IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received February 27, 2019, accepted April 2, 2019, date of current version April 18, 2019.

Digital Object Identifier 10.1109/ACCESS.2019.2909641

Boosting Arabic Named-Entity Recognition With Multi-Attention Layer

MOHAMMED NADHER ABDO ALI[1], GUANZHENG TAN[1], AND AAMIR HUSSAIN[2]
[1]School of Information Science and Engineering, Central South University, Changsha 410083, China
[2]Department of Computer Science, Muhammad Nawaz Shareef University of Agriculture Multan, Multan 60000,Pakistan

Corresponding author: Guanzheng Tan (tgz@csu.edu.cn)

This work was supported in part by the National Natural Science Foundation of China under Project 61403422.

ABSTRACT Sequence labeling models with recurrent neural network variants, such as long short-term memory (LSTM) and gated recurrent unit (GRU), show promising performance on several natural language processing (NLP) problems, including named-entity recognition (NER). Most existing models utilize word embeddings for capturing similarities between words. However, they lag when handling previously unobserved or infrequent words. Moreover, the attention mechanism has been used to improve sequence labeling tasks. In this paper, we propose an efficient multi-attention layer system for the Arabic named-entity recognition (ANER) task. In addition to word-level embeddings, we adopt character-level embeddings and combine them via an embedding-level attention mechanism. The output is fed into an encoder unit with bidirectional-LSTM, followed by another self-attention layer that is used to boost the system performance. Our model achieves approximately matched F1 score of 91% on the "ANERCorpus." The overall experimental results demonstrate that our method is superior to other systems. Our approach using multi-layer attention mechanism yields a new state-of-the-art result for the ANER.

INDEX TERMS ANER, self-attention, LSTM, NLP, word embedding, character embedding.

I. INTRODUCTION

The Named-Entity Recognition (NER) task is one of the most widely used Natural Language Processing (NLP) tasks to detect named entities (NEs) within texts and categorize them into predetermined classes, such as location, time, date, number, person, and organization [1]. NER is a crucial preprocessing phase to improve the overall performance of several NLP applications. It extracts useful information from raw texts and simplifies downstream tasks, such as text clustering, machine translation, question answering, and information retrieval [2].

Arabic is a language rich in morphology and syntax. Arabic is a Semitic and the most standard language spoken in the Arab world. The language, which is used in the Middle East, the Horn of Africa, and North Africa, is also one of the five official languages used by the United Nations. In the Arab world, approximately 360 million people speak Arabic in more than 25 countries [3]. In recent years, Arabic NER (ANER) has become a challenging task because of its characteristics and peculiarities [4], and given the limited availability of annotated datasets, it has received increased attention from researchers [5].

NER is one of the basic blocks of Arabic NLP tools and applications. Although considerable advancements have been achieved in ANER research in the recent two decades, it remains challenging due to various features of the Arabic language, such as the lack of capitalization, like in English where the nouns start with a capital letter to help identify the nouns from other words. Agglutination feature where the words in Arabic may compose of one or more prefixes, root, and suffixes in several groupings, that result in an intricated morphology, such as in the word وسفيرها (and- its- ambassador), that is fragmented into a conjunction, nominal and a pronominal mention-separated by a space character و- – سفير – ها (and- ambassador- its). These words in English would be considered as a separate word which in turn make it easy to identify the words that indicate names. Another challenge is the lack of uniformity in writing styles. The Arabic language has more speech sounds than other languages, so when names in a different language transliterated to Arabic, would have many NE variants for the same name and indicate the same meaning. As an example, transliterating the word "Anqara" will produce the following variants - أنقرة ، أنغره – أنجرة thus making it more complicated.

The associate editor coordinating the review of this manuscript and approving it for publication was Shuping He.

Another challenge is the Optional Short Vowels. Most vowels in Arabic are represented by diacritics that influence the phonetic representation and result in a different meaning to the same word. The modern version of Arabic which is used nowadays ignore using diacritics which lead to lexical ambiguity. As an example, the word مؤسسة may represent two different named entity types (e.g., مُؤسِسه [a founder] a trigger word for a person name; مُؤَسسة [a corporation] an organization name. Also, lack of resources is a challenge, where there exist a huge amount of data on the web and only a few resources available for ANER which are either limited in capacity and coverage or need to obtain an expensive license [6].

ANER models have been developed by two main methods. The first method is based on handcrafted rules, particularly the NERA system [7], whereas the second method relies on statistical learning, particularly ANERsys 2.0 [8]. Each method has its advantages and disadvantages. Rule-based NER systems primarily rely on handcrafted grammatical rules acquired from linguists; hence, the maintenance of these systems is time-consuming and labor-intensive, especially when the knowledge and background of the linguists are poor. On the contrary, systems based on machine learning (ML) automatically extract model patterns pertinent to the NER task from the training set of instances; thus, they do not require in-depth language-specific knowledge. The advantage of these ML-based systems over rule-based systems is that they are adjustable and updatable with minimum cost and time, providing sufficient corpus available.

Traditional Arabic NLP research methods cannot cope with the rapid evolution of massive amounts of Arabic data on the web and the increased demand for precise and durable processing tools. Therefore, neural networks have drawn much attention in recent years, and various models have been proposed. Researchers have combined different semi-supervised learning and deep neural networks to find an optimal solution to the problem of NER and other chunking tasks [9].

With the rapid advancements in deep learning techniques, NN models have accomplished a remarkable performance on various NLP tasks, such as NER [10] and text summarization [11]. Existing deep learning models, in particular, Recurrent Neural Network (RNN), generally utilize word embeddings, that enable them to learn analogous representations for semantically similar words. This approach is a significant enhancement compared with the conventional feature-based approach, thereby overcoming the language-dependent challenge on feature selection.

Despite the popularity of neural networks, they still face certain concerns when implemented to sequence labeling problems, especially to a high morphological language, such as Arabic. The most challenging problem is dealing with the previously unseen words. The slang spellings or transliterated technical terms may result in a considerable number of out-of-vocabulary (OOV) words to exist. These words might arbitrarily be initialized to some particular values because they have no corresponding word embeddings. Thus, several misclassifications of OOV words occur in the dataset.

At present, the attention mechanism is widely studied and has achieved excellent progress in various NLP tasks, such as in [12], [13]. In the context of sequence labeling classification, the attention mechanism weighs up a token or certain high-level feature representation acquired by learning a scoring function, which enables the system to focus on the most important tokens of texts for a classification tagging.

In the present work, we use multiple attention layers to boost the ANER task. A weighted mean of all the earlier states is used as a spare input to the function that calculates the subsequent state. This approach offers a system that can potentially take care of a state-generated a number of time steps earlier. Subsequently, the last state does not require the retention of the complete information [8].

The main involvement of this study is to propose a neural network-based system for ANER by considering the NER task as a classification task. The system is built using (character and word) embeddings and multiple attention layers. The model computes the accuracy and harmonic mean F-score measure for the tokens in the dataset. The proposed system has the following enhancements that boost the recognition efficiency and accuracy: (i) it utilizes (character and word) embeddings to overcome the problem of OOV; and (ii) it uses multiple attention layers, namely, the embedding attention layer over the embedding layer and the self-attention layer that creates sentence embedding. The attention provides two advantages: in addition to the best performance often acquired, it also facilitates sequence interaction between constituencies of a sequence.

The rest of the paper is structured into five sections. Section 2 briefly describes the related work. Section 3 presents the proposed approach for NER in details. Section 4 describes the experiments. Finally, Section 5 discusses the results and conclusion.

II. RELATED WORKS

ANER systems fall into two categories, namely, rule-based and ML-based systems. Rule-based systems primarily rely on handcrafted grammatical rules acquired from linguists, thereby requiring more time and labor work for maintenance. On the contrary, ML-based models automatically extract patterns related to the NER task from the training set of examples. Neural network models fall within this category and can enhance performance and accuracy.

A wide range of previous works on the use of rule-based approaches to solve the ANER task have been published, such as in [14]–[16], as well as those on ML-based approaches, as in [2], [17]–[19]. While works based on neural networks for ANER remain limited, numerous studies on other domains, such as the English text and the English biomedical text, are available [20], [21].

Most systems developed using neural network techniques use either the Convolutional Neural Network (CNN), RNN, or a combination of both. Systems built on CNN models, which use a variant of the multilayer perceptron that is fabricated to ensure minimum preprocessing, are efficient and achieve excellent results for various NLP problems. One of the first neural labeling systems using CNNs was presented by Pinheiro and Collobert *et al.* [22]. Their system achieved good results on various labeling tasks, including NER and POS, without depending on any manual feature engineering. Zhao *et al.* [23] proposed a novel multiple label CNN (MCNN) for disease NER from the biomedical literature, using character-level, word-level, and lexicon feature embeddings. Then, several convolutional layers are stacked over the concatenated embedding. This model attains state-of-the-art performance on NCBI and CDR corpora. Other systems utilize character level features and show its effectiveness for the task of named entity recognition such as in [24], [25].

Despite the success achieved by CNN, RNNs have been introduced with the new improvement that uses their internal state (memory) to deal with sequences of inputs, making them applicable to many NLP tasks with long sequences, such as NER, POS, text summarization, or speech recognition. Many systems have been developed for NER for different languages and in various domains, such as in [26], [27]. RNN also has many variants, such as bidirectional RNN, long short-term memory (LSTM), gated recurrent unit (GRU), bidirectional LSTM, and bidirectional GRU. Various systems have been developed with different architectures and designs, such as in [28]–[30].

Recently, attention mechanism has achieved great success for its abilities to deal with long sequences and increase or decrease the level of attention to words based on their involvement to the creation of the sentence meaning, thereby resulting in more accurate predictions [30]. Many models have been developed based on the attention mechanism of various NLP tasks, such as in [24], [32]–[34].

III. METHOD

In this section, we discuss the proposed attention-based neural network system for ANER. Our model is entirely constructed on a neural network and does not necessitate extra manual extraction of features compared with conventional ML models. The proposed system is composed of five stages, namely, embedding layer, embedding attention layer, encoding, self-attention layer, and prediction. Figure 1 presents the complete architecture of our model.

We used pre-trained embeddings to acquire distributed representation of words. Character-level representations are obtained via CNNs, similar to those reported by Kim *et al.* [35]. Then, instead of merely concatenating them, an embedding-level attention technique is used to merge the two features and dynamically decide which information comes from each other. Moreover, the output of the attention level is provided as input to the encoder unit-a BLSTM that processes the embedded sequence in either direction. The encoded sequence is acquired by concatenating the forward and backward hidden states of the LSTM, which is fed to another attention layer. Finally, the prediction layer predicts each token to one of the predetermined labels.

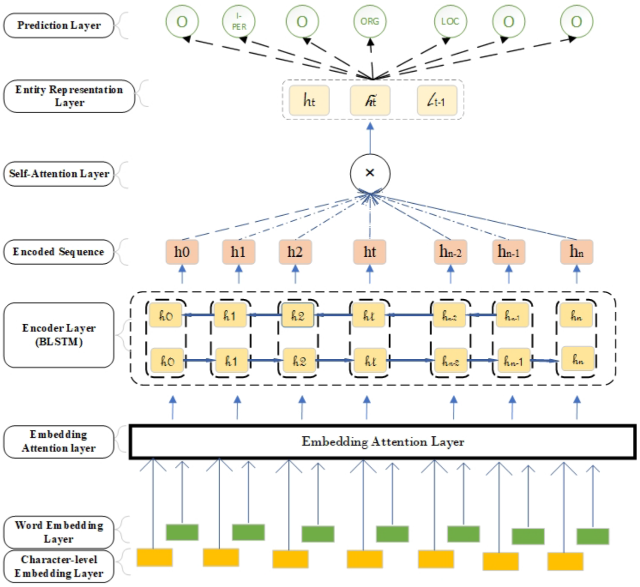

FIGURE 1. The main architecture of the network. Character embedding and Word embedding are given to an attention layer, which works like a gate mechanism to choose the best representation. The encoder unit is a BLSTM whose output is given to another self- attention layer that produces a vector of attention weights, a_z.

A. EMBEDDING LAYER

The work introduced by Mikolov [36] for word distributed representation can substitute for the conventional bag-of-words encoding method and accomplish a superior performance on various NLP tasks. In distributed embeddings, the model is more generalizable, because each word obtains maps to a space, such that semantically similar tokens can have similar vector representations. However, using word embeddings alone as the basic feature representation units can lead to the loss of some accurate information. For languages with rich morphology, such as Arabic, we must capture all morphological and orthographic information. Word embedding encodes semantic and syntactic word relationships, whereas character embedding carries important morphological and shape information. Inspired by this integration, as in [37], we acquire the sequence representations from the character- and word-level embeddings.

1) CHARACTER-EMBEDDING LAYE

Character sequence representations are useful for processing morphologically rich languages such as Arabic and for dealing with the OOV problems, including part of speech tagging and language modeling [38], or dependency parsing [39]. An interesting work was conducted by Kim *et al.* [35], where the author proposed a character-aware neural language model that learns character-level word representations using CNNs, CharCNN. We follow the same technique for generating the character embedding representation. More details about the implementation are discussed in [35].

2) WORD EMBEDDING

Word embedding refers to the representation of words as vectors in a continuous space, thus capturing many syntactic and semantic relations among them. The embeddings are considered fixed constants because they perform superior to handling them as learnable parameters [40]. In the current work, we adopt pre-trained word embeddings, AraVec 2.0 [41], to get the fixed word embedding of each word.

B. EMBEDDING ATTENTION LAYER

In word embedding, words are considered as the smallest unit, and any morphological resemblance between various words is disregarded, thereby resulting in OOV problems. On the contrary, character embedding can operate over individual characters in each word, thereby addressing OOV problems. However, research on character-level embedding is still in the initial phase, and systems that work solely on characters cannot compete with models on word-level on most tasks [34]. Therefore, to maximize the features of character- and word-level embedding, they are concatenated. We adopted an attention-level embedding that works as a gate mechanism to learn similar representations and allow the model to determine how to consolidate the information of each word. When the character feature of each word is obtained, we compute the attention matrix by a weighted sum given by:

$$z = \sigma(U_a \tanh(V_a x + W_a m)), \tag{1}$$

$$\tilde{x} = z.x + (1 - z).m \tag{2}$$

where U_a, V_a, and W_a are the weight matrices for computing the attention matrix z; $\sigma()$ is the sigmoid logistic function with values between zero and one; and x and m are the sequence representations of word and character-level, respectively. The dimension of vector z is x or m, which acts as the weight between the two vectors and permits the model to dynamically determine how much information is required from each other character-level or word-level embeddings.

C. ENCODER

The embedded sequence, which is composed of both embeddings, is inputted to the encoder, a BLSTM-based RNN. The input sequence is processed by the BLSTM either forward or backward in two different hidden states to capture prior and posterior information.

We assume a sentence with m tokens, represented in a series of word embeddings $S = (w_1, w_2, \ldots, w_m)$, where $S_i(1 \leq i \leq m)$ is a vector representing a dimensional word embedding for the ith token. In our encoder layer, the BLSTM is used to encode the tokens. It is composed of a forward LSTM unit that reads the sentence from Si_1 to Si_m and a backward LSTM unit that reads the sentence from Si_m to Si_1, as respectively shown in Equations (3) and (4) below.

$$\overrightarrow{h_t} = \overrightarrow{LSTM}(s_t, \overrightarrow{h_{t-1}}) \tag{3}$$

$$\overleftarrow{h_t} = \overleftarrow{LSTM}(s_t, \overleftarrow{h_{t-1}}) \tag{4}$$

The hidden state h_t can be obtained by concatenating the forward and backward LSTM $h_t = [\vec{h}_t; \overleftarrow{h}_t]$.

Where the number of each unidirectional LSTM hidden unit is u. We denote the entire hidden states as H, where $H \in \mathrm{R}^{n \times 2u}$

$$H_z = (h_1, h_2, \ldots, h_n) \tag{5}$$

D. SELF-ATTENTION LAYER

The objective behind using the attention mechanism is to provide high or low consideration to words based on their involvement in the creation of the sentence meaning [11], [31]. The attention layer is used after the encoding layer, and the output is not passed directly to the prediction layer to encode an adjustable size sentence into a constant length embedding with self-attention technique. This mechanism accepts the entire BLSTM hidden states H_z as input and produces a vector of attention weights, A_z, as output, as shown in Equation 6 below:

$$A_z = softmax\,(w_{s2} \tanh(w_{s1} H_z)) \tag{6}$$

where w_{s2} is a vector of parameters $w_{s2} \in \mathrm{R}^d$, and w_{s1} is a weight matrix of a shape $w_{s1} \in R^{d \times 2u}$, in which d is a hyper-parameter that can be fixed randomly. The softmax () function is applied to ensure that all computed attention weights add up to one.

Given the attention vector A_z, the sentence vector is acquired by weighted A_z as a weighted sum of the LSTM hidden states. The resulting matrix is the sentence embedding.

$$M = A_z H_z \tag{7}$$

E. PREDICTION LAYER

In the self-attention layer, for each token (t), a corresponding entity representation is constructed at every step. The labels or tags for entities are obtained using the equation

$$p\left(y_t \mid X_{seq}, M\right) = softmax\,(\tanh\left(W^{ent} s^{ent} + \mathrm{M} + b^{ent}\right)) \tag{8}$$

where S^{ent} is the concatenation of the current state h_t with the weighted representation of the encoded sequence $\tilde{h}_t$ and the last label embedding l_{t-1}.

A non-linearity function "tanh" is applied, and finally "softmax" function is used to normalize the probability distribution over the entity labels.

$$s^{ent} = [l_{t-1}; \tilde{h}_t; h_t] \tag{9}$$

IV. EXPERIMENT

Extensive experimentations were conducted to validate the methodology. The datasets used and the experimental setup is explained thoroughly in this section.

A. DATASETS

To train and test our ANER system, we evaluated it using two different freely available datasets. First is "ANERCorp," which is a dataset created by Benajiba from several online resources. The ANERCorp dataset is a manually annotated corpus, which is freely available for research purposes. Two corpora, namely, training and testing were used. One person labeled the corpus to ensure the consistency of the annotation, which had 4901 sentences with 150286 tokens[1]. Each line contained a single token for easy parsing. Each word in this dataset was tagged with one of the following tags: person, location, company, and others. The dataset distribution was as follows: Person represents 39%, 30.4% for Location, 20.6% for Organization, and the remaining 10% represents Miscellaneous. The second dataset is "AQMAR Arabic Wikipedia Named Entity Corpus & Tagger," which is developed by Behrang Mohit [42] from four different categories. It has a 74,000-tokens, downloaded from diversified Wikipedia articles and annotated by hand for NEs to guarantee the consistency of the annotation and can be downloaded from this website[2]. We merged the two data sets to obtain a total of 224286 annotated tokens and use them as a single unit dataset.

B. BASELINE

Many approaches have been proposed to address the ANER problems. We selected previous works that are comparable to ours and use the same dataset and evaluation metric. The following are selected to comprise the baseline:

- a work proposed by Kruschwitz and Poesio [43] for the ANER that uses minimally-supervised approach on the same dataset;
- an early approach by Benajiba and Rosso [44], who is the creator of the data set, in which he used Conditional Random Fields; and
- an ANN approach by Naji and Nazlia, who used a neural network technique on the same dataset [45]

C. SETTING

An NVIDIA GeForce GTX1080Ti (12 GB and Intel i7-6800 K 3.4 GHZx12 processor with 32 GB RAM) was used to train the model. It was built on Ubuntu and implemented in the Keras environment. For each token, the model was trained to predict either one of the eight appropriate labels described in Section IV.A. we ran many trails and test experiments to optimize the hypermeters settings. The maximum sequence length was set to 100, the embedding dimension was fixed to 100, and the hidden-state size was set to 200. The combination of forward and backward LSTM resulted in a dimension of 400. For the objective function, we experiment with different activation functions, namely, softmax, tanh, cross-entropy, ReLu, and sigmoid. Evidently, categorical cross-entropy performs superior among the activation functions tested; hence, it is adopted in our model. L2-regularization component was added to the cost function for tuning the output. For the over-fitting problem, 50% dropout was used as an additional measure to control the inputs in the LSTM network and the Softmax layer. among the different available optimization functions (Adam, Adagrad, and RMSprop), Adagrad yields the best result when used in our model. The batch sizes were tuned to 128. For the number of epochs, we test different values ranging from 5 to 40, and the best result is obtained with 30 epochs.

TABLE 1. Results of the base Model with BLSTM and concatenated word embedding.

	P	R	F
person	80.71	78.16	79.41
organization	85.11	86.7	85.90
location	87.54	86.41	86.97
other	84.32	83.12	83.72
overall	84.42	83.60	84.01

D. EVALUATION

The performance of the system was evaluated using precision (P), recall (R), and F-measure score (F) as the evaluation metrics. These are the standard measures for NER.

$$precision(P) = \frac{X}{Y} \tag{10}$$

$$Recall(R) = \frac{X}{Z} \tag{11}$$

$$F - measure(F) = \frac{2 \times Precision \times Recall}{Precision + Recall} \tag{12}$$

X is the number of accurately extracted entities, Y is the total number of recognized entities, and Z is the total number of accurate entities existing in the dataset.

V. RESULT

We ran our experiments on the merged "ANERcorp" and "AQMAR" datasets. The model predicted NEs as a person, location, organization, and other. Excellent accuracy is achievable without using any feature engineering or handcrafted rules. RNN shows the ability to effectively handle sequence labeling problems without requiring additional information, such as Chunks or Gazetteers, which are essential especially for NER task. To test the importance of character-level features on the model's performance, we initially ran the model on a baseline setup that concatenate word embeddings and character features directly. The results are shown in Table 1.

Then, we have combined both word and character embeddings feature through the embedding-level attention mechanism. The model can dynamically choose between the two embeddings and obtain a richer sequence representation, therefore significantly improves the performance of the model as shown in Table 2.

Afterward, we added the self-attention layer above the encoder and conducted several experiments to tune the system parameters and evaluate its prediction performance. The results are shown in Table 3.

[1] http://curtis.ml.cmu.edu/w/courses/index.php/ANERcorp

[2] http://www.ark.cs.cmu.edu/AQMAR/

TABLE 2. Results of the base Model with BLSTM and embedding attention layer.

	P	R	F
person	83.12	80.66	81.87
organization	87.31	88.42	87.86
location	90.04	90.73	**90.38**
other	86.94	84.24	85.57
overall	**86.85**	**86.01**	**86.43**

TABLE 3. Results of the model with the self-attention unit.

	P	R	F
person	88.12	89.32	88.72
organization	90.52	89.05	89.78
location	91.15	90.24	90.69
other	87.54	86.44	86.99
overall	**89.33**	**88.76**	89.05

TABLE 4. Results of the full model implementation.

	P	R	F
person	90.45	91.67	91.06
organization	91.76	90.15	90.95
location	92.89	93.86	93.37
other	89.58	90.11	89.84
overall	**91.17**	**91.45**	91.31

TABLE 5. Comparative results of our model concerning the baseline system.

Model	Entity type	Precision	Recall	F-measure
Maha Althobaiti[44]	person	77.87	63.86	70.17
	Location	81.52	59.86	69.03
	organization	95.44	34.31	50.47
Benajiba (ANERsys 2.0)[45]	person	56.3	48.6	52.1
	Location	91.7	82.2	86.7
	organization	48	45.0	46.4
Nazlia Omar[46]	person	93.10	55.90	69.90
	Location	29.50	81.10	43.30
	organization	72.50	50	59.20
OUR Model	person	90.45	91.67	91.06
	Location	91.76	90.15	90.95
	organization	92.89	93.86	93.37

Finally, we tested the full model implementation, which included the embedding attention layer above the character- and word-level embeddings, as well as the self-attention layer above the encoder unit. The state-of-the-art results indicate great improvement in the performance of the model. Table 4 summarizes the result of the full model implementation.

Furthermore, given these results, we provide the following observations:

- The inclusion of character embedding has a considerable impact on the performance of the model prediction, because it improves the handling of any previously unseen tokens, especially for the Arabic language with rich morphology and low resource availability. Tables 2 and 3 show the results of the Model with and without the inclusion of the character-level embedding, respectively. The defect is clearly visible.
- The use of the embedding attention layer mechanism improves the system performance remarkably. The system does not require computing word and character embeddings simultaneously. However, it dynamically decides how much information is required from the character- or word -level embedding, thereby providing flexibility to select the best representation from either embedding.
- For the encoder layer, we have tested BLSTM and BGRU and found that the performance is almost similar to a slightly better performance for LSTM, and less running time for GRU due to the simple structure of GRU.
- We analyze the performance of our model using a traditional feature-based baseline system. We enlarged the main dataset "ANERCorpus" in the baseline systems and merged it with the "AQMAR" dataset. However, our system still outperforms their performance in a considerable margin, as shown in Table 4.
- Finally, our model with multi-attention layer and character embedding leads to better sequence labeling prediction, which in turn, yields state-of-the-art performance on the ANER task.

VI. CONCLUSION

In this study, we investigate a neural network model with a multi-attention layer to extract Arabic NEs. Two attention units are used. The first is the embedding attention layer, which is used to select the best word representation from character and word embeddings. This approach remarkably improves the system performance, especially for labeling previously unseen words. The other one is the self-attention unit, which is used above the encoder layer to boost the model performance by focusing on words with more meanings that contribute to the labeling prediction task. Our model achieves approximately matched F1 score of 91% on the "ANERCorpus" and surpasses the existing state-of-the-art approaches for ANER by a notable margin. Furthermore, we find that deep neural networks can accomplish competitive performance with less work on feature engineering for a language with rich morphology, such as Arabic, and less dependence on external resources, such as gazetteers or chunks. Moreover, the attention mechanism plays an essential role in improving not only NER and sequence labeling task models, but also other NLP tasks. Our future work aims to extend our approach to rely more on attention mechanism and attempt different attention architectures.

REFERENCES

[1] D. Nadeau and S. Sekine, "A survey of named entity recognition and classification," *LingvisticÃę Investigationes*, vol. 30, no. 1, pp. 3–26, Jan. 2007.

[2] K. Shaalan and M. Oudah, "A hybrid approach to Arabic named entity recognition," *J. Inf. Sci.*, vol. 40, no. 1, pp. 67–87, Feb. 2014.

[3] M. K. Nydell, *Understanding Arabs: A Guide for Modern Times*. London, U.K.: Hodder & Stoughton, 2018.

[4] A. Farghaly and K. Shaalan, "Arabic natural language processing: Challenges and solutions," *ACM Trans. Asian Lang. Inf. Process.*, vol. 8, no. 4, Dec. 2009, Art. no. 14.

[5] A. Zirikly and M. Diab, "Named entity recognition for arabic social media," in *Proc. 1st Workshop Vector Space Modeling Natural Lang. Process.*, pp. 176–185, 2015.

[6] K. Shaalan, "A survey of arabic named entity recognition and classification," *Comput. Linguistics*, vol. 40, no. 2, pp. 469–510, Jun. 2014.

[7] K. Shaalan and H. Raza, "Arabic named entity recognition from diverse text types," in *Advances in Natural Language Processing* (Lecture Notes in Computer Science), vol. 5221. Berlin, Germany: Springer, 2008, pp. 440–451.

[8] Y. Benajiba and P. Rosso, "ANERsys 2.0: Conquering the NER task for the Arabic language by combining the maximum entropy with POS-tag information," in *Proc. 3rd Indian Int. Conf. Artif. Intell. (IICAI)*, Dec. 2007, pp. 1814–1823.

[9] M. Tomas, "Statistical language models based on neural networks," Brno Univ. Technol., Brno, Czech Republic, Tech. Rep., Apr. 2012.

[10] M. Habibi, L. Weber, M. Neves, D. L. Wiegandt, and U. Leser, "Deep learning with word embeddings improves biomedical named entity recognition," *Bioinformatics*, vol. 33, no. 14, pp. i37–i48, Jul. 2017.

[11] K. Al-Sabahi, Z. Zuping, and M. Nadher, "A hierarchical structured self-attentive model for extractive document summarization (HSSAS)," *IEEE Access*, vol. 6, pp. 24205–24212, 2018.

[12] M.-T. Luong, H. Pham, and C. D. Manning. (2015). "Effective approaches to attention-based neural machine translation," [Online]. Available: https://arxiv.org/abs/1508.04025

[13] C. dos Santos, M. Tan, B. Xiang, and B. Zhou. (2016). "Attentive pooling networks." [Online]. Available: https://arxiv.org/abs/1602.03609

[14] S. Abdallah, K. Shaalan, and M. Shoaib, "Integrating rule-based system with classification for arabic named entity recognition," in *Proc. Int. Conf. Intell. Text Process. Comput. Linguistics*, 2012, pp. 311–322.

[15] W. Zaghouani, "RENAR: A rule-based Arabic named entity recognition system," *ACM Trans. Asian Lang. Inf. Process.*, vol. 11, no. 1, Mar. 2012, Art. no. 2.

[16] K. Shaalan and H. Raza, "NERA: Named entity recognition for Arabic," *J. Assoc. Inf. Sci. Technol.*, vol. 60, no. 8, pp. 1652–1663, Aug. 2009.

[17] Y. Benajiba, P. Rosso, and J. M. BenedíÂruiz, "Anersys: An arabic named entity recognition system based on maximum entropy," in *Proc. Int. Conf. Intell. Text Process. Comput. Linguistics*, 2007, pp. 143–153.

[18] M. M. Oudah and K. Shaalan, "A pipeline arabic named entity recognition using a hybrid approach," in *Proc. COLING*, Dec. 2012, pp. 2159–2176.

[19] Y. Benajiba, M. Diab, and P. Rosso, "Arabic named entity recognition: An SVM-based approach," in *Proc. 2008 Arab Int. Conf. Inf. Technol. (ACIT)*, 2008, pp. 16–18.

[20] A. J. Masino, D. Forsyth, and A. G. Fiks, "Detecting adverse drug reactions on twitter with convolutional neural networks and word embedding features," *J. Healthcare Inform. Res.*, vol. 2, nos. 1–2, pp. 25–43, Jun. 2018.

[21] S. K. Sahu and A. Anand, "Drug-drug interaction extraction from biomedical texts using long short-term memory network," *J. Biomed. Inform.*, vol. 86, pp. 15–24, Oct. 2018.

[22] P. O. Pinheiro and R. Collobert, "Recurrent convolutional neural networks for scene labeling," in *Proc. 31st Int. Conf. Int. Conf. Mach. Learn. (ICML)*, Jun. 2014, pp. I-82–I-90.

[23] Z. Zhao *et al.*, "Disease named entity recognition from biomedical literature using a novel convolutional neural network," *BMC Med. Genomics*, vol. 10, no. 5, p. 73, Dec. 2017.

[24] P. Ding, X. Zhou, X. Zhang, J. Wang, and Z. Lei, "An attentive neural sequence labeling model for adverse drug reactions mentions extraction," *IEEE Access*, vol. 6, pp. 73305–73315, 2018.

[25] O. Kuru, O. A. Can, and D. Yuret, "Charner: Character-level named entity recognition," in *Proc. 26th Int. Conf. Comput. Linguistics, Tech. Papers (COLING)*, Dec. 2016, pp. 911–921.

[26] L. Li, L. Jin, Z. Jiang, D. Song, and D. Huang, "Biomedical named entity recognition based on extended recurrent neural networks," in *Proc. IEEE Int. Conf. Bioinf. Biomed. (BIBM)*, Nov. 2015, pp. 649–652.

[27] A. N. Jagannatha and H. Yu, "Structured prediction models for RNN based sequence labeling in clinical text," in *Proc. Conf. Empirical Methods Natural Lang. Process.*, Nov. 2016, p. 856.

[28] L. Ouyang, Y. Tian, H. Tang, and B. Zhang, "Chinese named entity recognition based on B-LSTM neural network with additional features," in *Proc. Int. Conf. Secur., Privacy Anonymity Comput., Commun. Storage*, Dec. 2017, pp. 269–279.

[29] A. N. Jagannatha and H. Yu, "Bidirectional RNN for medical event detection in electronic health records," in *Proc. Conf. Assoc. Comput. Linguistics North Amer. Chapter Meeting*, Jun. 2016, p. 473.

[30] J. P. C. Chiu and E. Nichols, "Named entity recognition with bidirectional LSTM-CNNs," *Trans. Assoc. Comput. Linguistics*, vol. 4, pp. 357–370, Dec. 2016.

[31] Z. Lin *et al.* (2017). "A structured self-attentive sentence embedding." [Online]. Available: https://arxiv.org/abs/1703.03130

[32] S. Ramamoorthy and S. Murugan. (2018). "An attentive sequence model for adverse drug event extraction from biomedical text." [Online]. Available: https://arxiv.org/abs/1801.00625

[33] W. Zheng *et al.*, "An attention-based effective neural model for drug-drug interactions extraction," *BMC Bioinf.*, vol. 18, no. 1, p. 445, Oct. 2017.

[34] M. Rei, G. K. O. Crichton, and S. Pyysalo. (2016). "Attending to characters in neural sequence labeling models." [Online]. Available: https://arxiv.org/abs/1611.04361

[35] Y. Kim, Y. Jernite, D. Sontag, and A. M. Rush, "Character-aware neural language models," in *Proc. 30th AAAI Conf. Artif. Intell. (AAAI)*, Mar. 2016, pp. 2741–2749.

[36] T. Mikolov, K. Chen, G. Corrado, and J. Dean. (2013). "Efficient Estimation of Word Representations in Vector Space." [Online]. Available: https://arxiv.org/abs/1301.3781 https://arxiv.org/abs/1301.3781

[37] F. Li, M. Zhang, G. Fu, and D. Ji, "A neural joint model for entity and relation extraction from biomedical text," *BMC Bioinf.*, vol. 18, no. 1, p. 198, Mar. 2017.

[38] W. Ling *et al.* (2015)."Finding function in form: Compositional character models for open vocabulary word representation." [Online]. Available: https://arxiv.org/abs/1508.02096

[39] M. Ballesteros, C. Dyer, and N. A. Smith. (2015). "Improved transition-based parsing by modeling characters instead of words with LSTMs." [Online]. Available: https://arxiv.org/abs/1508.00657

[40] A. Cocos, A. G. Fiks, and A. J. Masino, "Deep learning for pharmacovigilance: Recurrent neural network architectures for labeling adverse drug reactions in Twitter posts," *J. Am. Med. Inform. Assoc.*, vol. 24, no. 4, pp. 813–821, Jul. 2017.

[41] A. B. Soliman, K. Eissa, and S. R. El-Beltagy, "AraVec: A set of arabic word embedding models for use in arabic NLP," *Procedia Comput. Sci.*, vol. 117, pp. 256–265, Jan. 2017.

[42] B. Mohit, N. Schneider, R. Bhowmick, K. Oflazer, and N. A. Smith, "Recall-oriented learning of named entities in arabic Wikipedia," in *Proc. 13th Conf. Eur. Chapter Assoc. Comput. Linguistics*, Apr. 2012, pp. 162–173.

[43] M. Althobaiti, U. Kruschwitz, and M. Poesio, "Combining minimally-supervised methods for arabic named entity recognition," *Trans. Assoc. Comput. Linguistics*, vol. 3, no. 1, pp. 243–255, 2015.

[44] Y. Benajiba and P. Rosso, "Arabic named entity recognition using conditional random fields," in *Proc. Workshop HLT NLP Arabic World, LREC*, May 2008, pp. 143–153.

[45] N. F. Mohammed and N. Omar, "Arabic named entity recognition using artificial neural network," *J. Comput. Sci.*, vol. 8, no. 8, pp. 1285–1293, Aug. 2012.

[46] G. Lample, M. Ballesteros, S. Subramanian, K. Kawakami, and C. Dyer. (2016). "Neural architectures for named entity recognition." [Online]. Available: https://arxiv.org/abs/1603.01360

[47] E. Strubell, P. Verga, D. Belanger, and A. McCallum. (2017). "Fast and accurate entity recognition with iterated dilated convolutions." [Online]. Available: https://arxiv.org/abs/1702.02098

MOHAMMED NADHER ABDO ALI received the B.Sc. degree in computer science from Mysore University, Mysore, India, in 2009, and the M.Sc. degree in computer science from VIT University, Vellore, India, in 2011. He is currently pursuing the Ph.D. degree in computer science with the School of Information Science and Engineering, Central South University, Changsha, Hunan, China. His research interests include deep learning, natural language processing, and data mining.

GUANZHENG TAN received the B.Sc. degree from the Department of Aeronautical Engine, Nanjing Aeronautical Institute, Nanjing, China, in 1979, the M.Sc. degree from the Department of Automatic Control, National University of Defense Technology, Changsha, China, in 1988, and the Ph.D. degree from the Department of Mechanical Engineering, Nanjing University of Aeronautics and Astronautics, Nanjing, in 1992. From 2004 to 2005, he was a Visiting Professor with the School of Computer Science, University of Birmingham, U.K. He is currently a Professor with the School of Information Science and Engineering, Central South University, Changsha. His research interests include artificial intelligence and robotics, intelligent systems, and intelligent control.

AAMIR HUSSAIN received the Ph.D. degree in computer science and technology from the School of Computer Science and Technology, Wuhan University of Technology, China, in 2016. He is currently an Assistant Professor with the Department of Computer Science, Muhammad Nawaz Shareef University of Agriculture Multan, Pakistan. His research interests include wireless body area networks, the Internet of Things, and software-defined networks (SDN).