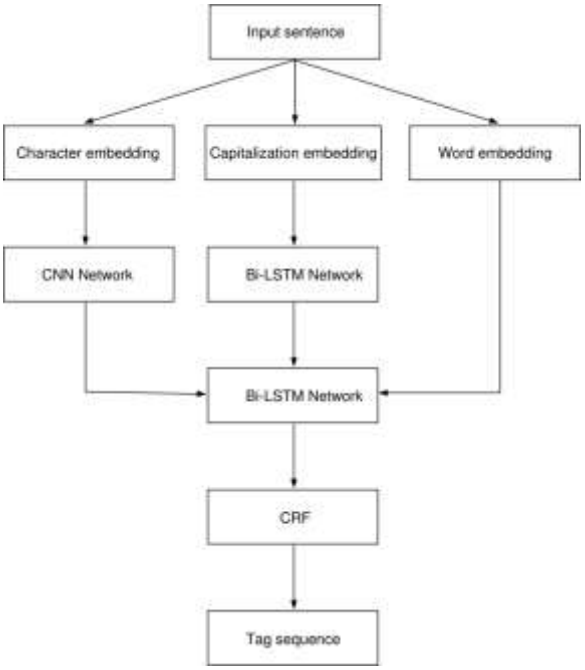

Fig. 2. The Combined Bi-LSTM-CNN-CRF Model for the task of NER.

III. Datasets and Pretrained Word Embeddings

A. Datasets

Bellow we briefly describe six datasets that were used to evaluate the model performance:

- Named Entity 5 (NE5), Named Entity 3 (NE3) [17]: Two Russian datasets published by Information Research Laboratory[4]. There are 5 different entity types in NE5 dataset: Person, Organization, Location, Media and Geopolit. NE3 is a variant of NE5 by combining Media with Organization and Geopolit with Location.
- Gareev's dataset: The Russian dataset received from Gareev *et al*. [18]. This dataset contains two entity types: Person and Organization.
- VLSP-2016: The Vietnamese dataset provided by the Vietnamese Language and Speech Processing community[5].
- CoNLL-2003 [19]: The English dataset in the shared task for NER at Conference on Computational Natural Language Learning, 2003.
- MSRA: Due to the difficulty of finding an official Chinese dataset for the task of NER, we decided to use MSRA dataset[6]. This dataset was annotated by the Natural Language Computing group within Microsoft Research Asia.

The detail statistic of all these datasets is shown in the Table II.

TABLE II: Dataset Statistic

Datasets	Per	Org	Log	Misc	Geo	Med
NER5	10623	7032	3143	-	4103	1509
NER3	10623	8541	7244	-	-	-
Gareev's	485	1311	-	-	-	-
VLSP-2016	7480	1210	6244	282	-	-
(train/test)	1294	274	1377	49		
MSRA	17610	20584	36616	-	-	-
(train/test)	1973	1330	2863			
CoNLL-2003	6600	6321	7140	3438	-	-
(train/dev/test)	1842	1341	1181	1010		
	1617	1661	1668	702		

B. Pretrained Word Embeddings

In our experiments the following pre-trained word embeddings were used to initialize word lookup tables:

- Glove6B100d [7] : The English pre-trained word embedding developed by Jeffrey Pennington, Richard Socher, Christopher D. Manning.
- Lenta: The Russian pre-trained word embedding we created using fastText[8] to train on Lenta corpus[9].
- Word2vecvn_2016 [20]: Vietnamese word embedding published by Xuan-Son Vu [20].
- Wiki_100.utf8: The Chinese pre-trained word embedding; available download at: https://github.com/zjy-ucas/ChineseNER.

IV. Experiments

Our experiments were performed on GPU NVIDIA GeForce GTX 1080Ti. The training times on each dataset took about from 1 to 3 hours.

Labeling schemes used in datasets mentioned in the previous section are IOB and IOBES. To evaluate the performance of our model, we use conlleval script, an evaluation program given in the shared task of CoNLL-2003 conference[10], in which F-measure are calculated by bellow formula:

$$F_1 = \frac{2\times P\times R}{P+R}, \quad (10)$$

where P, R, F denote precision, recall, and F-measure, respectively.

Our experiments are divided into three groups:

- Run the full model on six datasets to evaluate the ability of our model to generalize to different languages.
- Experiment the variants of our model on three datasets:

[4] http://labinform.ru/pub/named_entities/descr_ne.htm
[5] http://vlsp.org.vn
[6] https://www.microsoft.com/en-us/download/details.aspx?id=52531
[7] https://nlp.stanford.edu/projects/glove/
[8] https://fasttext.cc/
[9] https://github.com/yutkin/lenta.ru-news-dataset
[10] https://www.clips.uantwerpen.be/CoNLL-2003/

VLSP-2016, CoNLL-2003 and Gareev's dataset to analyze effect of input features on the model performance in different languages.

- Train the full model with small amounts of training data to see how well the model adapts to the decreasing in the training data.

In the first group, firstly, we tested our model on two Russian datasets: Named Entity 5, Named Entity 3. These datasets are divided into three parts for training, validation and testing in the ratio 3:1:1. Achieved results are shown in the Table III. After that, we tested our model on Gareev's dataset using k-fold cross validation because of small size of this dataset (See Table IV). This result is not really as high as we expected. To further improve the performance of the model, we decided to use the model trained on Named Entity 3 dataset as pre-trained model to train on Gareev's dataset. This helped our model increase the prediction accuracy by about 3%. More details of this experiment are shown in the Table V.

TABLE III: TAGGING PERFORMANCE ON NE3 AND NE5

Dataset	M.	Per	Org	Loc	Geo	Med	Overall
NE5	P	97.13	90.35	93.92	95.89	90.06	94.33
	R	98.43	91.75	91.67	98.08	90.06	95.29
	F	97.78	91.04	92.78	96.97	90.06	94.81
NE3	P	98.12	93.08	96.19	-	-	95.95
	R	98.58	94.14	97.68	-	-	96.88
	F	98.35	93.60	96.93	-	-	96.41

TABLE IV: TAGGING RESULTS ON GAREEV'S DATASET USING K-FOLD CROSS VALIDATION

Metric	Fold 1	Fold 2	Fold 3	Fold 4	Fold 5	Overall
P	88.11	89.66	88.30	86.02	83.24	87.07
R	93.42	89.66	89.61	91.32	88.00	90.40
F	90.69	89.66	88.95	88.59	85.56	88.69

TABLE V: TAGGING RESULTS ON GAREEV'S DATASET AFTER TRAINING ON NE3

Metric	Fold 1	Fold 2	Fold 3	Fold 4	Fold 5	Overall
P	96.93	88.53	88.20	90.83	87.47	89.73
R	92.60	91.73	93.18	91.60	93.71	92.56
F	93.11	90.10	90.62	91.21	90.48	91.10

Next, we tested our model on VLSP-2016 and CoNLL-2003 datasets. These datasets contain two additional features: POS and Chunk. We experimented our model in both cases: with and without using POS and Chunk features. The tagging results on these datasets are shown in the Tables VI, VII. Tables VII, IX show our results in comparison with cutting-edge models for Vietnamese and English. Our model outperforms previously state-of-the-art models for the task of Vietnamese NER. Besides that, our result on CoNLL-2003 is very close to Wang *et al.*'s result [21].

The last experiment in the first group is to test our model on a Chinese dataset. Due to difficultly finding an official Chinese dataset, we choose MSRA. This dataset is annotated by the Natural Language Computing group within Microsoft Research Asia. From our view, Chinese language is more complicate than English due to the lack of word boundary. Therefore, we decided to employ word segmentation as an input feature instead of the capitalization feature we mentioned before. The performance of our model on MSRA dataset are shown in the Table X.

TABLE VI: TAGGING RESULT ON VLSP-2016

Features	Metric	Per	Org	Misc	Loc	Overall
word + char. + cap.	P	95.35	73.79	100	88.58	90.61
	R	91.96	55.47	79.59	89.41	87.25
	F	93.63	63.33	88.64	88.99	88.90
word + char. + cap. + pos + chunk	P	96.43	90.17	100	94.15	94.91
	R	95.98	77.01	87.76	95.65	93.96
	F	96.20	83.07	93.48	94.89	94.43

TABLE VII: TAGGING RESULT ON CONLL-2003

Features	Metric	Per	Org	Misc	Loc	Overall
word + char. + cap.	P	97.75	90.16	77.50	89.74	90.44
	R	94.12	86.90	83.98	94.16	90.76
	F	95.90	88.50	80.61	91.90	90.60
word + char. + cap. + pos + chunk	P	97.10	90.01	80.61	90.35	90.91
	R	95.24	88.16	83.41	94.65	91.52
	F	96.16	89.08	81.99	92.45	91.22

TABLE VIII: TAGGING PERFORMANCE ON VLSP-2016 COMPARED WITH SOME STATE-OF-THE-ART MODELS

Model	P	R	F
Pham *et al.* (2017) [22]	91.09	93.03	92.05
Pham *et al.* (2017) [23]	92.76	93.07	92.91
(Ours)	**94.91**	**93.96**	**94.43**

TABLE IX: TAGGING PERFORMANCE ON CONLL-2003 COMPARED WITH SOME STATE-OF-THE-ART MODELS

Model	P	R	F
Zhiheng Huang *et al.* (2015) [9]	-	-	90.10
Strubell *et al.* (2017) [10]	-	-	90.54
Passos *et al.* (2014) [24]	-	-	90.90
Lample *et al.* (2016) [14]	-	-	90.94
Gang Luo *et al.* (2015) [7]	91.50	91.40	91.20
Wang *et al.* (2017) [21]	**91.39**	91.09	**91.24**
(Ours)	90.91	**91.52**	91.22

TABLE X: TAGGING RESULT ON MRSA DATASET

Metric	Per	Org	Log	Overall
P	91.18	89.85	93.51	91.99
R	94.41	91.62	94.82	93.92
F	92.77	90.73	94.16	92.95

To analyze effect of input features on the model performance in different languages we tested four variants:

- Baseline: Word Bi-LSTM + CRF
- Baseline + Character CNN
- Baseline + Character CNN + Capitalization Bi-LSTM
- Baseline + Character CNN + Capitalization Bi-LSTM + Pos, Chunk features.

The original dataset received from Gareev *et al.* did not include Pos and Chunk features. We, therefore, had to use the third-party system, UDPipe[11], to generate POS feature for Gareev's dataset. The experimental results showed that the Char. CNN sub-network helped to significantly boost the model performance: about 12%, 5% and 15% for VLSP-2016, CoNLL-2003 and Gareev's datasets, respectively. The character CNN sub-networks are very useful in the case of small training data or the large number of unknown words. This is pointed out in the experiments on VLSP-2016 and Gareev's datasets. Besides that, the enhancement by about 3% of F1 was obtained by applying the Cap. Bi-LSTM sub-network. One more interesting finding from this

[11] The tagging system wrote by Milan Straka and Jana Straková at Institute of Formal and Applied Linguistics, Charles University, Czech Republic

experiment was that adding pos and chunk features made the big improvement of the model's performance on VLSP-2016: about 5%, whereas the change was negligible on CoNLL-2003 dataset. This partly showed that the syntactic features in Vietnamese play a more important role than in English in the context of NER task. See Fig. 3 for more details.

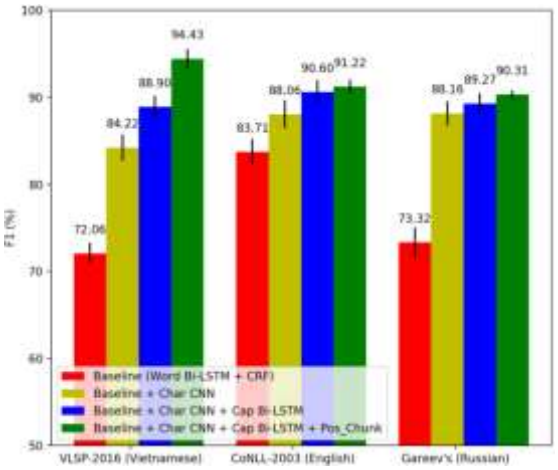

Fig. 3. Tagging performance of variants of the model across the datasets.

In the final test, we evaluated the performance of the model when training on small amounts of training data. To do this, we created five pairs of training and development sets which contain 100, 200, 500, 800 and 1000 entities per each type. The ratio of entities between training set and development set was 4:1.

First, we tested the full model on four datasets (See Fig. 4). The experimental results pointed out that our model can obtain an acceptable performance (about 70% of F1) on almost given datasets with only 80 samples for training and 20 samples for validation. When the number of samples is increased to 1000, our model nearly yields near best performances. The low result on MSRA dataset can be explained by the features we used to train, the character-level feature and word segmentation, and the complexity of Chinese language as mentioned in the Section III.

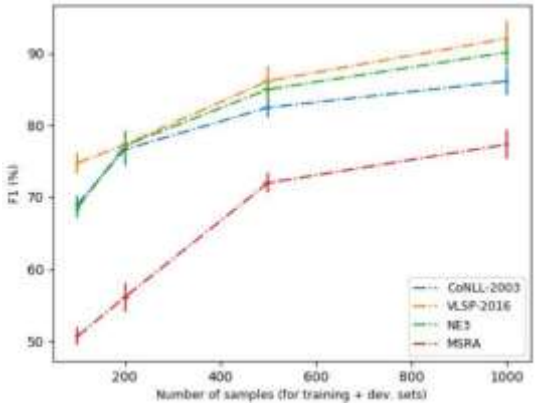

Fig. 4. Tagging performance with the different amounts of training data across the datasets.

Second, we tested variants of the model on CoNLL-2003 dataset. The average values of F1 are shown in Fig. 5. It is easy to see that leveraging character embedding encoded by CNN significantly increases the model performance, especially when training on only few hundreds of samples. Besides that, using the capitalization embedding and additional features, such as pos and chunk, also helps to improves the performance, but the improvement was not really impressive.

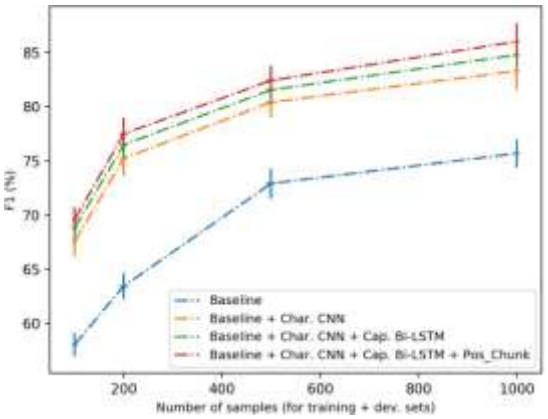

Fig. 5. Tagging performance of variants of the model on CoNLL-2003 dataset with different amounts of training samples.

V. Conclusions and Future Works

Character-level, capitalization and word contextual features are key input features for the task of NER. The word contextual feature is the main feature that is exploited in almost all deep learning-based NER systems. In our model, to extract this feature we use Bi-LSTM network that has the ability to capture both left and right contexts of each input word. A pre-trained word embedding is used to initialize the word embedding in order to reduce the training time and partly improve the model performance. In the decoding stage, using CRF model is absolutely better than just applying the softmax function due to the ability of CRF model to make global decisions that depend on not only representation vectors of input words but also the linear dependencies between tagging decisions. Our baseline model (the red bars in the Fig. 3) achieved about 72% of F1 on all datasets. Besides, using character-level features significantly improves the tagging accuracy, especially in the case that the input word does not exist in the word dictionary and in the training and development sets. In our model, we use a CNN network to capture character-level features due to its fast-speed compared with Bi-LSTM network. A named entity is often a combination of several words starting with upper-case letters. Therefore, using capitalized sequences converted from raw input sentences can increase the tagging accuracy. This increasing is more or less heavily depending on language characteristics. In addition to above key features, POS and Chunk also are good features for the task of NER. In the experiments on Vietnamese and English datasets, we concatenated these features with the word representation vector. This helped to increase a little bit on the CoNLL-2003 dataset, but remarkable on the VLSP-2016 dataset.

In conclusion, in this paper, we proposed a deep hybrid neural network model that uses three sub-networks to fully exploit the key input features, followed by a CRF layer to capture the implicit constraints on the order of output tags. Our experiments showed that the model generalizes to different languages and obtains state-of-the-art performances on Vietnamese, English and Russian datasets. Besides that, our model still remains a good performance even with small

amounts of training data.

One of the drawbacks of building deep neural network model is difficulty in making a large enough dataset for training. In some special domain, this work is almost unfeasible. Because of this reason, our future works tend to build NER models with small training datasets. We hope to create a cutting-edge model that obtains state-of-the-art performance by training on only several hundreds of samples. One of our ideas is to combine language modeling with the task of NER to share the hidden representation layer in the encoding module. Firstly, the parameters in the encoding module are adjusted by training the language modeling task with large-scaled corpus (crawled from Wikipedia, for example). After that, the model will be trained on a small dataset for the task of NER with supporting of the transfer learning technique. Besides that, the character embedding can be calculated directly from a pre-trained word embedding[12]. If this idea succeeds, we will easily apply the model to any specific domain without having to worry about building a large-scale dataset for training.

ACKNOWLEDGEMENT

The statement of author contributions. AL conducted initial literature review, proposed and implemented the model, collected and preprocessed datasets, run experiments under supervision of MB. AL drafted the first version of the paper. MB edited and extended the manuscript.

This work was supported by National Technology Initiative and PAO Sberbank project ID 0000000007417F630002.

REFERENCE

[1] S. Narzary, A. Brahma, S. Nandi, and B. Som, "Deep Learning based Named Entity Recognition for the Bodo Language," Procedia Computer Science 235, 2405– 2421,2024.

[2] R. Cotterell and K. Duh, "Low-Resource Named Entity Recognition with CrossLingual, Character-Level Neural Conditional Random Fields," arXiv preprint, arXiv:2404.09383, 2024.

[3] R. Nath, and B. Mahanta," Data Augmentation for Bodo Named Entity Recognition," International Journal of Data Science and Analysis 9(2),2023.

[4] M. Sireesha, Srikanth Vemuru, and S. N. TirumalaRao, "Coalesce based binary table: an enhanced algorithm for mining frequent patterns," International Journal of Engineering and Technology, vol. 7, no. 1.5, pp. 51–55, 2018.

[5] S. Torge, A. Politov, C. Lehmann, B. Saffar, and Z. Tao, "Named Entity Recognition for Low-Resource Languages - Profiting from Language Families," in Proceedings of the 9th Workshop on Slavic Natural Language Processing (SlavicNLP 2023), Association for Computational Linguistics, pp. 1–10, 2023.

[6] R. Alfred, L. C. Leong *et al*., "A rule-based named-entity recognition for Malay articles," *Advanced Data Mining and Applications, Lecture Notes in Computer Science*, Berlin, Heidelberg, vol. 8346, pp. 288-299, 2013.

[7] G. Luo, X. Huang, C. Lin, and Z. Nie, "Joint named entity recognition and disambiguation," in *Proc. the 2015 Conference on Empirical Methods in Natural Language Processing*, pp. 879-888, 2015.

[8] E. F. T. K. Sang and F. D. Meulder, "Introduction to the conll-2003 shared task: Language-independent named entity recognition," in *Proc. CoNLL-2003*, Edmonton, Canada, pp. 142-147, 2003.

[9] Z. Huang, W. Xu, and K. Yu. (August 2015). *Bidirectional LSTM-CRF Models for Sequence Tagging*. [Online]. Available: https://arxiv.org/abs/1508.01991

[10] E. Strubell, P. Verga *et al*., "Fast and accurate entity recognition with iterated dilated convolutions" in *Proc. the 2017 Conference on Empirical Methods in Natural Language Processing*, pp. 2660-2670, 2017.

[11] J. Pennington *et al*., "Glove: Global vectors for word representation," in *Proc. the 2014 Conference on Empirical Methods in Natural Language Processing*, pp. 1532-1543, 2014.

[12] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Journal Neural Computation*, vol. 9, issue 8, pp. 1735-1780, 1997.

[13] C. Sutton and A. McCallum, "An introduction to conditional random fields," *Foundations and Trends in Machine Learning*, vol. 4, no. 4, pp. 267-373, 2012.

[14] G. Lample, M. Ballesteros, S. Subramanian, K. Kawakami, and C. Dyer, "Neural architectures for named entity recognition," in *Proc. the 2016 Conference of the North American Chapter of the Association for Computational Linguistics*, pp. 260-270, 2016.

[15] X. Zhang, J. Zhao, and Y. LeCun. (September 2015). *Character-level Convolutional Networks for Text Classification*. [Online]. Available: https://arxiv.org/abs/1509.01626

[16] Y. Kim, Y. Jernite, D. Sontag, A. M. Rush. (August 2015). *Character-Aware Neural Language Models*. [Online]. Available: https://arxiv.org/abs/1508.06615

[17] V. Mozharova and N. Loukachevitch, "Two-stage approach in russian named entity recognition," in *Proc. International FRUCT Conference on Intelligence, Social Media and Web*, 2016.

[18] R. Gareev, M. Tkachenko, V. Solovyev, A. Simanovsky, and V. Ivanov, "Introducing baselines for Russian named entity recognition," *Computational Linguistics and Intelligent Text Processing*, Springer, Berlin, Heidelberg, pp. 329-342, vol. 7816, 2013.

[19] E. F. T. K. Sang and F. D. Meulder, "Introduction to the CoNLL-2003 shared task: Language-independent named entity recognition," in *Proc. Conference on Computational Natural Language Learning*, pp. 142–147, 2003.

[20] X.-S. Vu. Pre-trained Word2vec models for Vietnamese. [Online]. Available: https://github.com/sonvx/word2vecVN, 2016

[21] C. Wang, W. Chen, and B. Xu, "Named entity recognition with gated convolutional neural networks," *Chinese Computational Linguistics and Natural Language Processing Based on Naturally Annotated Big Data*, CCL 2017, NLP-NABD 2017, Springer, Cham, vol. 10565, pp. 110-121, 2017.

[22] T.-H. Pham and L.-H. Phuong, "The importance of automatic syntactic features in Vietnamese named entity recognition" in *Proc. 31st Pacific Asia Conference on Language, Information and Computation*, Cebu City, Philippines, pp. 97-103, 2017.

[23] T.-H. Pham, X.-K. Pham, T.-A. Nguyen, and L.-H. Phuong, "NNVLP: A neural network-based vietnamese language processing toolkit," in *Proc. 8th International Joint Conference on Natural Language Processing*, Taipei, Taiwan, 2017.

[24] Passos, V. Kumar, and A. McCallum, "Lexicon infused phrase embeddings for named entity resolution," in *Proc. the Eighteenth Conference on Computational Language Learning*, Baltimore, Maryland USA, pp. 78-86, 2014.

.

[12] http://minimaxir.com/2017/04/char-embeddings/