# Effectiveness of Character Language Model for Vietnamese Named Entity Recognition

**Xuan-Dung Doan, Trung-Thanh Dang**
Viettel Cyberspace Center
{dungdx4,thanhdt13}@viettel.com.vn

**Le-Minh Nguyen**
Japan Advanced Institute of Science and Technology
nguyenml@jaist.ac.jp

## Abstract

Recently, many studies indicate that character language model can capture syntactic-semantic word features, resulting in state-of-the-art performance in typical NLP sequence labeling tasks. This paper shows the effectiveness of character language model for Vietnamese Named Entity Recognition by comparing several methods. We evaluate the proposed model on the VLSP 2016 dataset and our own VTNER dataset. Experimental results show that our model is the current state-of-the-art end-to-end obtains for the task.

## 1 Introduction

Named-entity recognition (NER) is the task of automatically identifying and classifying elements of the document into several categories, such as organization, person, location, currency, time, etc. NER is used in data mining systems, text summarization, question answering, translation, etc. Most methods for NER are based on machine learning.

For example, given the following sentence as input:

> Đại học Bách khoa Hà Nội nằm trên đường Đại Cồ Việt. (The Hanoi University of Science and Technology is on Dai Co Viet street.)

The output is:

> [Đại học Bách khoa Hà Nội]$_{\text{ORGANIZATION}}$ nằm trên [đường Đại Cồ Việt]$_{\text{LOCATION}}$.

Many methods for NER are based on supervised learning. In particular, Conditional Random Field - CRF (Lafferty et al., 2001; Sutton and McCallum, 2006) and Long Short Term Memory - LSTM (Hochreiter and Schmidhuber, 1997) are the most popular methods.

(Le, 2016) combined regular expressions over tokens and a bidirectional inference method in a sequence labeling model. (Pham and Le, 2017) combined Bi-LSTM, CNN and CRF that achieved the same performance with (Le, 2016). This system is the end-to-end architecture that required only word embeddings. After that, (Pham and Le, 2017) system surpassed both (Le, 2016) and end-to-end (Pham and Le, 2017) systems by using Bi-LSTM with automatically syntactic to present a state-of-the-art named entity recognition system for the Vietnamese language. Minh (2018) showed the effectiveness of rich features on CRF methods by using default features for CRF with POS and chunking tags and achieved best results on F1 score.

Recently, Contextualized word embeddings (Peters et al., 2017; Peters et al., 2018) capture word sematics in context to address the polysemous and context-dependent nature of words. They report new state-of-the-art results for NER but this approach require a larger model, external corpus and time-consuming training. (Liu et al., 2018) proposed a sequence labeling framework, LM-LSTM-CRF, and (Akbik et al., 2018) suggested Contextual String Embeddings, both of which achieved state-of-the-art results in English datasets. Their model leverages both word-level and character-level knowledge.

Thus, we wanted to implement the methods: CRF,


32nd Pacific Asia Conference on Language, Information and Computation
Hong Kong, 1-3 December 2018

LSTM-CRF and checked the effectiveness of handcrafted features on the models. Besides, we used the character language model that (Liu et al., 2018) and (Akbik et al., 2018) proposed, on the VLSP dataset (Nguyen and Vu, 2016) and our VTNER dataset.

Contributions: We overview the methods for Vietnamese Named Entity Recognition. We indicate the effectiveness of character language model in named entity recognition. We make our VTNER dataset publicly available to all community researchers.

## 2 Methodology

### 2.1 Conditional Random Field

As proposed by (Lafferty et al., 2001), (Sutton and McCallum, 2006), CRF is a popular method for sequence labeling.

With X, Y as random vectors, $\theta = \lambda_k \epsilon R^K$ is a parameter vector, $f_k(y, y', x_t)_{t=1}^K$ is a set of feature function values. Linear-chain CRF model calculates the probability $p(y|x)$:

$$p(y|x) = \frac{1}{Z(x)} exp \sum_{k=1}^{K} \lambda_k f_k(y_t, y_{t-1}, x_t) \quad (1)$$

where Z(x) is a normalization function.

Estimation $\theta = \lambda_k$ is calculated by maximum log-likelihood. Log-likelihood of probability $p(y|x)$ is calculated by:

$$l(\theta) = \sum_{i=1}^{N} logp(y^{(i)}|x^{(i)}) \quad (2)$$

After estimating $\theta$, the inference phase is run by performing Viterbi algorithm.

### 2.2 Long Short Term Memory

Recurrent Neural Network (RNN) (Goller and Kuchler, 1996) can summarize semantic sentences in lower-dimension vectors. Given a sequence input $x_1, x_2, \ldots, x_T$, a RNN calculate:

$$h_i = f(h_{i-1}, x_i), i = 1, ..., T \quad (3)$$

where $h_i$ identifies a hidden state of sequence after observation $x_i$. Long Short Term Memory (LSTM) (Hochreiter and Schmidhuber, 1997) encodes long sentence and handles RNN's vanishing gradient problem. An LSTM unit is defined as:

$$[f_t, i_t, o_t] = \sigma(W[h_{t-1}, x_t] + b) \quad (4)$$

$$l_t = tanh(V[h_{t-1}, x_t] + d) \quad (5)$$

$$c_t = f_t c_{t-1} + i_t l_t \quad (6)$$

$$h_t = o_t tanh(c_t) \quad (7)$$

where $c_t$ is a memory cell and $f_t$, $i_t$, $o_t$ are forget gate, input gate and output gate respectively. A popular RNN network is the bidirectional network (BRNN), which can summarize information bidirectionally. Besides BRNN, character level and word level are created in the prediction model. The final layer is defined as:

$$\widehat{a}_q = \sigma(W_{dec}.\phi(W_{fushion}[\overrightarrow{h_T}, \overleftarrow{h_0}])) \quad (8)$$

where $W_{fusion}$ is unified bidirectional RNN state, $\sigma$ is a sigmoid function, $\phi$ is a non-linear function, $\widehat{a}_q$ is the prediction output.

Finally, we minimize logistic error to optimize our network.

### 2.3 LSTM-CRF

(Lample et al., 2016) proposed LSTM-CRF model as joint model LSTM and CRF. The result is better than LSTM model and CRF model. The idea uses Viterbi inference after the final layer in LSTM model.

After that, (Ma and Hovy, 2016) proposed Bidirectional LSTM-CNN-CRF model as an end-to-end sequence labeling model. The result is better than LSTM-CRF. The idea uses CNN layer in character embedding.

Recently, (Liu et al., 2018) proposed an effective sequence labeling framework, LM-LSTM-CRF. They incorporated a neural language model with the sequence labeling task and conduct multi-task learning to guide the language model towards task specific key knowledge. They combined CRF model and neural language model to create a joint object function:

$$J = -\sum_i (p(y_i|Z_i) + \lambda(logp_f(x_i) + logp_r(x_i)) \quad (9)$$

where $p(y_i|Z_i)$ is the probability calculated by the CRF layer, $p_f(x_i)$ is the prediction probability for words by taking the character sequence as inputs from left to right. $p_r(x_i)$ is the prediction probability for words by taking the character sequence as inputs from right to left.

### 2.4 Proposed Method

We compared CRF model, LSTM-CRF model on VLSP 2016 dataset and VTNER dataset. We checked the effectiveness of each feature for NER accuracy in each model. In particular, we integrated Character language model that (Liu et al., 2018) and (Akbik et al., 2018) proposed, with our system (LM-LSTM-CRF).

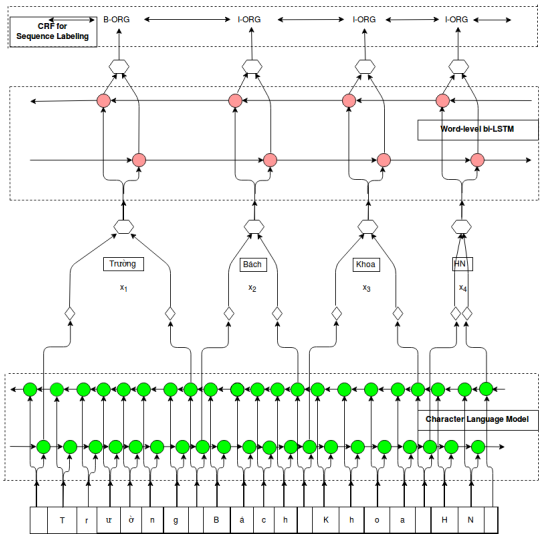


Figure 1: LM-LSTM-CRF Neural Architecture

Given input $x = (x_1, x_2, \ldots, x_T)$ and output's annotations $y = (y_1, y_2, ..., y_T)$. Its character-level input is recorded as $c = (c_{0,\text{-}}, c_{1,1}, c_{1,2}, ..., c_{1,\text{-}}, c_{2,1}, ..., c_{n,\text{-}})$, where $c_{i,j}$ is the j-th character for word $w_i$ and $c_{i,\text{-}}$ is the space character after $w_i$. By training a language model, we learn $P_f(x_i|c_{0,\text{-}}, ..., c_{i-1,\text{-}})$, an estimate of the predictive distribution over the next word given previous characters.

$$P_f(x_i|c_{0,\text{-}}, ..., c_{i-1,\text{-}}) = softmax(Vf_t + b) \quad (10)$$

where $f_t$ represents the entire previous character sequence from left to right. $V$ and $b$ are weights and biases parameters.

We also adopted a reversed-order language model, which calculated the generation probability from right to left $P(x_i|c_{i+1,\text{-}}, ..., c_{n,\text{-}})$, to extract knowledge in both directions.

$$P(x_i|c_{i+1,\text{-}}, ..., c_{n,\text{-}}) = softmax(Vr_t + b) \quad (11)$$

where $r_t$ represents the entire previous character sequence from right to left.

The results show the effectiveness of our model on both datasets.

### 2.5 Data

To evaluate our system, we used the VLSP 2016 dataset, in which we used 80% of the data as a training set, and the remaining as a testing set. We used 10% of the training set as a development set when training.

In addition, we created our own VTNER dataset following the annotation guide from VLSP 2018 organization. The dataset consists of articles crawled from a popular online news website VnExpress[1]. We used VnTokenizer tool[2] to determine word segmentation and VnTagger tool[3] to determine POS tagging which is a noun, adjective or verb. Three annotators were asked to perform annotations independently. Each person annotated three files with different number of sentences. The number of sentences in the first file, the second file and the third file is 1000 sentences, 2000 sentences and 3000 sentences. After the annotation process was completed, we asked each annotator to check another annotator's files. Finally, another expert annotator was asked to check all datasets. The datasets have nine files.

- a1.conll, b1.conll, c1.conll (each file contains about 1000 sentences)
- a2.conll, b2.conll, c2.conll (each file contains about 2000 sentences)
- a3.conll, b3.conll, c3.conll (each file contains about 3000 sentences)

The development set contains a1.conll, b1.conll, c1.conll. We use 3-fold cross-validation with 3 test

[1]https://vnexpress.net/
[2]http://mim.hus.vnu.edu.vn/dsl/tools/tokenizer
[3]http://mim.hus.vnu.edu.vn/dsl/tools/tagger

sets: a3.conll, b3.conll, c3.conll. The training set contains a2.conll, b2.conll, c2.conll and 2 of 3 files a3.conll, b3.conll, c3.conll.

Table 1: VTNER dataset

| | Number Document | Number sentence | Number entity | | | |
|---|---|---|---|---|---|---|
| | | | PER | LOC | ORG | MISC |
| Total | 990 | 20509 | 5041 | 11948 | 6912 | 914 |

We use F1 score to measure the performance.

$$F1 = \frac{2 * P * R}{P + R} \tag{12}$$

where Precision (P) is the percentage of named entities found by the learning system that is correct. Recall (R) is the percentage of named entities present in the corpus that is found by the system.

## 3 Experiments

### 3.1 Experimental settings

We implemented CRF model with features:

- Word and neighbor word
- POS tags
- Word and neighbor word are in Vietnamese dictionary
- Word is a person name: first name, mid name, last name
- Word and neighbor word is a location name
- Capital feature: the first character is capitalization, all character is capitalization
- Word is punctuation and special character.
- Word is the first word in a sentence.

We used a window size for neighbor word of 7 (three words after and three words before). We used CRFsuite[4] to implement the model. We experimented CRF without POS tags (CRF-without tag), CRF with the window size of 3 (CRF-window 3), CRF with the window size of 5 (CRF-window 5). In addition, we used the CRF with Brown cluster (CRF-with Brown) which is published in Minh (2018). Brown cluster is created by performing on 6.3 G segment text.

We also used the LSTM-CRF model with features as follows,

- LSTM-CRF without Character (LSTM-CRF-not char)
- LSTM-CRF with Capital feature (LSTM-CRF-cap)
- LSTM-CRF with POS tagging (LSTM-CRF-pos)
- LSTM-CRF with Capital feature and POS tagging feature with 100 dimensions (LSTM-CRF-cap-pos-100)
- LSTM-CRF with Capital feature and POS tagging feature with 30 dimensions (LSTM-CRF-cap-pos-30)
- LSTM-CRF loads the embedding matrix (300 dimensions) and Capital feature and POS tagging feature with 30 dimensions (LSTM-CRF-cap-post-emb-300)
- LSTM-CRF loads the embedding matrix (100 dimensions) and Capital feature and POS tagging feature with 30 dimensions (LSTM-CRF-cap-post-emb-100)
- LSTM-CRF loads the embedding matrix (100 dimensions) and Capital feature and POS tagging feature with 30 dimensions and chunking feature (LSTM-CRF-cap-post-emb-chunk)

We used Glove[5] pre-trained word embedding released by Stanford on 6.3G segment text.

Optimization: We employed mini-batch stochastic gradient descent (SGD) with a learning rate of 0.01 and a gradient clipping of 5.0. We set the dropout rate to 0.5.

We used the embedding matrix with 100 dimensions on LM-LSTM-CRF for VLSP 2016 and our VTNER dataset.

Specially, we used LM-LSTM-CRF model which is proposed by (Liu et al., 2018). This model used highway layers (Srivastava et al., 2015) and the co-training strategy.

[4]http://www.chokkan.org/software/crfsuite/

[5]https://nlp.stanford.edu/projects/glove

### 3.2 Results

We show the results of VLSP 2016 datasets and VTNER dataset.

**VLSP 2016 dataset**

Table 2: CRF models on VLSP 2016 datasets

| | F1 |
|---|---|
| CRF | 86.21 |
| CRF-without tag | 84.12 |
| CRF-window 3 | 86.43 |
| CRF-window 5 | 85.25 |
| CRF-brown | **87.96** |

Table 3: LSTM-CRF models on VLSP 2016 datasets

| | F1 |
|---|---|
| LSTM-CRF | 87.33 |
| LSTM-CRF-not char | 81.15 |
| LSTM-CRF-cap | 87.34 |
| LSTM-CRF-pos | 89.39 |
| LSTM-CRF-cap-pos-100 | 89.36 |
| LSTM-CRF-cap-pos-30 | 88.12 |
| LSTM-CRF-cap-pos-emb-300 | 90.13 |
| LSTM-CRF-cap-pos-emb-100 | 90.58 |
| LSTM-CRF-emb-cap-pos-chunk | **94.56** |
| LM-LSTM-CRF | 91.89 |
| LM-LSTM-CRF-highway-co-training | 92.17 |

Table 4: The SOTA models on VLSP 2016 datasets

| | F1 |
|---|---|
| vitk (Le, 2016) | 89.66 |
| end-to-end (Pham and Le, 2017) | 88.59 |
| vie-ner-lstm (Pham and Le, 2017) | 92.05 |
| feature-rich (Minh, 2018) | **93.93** |

- CRF models are showed in Table 2
  CRF model which contains POS tags feature and the window size of 7 gets the best score. POS tags feature increases F1 result by 2%. CRF with POS tags and Brown cluster feature gets the best score.

- LSTM-CRF models are showed in Table 3
  The results indicate that LSTM-CRF models' scores are higher than CRF models'. POS tags feature increases F1 score by 2%. POS tags feature with 100 dimensions score is higher than 300 dimensions. LSTM-CRF model load pre-train embedding matrix gets better score than LSTM-CRF model. LSTM-CRF model with 100 dimensions pre-trained embedding gets better score than the model with 300 dimensions. LSTM-CRF model with pre-trained embedding, capital feature, POS tags feature and chunking feature gets the best score (94.56% F1).
  The results of LM-LSTM-CRF model are only lower than LSTM-CRF models' with pre-trained embedding, capital feature, POS tags feature and chunking feature. But LM-LSTM-CRF is the end-to-end model and handcrafted feature as POS tags and chunking are hard to apply to new tasks or domains.

- The SOTA models are showed in Table 4
  We compared with the SOTA models on VLSP 2016. LM-LSTM-CRF scores are higher than Vitk (Le, 2016), end-to-end (Pham and Le, 2017). Our system scores are lower than viet-ner-lstm (Pham and Le, 2017), feature-rich (Minh, 2018) because our system is the end-to-end model and the viet-ner-lstm and feature-rich models use handcrafted features including chunking feature.

**VTNER dataset**

Table 5: CRF models on VTNER datasets

| F1 | a3 | b3 | c3 |
|---|---|---|---|
| CRF | **75.74** | **85.57** | **84.83** |
| CRF-without tag | 74.21 | 84.7 | 84.72 |
| CRF-window 3 | 73.88 | 84.09 | 83.15 |
| CRF-window 5 | 74.78 | 85.41 | 84.02 |

Table 6: LSTM-CRF models on VTNER datasets

| F1 | a3 | b3 | c3 |
|---|---|---|---|
| LSTM-CRF | 86.06 | 88.46 | 89.46 |
| LSTM-CRF-cap-pos | 86.99 | 88.99 | 89.72 |
| LM-LSTM-CRF | 86.81 | 90.15 | 91.50 |
| LM-LSTM-CRF-highway-co-training | **87.38** | **90.58** | **91.92** |

The effectiveness of handcrafted features to results is same with VLSP 2016 dataset. Although, LSTM-CRF models used capital and POS tags feature, LM-LSTM-CRF's scores are higher than those of CRF and LSTM-CRF models. Besides, the LM-LSTM-CRF scores are higher than LSTM-CRF models because we didn't use chunking feature in LSTM-CRF models as on VLSP 2016 dataset. The chunking features are difficult feature in the Vietnamese language.

Especially, LM-LSTM-CRF with highway and co-training obtain the best scores. This was because highway networks transform the output of character-level layers into different semantic spaces. Beside, co-training transform the output of character-level layers into different semantic spaces for different objectives. Hence, our language model can provide related knowledge of the sequence labeling, without forcing it to share the whole feature space.

## 4 Analysis

In some cases, LM-LSTM-CRF model is better at recognizing than LSTM-CRF-cap-pos and CRF models.

Table 7: Some case studies

| Result | CRF | LSTM-CRF-cap-pos | LM-LSTM-CRF |
|---|---|---|---|
| Bác Hồ sinh ngày bao nhiêu?<br>What is Bac Ho 's birthday? | Bác Hồ - PER | | Bác Hồ - PER |
| Trấn Thành là ai?<br>Who is Tran Thanh? | Trấn Thành - ORG | | Trấn Thành - PER |
| Hương có chồng hay chưa?<br>Is Huong married? | | | Hương - PER |
| Nhà văn Hemingway là ai?<br>Who is Hemingway? | | Hemingway - PER | Hemingway - PER |
| LG là công ty gì?<br>What company is LG? | | | LG - ORG |

Table 7 shows some examples in which our system can perform more accurately than the others. In the first four examples, LM-LSTM-CRF correctly identifies all person names, while CRF and LSTM-CRF-cap-pos can only correctly identify one of four cases. In the last example, LM-LSTM-CRF correctly identifies organization name, while CRF and LSTM-CRF-cap-pos fail to do so.

## 5 Conclusion

In this paper, we carefully conducted various experimental results on named-entity recognition for Vietnamese. We also indicated which is the state of the art model for standard data. We created a VTNER dataset with 20500 sentences. The best result is our LM-LSTM-CRF model on the VTNER dataset. On VLSP 2016 dataset, LM-LSTM-CRF result is lower than LSTM-CRF model with word embedding feature, capital, POS tags and chunking feature. But chunking features and other than handcrafted features are hard for applying to new tasks or domains. The results show that LM-LSTM-CRF with highway and co-training is the current state-of-the-art end-to-end method for NER task.

We made our VTNER data containing 6439 sentences publicly available for the research community. The dataset contains three files: train.conll, dev.conll and test.conll.[6]

In future, we plan to extract and incorporate knowledge from pre-trained word-level language models which (Peters et al., 2017; Peters et al., 2018) proposed.

## References

John Lafferty, Andrew McCallum, Fernando CN Pereira 2012. Conditional Random Fields: Probabilistic Models for Segmenting and Labeling Sequence Data. *Proceedings of the 18th International Conf. on Machine Learning.*

Charles Sutton, Andrew McCallum. 2006. An introduction to conditional random fields for relational learning.

Hochreiter, S., Schmidhuber, J. 1997. Long short-term memory. *Neural computation 9(8), 1735–1780.*

Guillaume Lample, Miguel Ballesteros, Sandeep Subramanian, Kazuya Kawakami, Chris Dyer. 2016. Neural Architectures for Named Entity Recognition. *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*.

Xuezhe Ma and Eduard Hovy. 2016. End-to-end Sequence Labeling via Bi-directional LSTM-CNNs-CRF. *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (ACL).*

Liyuan Liu, Jingbo Shang, Frank F. Xu, Xiang Ren, Huan Gui, Jian Peng, Jiawei Han. 2018. Empower Sequence Labeling with Task-Aware Neural Language Model. *In Proceedings of Thirty-second AAAI Conference on Artificial Intelligence (AAAI).*

Thai-Hoang Pham, Hong-Phuong Le 2017. End-to-end Recurrent Neural Network Models for Vietnamese Named Entity Recognition. *International Conference*

[6] https://github.com/dungdx34/vtner

*of the Pacific Association for Computational Linguistics (PACLING)*.

Thai-Hoang Pham, Xuan-Khoai Pham, Tuan-Anh Nguyen, Hong-Phuong Le. 2017. NNVLP: A Neural Network-Based Vietnamese Language Processing Toolkit. *Proceedings of the 8th International Joint Conference on Natural Language Processing - System Demonstrations (IJCNLP)*.

Pham Quang Nhat Minh. 2018. A Feature-Rich Vietnamese Named-Entity Recognition Model. *arXiv preprint arXiv: 1803.04375*.

Pham Quang Nhat Minh. 2018. A Feature-Based Model for Nested Named-Entity Recognition at VLSP-2018 NER Evaluation Campaign. *arXiv preprint arXiv: 1803.08463*.

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

Nguyen Thi Minh Huyen and Vu Xuan Luong 1997. Vlsp 2016 shared task: Named entity recognition. *In: Proceedings of Vietnamese Speech and Language Processing (VLSP)*.

Liang, P. 2005. Semi-supervised learning for natural language. *PhD thesis, Massachusetts Institute of Technology (2005)*.

Pennington, J., Socher, R., Manning, C.D. 2014. Glove: Global vectors for word representation. *In: Empirical Methods in Natural Language Processing (EMNLP)*.

Okazaki, N. 2007. Crfsuite: a fast implementation of conditional random fields (crfs).

Christoph Goller and Andreas Kuchler. 1996. Learning task-dependent distributed representations by backpropagation through structure. *In Neural Networks, 1996., IEEE International Conference on, volume 1, pages 347–352. IEEE.*

Le, H.P. 2016. Vietnamese named entity recognition using token regular expressions and bidirectional inference. *Proceedings of Vietnamese Speech and Language Processing (VLSP)*.

Pham, T.H., Le, H.P. 2017. The importance of automatic syntactic features in vietnamese named entity recognition. *Proceedings of the 31st Pacific Asia Conference on Language, Information and Computation (PACLIC)*.

Akbik, Alan and Blythe, Duncan and Vollgraf, Roland. 2018. Contextual String Embeddings for Sequence Labeling. *27th International Conference on Computational Linguistics (COLING)*.

Srivastava, R. K.; Greff, K.; and Schmidhuber. 2015. Highway networks. *arXiv preprint arXiv: 1505.00387*.

Matthew E. Peters, Waleed Ammar, Chandra Bhagavatula, and Russell Power. 2017. Semi-supervised sequence tagging with bidirectional language models *In Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), pages 1756–1765, Vancouver, Canada, July. Association for Computational Linguistics*.

Matthew E. Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, Luke Zettlemoyer. 2018. Deep contextualized word representations. *The 2018 Conference of the Association for Computational Linguistics will be held in New Orleans, Louisiana, 2018*.